International Journal of Advanced Technology and Engineering Exploration, Vol 10(105)
ISSN (Print): 2394-5443 ISSN (Online): 2394-7454
http://dx.doi.org/10.19101/IJATEE.2023.10101411

Research Article

Proactive DDoS attack detection in software-defined networks with Snort rule-based algorithms

Nor Shahniza Kamal Bashah[1*], Twiene Selynda Simbas[2], Norjansalika Janom[1] and Syaripah Ruzaini Syed Aris[1]
School of Computing Sciences, College of Computing, Informatics and Mathematics, Universiti Teknologi MARA, 40450 Shah Alam, Selangor, Malaysia[1]
Dell Global Business Center Sdn. Bhd., 2900, Persiaran Apec, Cyberjaya, 63000 Cyberjaya, Selangor, Malaysia[2]

Received: 17-April-2023; Revised: 01-August-2023; Accepted: 03-August-2023

Abstract
The exponential growth of application-layer programs has imposed significant constraints on the existing underlying network infrastructure. To address this escalating demand, a transition towards a software-oriented network infrastructure becomes indispensable. Software-defined networks (SDN), which decouples the data and control planes, transforming them into a programmable network controlled by a central controller, emerges as the solution. This approach enhances network management, leading to reduced operational expenditures (OPEX), heightened quality of service, and the achievement of desired scalability. However, the shift towards a programmable network infrastructure exposes vulnerabilities to existing security threats. In this research, additional security measures were proposed with the aim of detecting and preventing security threats, particularly distributed denial of service (DDoS) attacks. For simulation purposes, the Mininet platform is employed. The Ryu controller is configured as an SDN controller, responsible for transmitting and removing OpenFlow messages to and from switches, along with handling incoming packets. Snort plays a crucial role in analyzing suspicious traffic entering the network. This incoming traffic undergoes examination based on predefined rules, triggering an alert if any traffic matches these rules. The internet control message protocol (ICMP) flooding method was employed to execute DDoS attacks. Based on the results and findings, an extensive volume of packets was observed during attacks on the SDN network. Furthermore, connectivity tests conducted through ping tests towards the targeted machine resulted in 100% packet loss. This outcome signified the denial of resource access on the targeted machine during an attack, consequently leading to a decline in overall network performance. Analysis of the amassed data revealed that early detection through rule-based Snort implementation could significantly mitigate the impact on SDN networks. Consequently, the adoption of Snort for proactive DDoS attack detection in SDN networks was proposed. This approach empowered network administrators to respond promptly upon the occurrence of a Snort-generated alert.

Keywords
Security attack, Network degradation, Proactive detection, Rule-based algorithm, Snort alert.

1.Introduction
In year 2020, the whole world was lockdown due to the Covid-19 pandemic. Even now, some countries are still struggling to flatten the statistical curve of their Covid-19 cases [1, 2]. However, this does not prevent cybercriminals from taking advantage of this situation. The increase in cyber security attacks during this pandemic led to a new term called cyber security pandemic [3–6]. In addition, cyber-related activities began to gain world attention because of the Israeli-Palestinian conflict, which is cyber war [7].

During the conflict, both sides use cyber warfare as one of the methods to attack each other. This becomes possible, because many systems or applications are connected to the Internet and its programmable nature. So, there is a need to secure software-defined networks (SDN) infrastructure from being exploited.

SDN is an approach to design, build and manage networks by separating the network control plane (brain) and forwarding plane (muscle) [8, 9]. In traditional networks, the control plane and the forwarding plane are handled by the device itself

*Author for correspondence

physically. This, in return, exhausts resources as it expands a significant amount on computing [10]. Decoupling the control plane from the data plane shifts management and automation to the control plane. This control plane is programmable and serves as the center for processing and computing, effectively acting as the brain. The data plane, on the other hand, solely focuses on forwarding or executing the actions programmed by the control plane. With this arrangement, the management of these components can be centralized and done through a control console, avoiding the need to access the device itself. Automation makes the configuration of network elements easier. By separating the data plane and the control plane, the network infrastructure is no longer physically bound but transforms into programmable software. Although many SDN systems have been widely adopted and new ones introduced, one certainty remains: it attracts attackers who target the system.

According to the open networking foundation (ONF) in their release paper for the OpenFlow specification v 1.5.1 it states that the default security mechanism for the OpenFlow protocol [11] is transport layer security (TLS) to encrypt the communication channel between switches and controllers. Another alternative is to use the regular transmission control protocol (TCP); however, this is not recommended as data traffic won't be encrypted. It is the channel that needs to be protected and based on the specifications stated above; it is usually encrypted using TLS but may also run directly over TCP. OpenFlow communication protocol is widely used in SDN for communication between controller and switch. Despite the TLS enabling in the SDN infrastructure, this only secures the communication channel between the controller and the switch [12]. By securing this communication channel alone does not mean that SDN are not vulnerable or immune to any security attack. National Security Council Malaysia [13] mentioned that SDN is still vulnerable to various security threats, such as distributed denial of service (DDoS) attacks and man-in-the-middle attacks. With this threat, attackers can launch attacks and compromise the confidentiality, integrity, and availability of data in computer networks. *Figure 1* shows the types of attacks that a hacker can launch at each layer that exists in the SDN architecture. These layers only exist in SDN because of the decoupling of the control plane and the data plane, which allows the network to be switched to a programmable network.

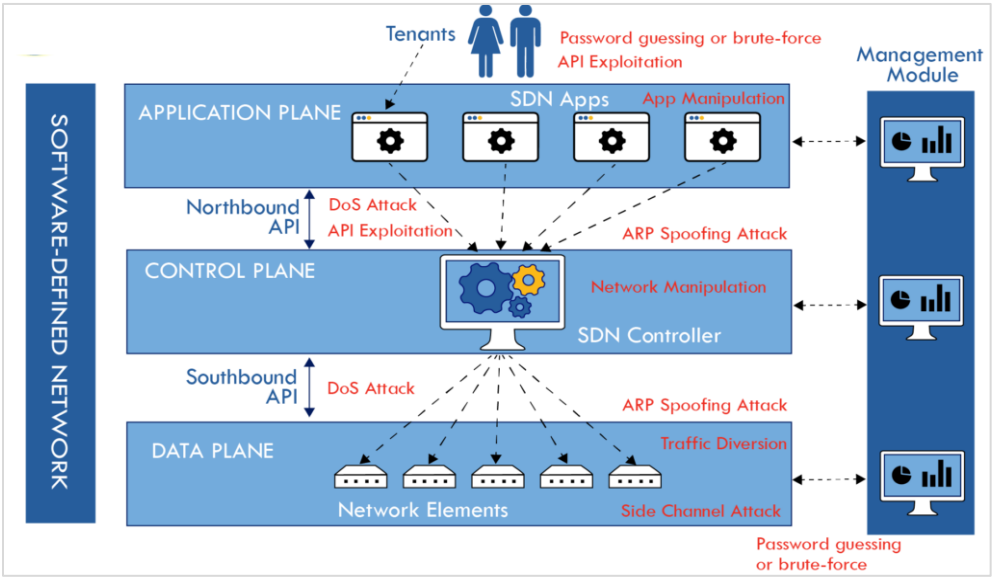


Figure 1 Type of attacks or threats on each later of SDN network architecture

Among all the attacks or threats listed in *Figure 1*, the Verizon Data Breach Report presented in *Figure 2* indicates that in 2020, the highest number of reported security breach incidents originated from denial of service (DoS) hacking attacks [14]. The report also highlights that the most common tactic employed by attackers involved sending a large number of packets with the objective of denying access to the targeted machine.

DoS attacks rank among the top 5 most frequently reported security breach incidents within schools

categorized under the K-12 system [15]. K-12 refers to the educational framework encompassing kindergarten through 12th grade. As shown in *Figure 3*, 5% of the reported cyber-attack incidents are attributed to DoS attacks. According to the report, the pandemic triggered a significant increase in targeting online learning systems, given the widespread adoption of virtual classes and meetings through platforms like Zoom. The school/district community primarily experienced incidents such as classroom intrusions, meeting intrusions, and email intrusions.

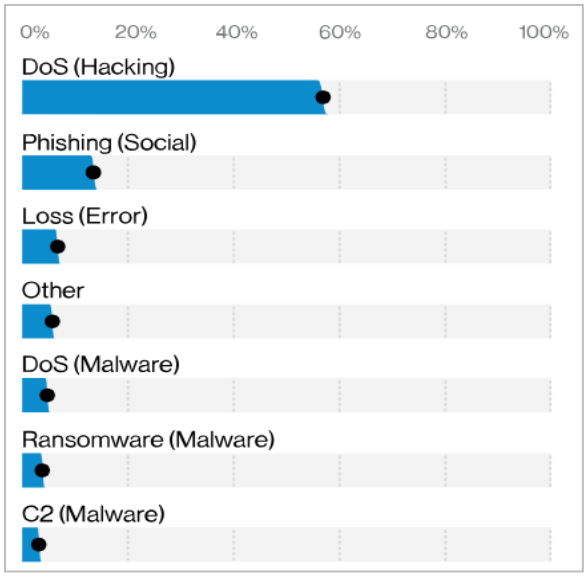


Figure 2 Top threat action varieties in incidents reported

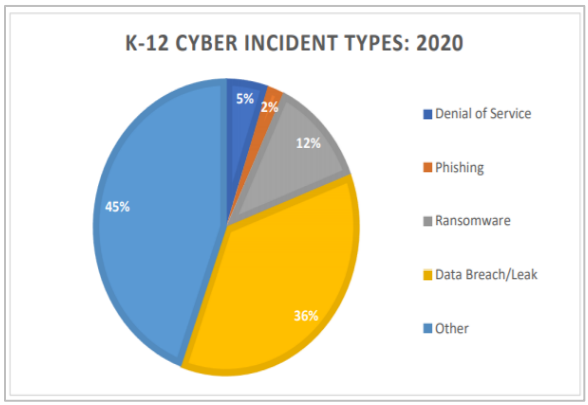


Figure 3 Count of incidents reported by k-12 school in 2020

The challenge with detecting DoS/DDoS attacks lies in the fact that the volume of traffic can resemble normal network activity [16–19]. For instance, high traffic during peak business hours can explain the network's increased activity.

Another situation arises when launching or selling a new product, leading to higher transactions that generate increased server traffic. DDoS attacks can disrupt network availability by inundating the victim with an excessive amount of unauthorized traffic, overwhelming its capacity and hindering the passage of legitimate data [20]. Moreover, DDoS attacks are increasingly jeopardizing cyber-physical systems (CPSs) due to their ease of execution and the havoc they cause. Additionally, due to the constant evolution of attack techniques, a method is urgently needed to defend against both known and unknown DDoS attacks [21].

Although some existing research endeavors attempt to address the issues, such as [22], which introduced a two-stage DDoS detection system in SDN using triggers with multiple features and self-adaptive thresholds, [23], which proposed a solution for two types of DDoS attacks—specifically Slowloris in an SDN environment, and [24], which put forth an AI-based DDOS attack detection method in SDN networks, a deficiency still exists, particularly in the domain of alerts. Consequently, a proactive notification system for imminent attacks remains lacking. Equipping network operators with alert notification tools for detecting suspicious inbound activity is vital. This empowers them to investigate the origins of the threat and respond promptly to potential attacks.

Hence, the focus of this research is to establish a proactive DDoS attack detection method in SDN using the Snort rule-based algorithm. Snort possesses the ability to constantly monitor the SDN network, triggering alerts whenever traffic matches the predefined rule criteria. These alerts serve as early warnings for network administrators, affording them time to formulate appropriate responses. The proactive monitoring tool plays a pivotal role in safeguarding the SDN network from collapsing under the pressure of a DDoS attack. The test results analyzed within this research further enhance our understanding of how DDoS attacks impact SDN network performance and availability.

The literature review is discussed in section 2. Next, the method section is explored in section 3. The results of the simulation tests are then thoroughly analyzed and described in section 4. Discussions are

conducted in section 5, followed by conclusions and recommendations for future work.

2.Literature review

Due to the increasing complexity of traditional network management and operations, when the network scale increases, it leads to a decrease in service quality and lowers network performance. Therefore, it becomes inevitable to switch to SDN network that offer flexibility due to their programmability [25–28]. By securing the communication channel between the controller and the switch using the TLS protocol, it ensures encrypted and secure communication. Although the ONF mentions in the standard OpenFlow specification, plain TCP can be used as an alternative to TLS, and ONF should make it mandatory to use TLS. As network operators, they should not ignore the importance of TLS to secure communication channels, as it provides an additional layer of protection to SDN networks.

With the current trends in cyber-attacks, it is important to protect SDN networks from possible security treat that exist out there. Due to its programmable features, the network becomes vulnerable. Attackers can launch DDoS attacks on controllers in isolation or target OpenFlow switches and even hosts in the network. A successful DDoS attack can degrade overall network performance and affect network availability. One major drawback of making the controller the brain of an SDN network is that it makes the controller a single point of failure. A successful attack on the controller may bring down the entire SDN network. Once the controller is down, no component in the SDN network can make decisions and computations.

What makes DDoS dangerous and devastating is that it takes time to detect the system under a DDoS attack. Sometimes the increase in traffic volume seen on the monitoring tool does not mean the network is under attack, it may be caused by more users needing to access the resource at that time. Recommended snort as an intrusion detection system (IDS) as one of the solutions against DDoS attacks in SDN network [29]. With the IDS enabled in the SDN network, it can detect incoming DDoS attacks and alert system administrators for further actions to stop the DDoS attack before it gets worse and becomes unsustainable.

There are also some researchers that proposed k-means in IDS. Proposed k-means and support vector machine (SVM) as a hybrid method by using CIC-IDS2017 dataset to efficiently conduct comprehensive intrusion detection focus in vehicular ad hoc network (VANET) [30]. According to the findings of this study, the proposed approach can enhance detection accuracy, increasing the effectiveness of IDS even more. Despite that, they intend to implement the solution in various real-world contexts, such as the internet of vehicles (IoV), in the future and analyze its performance. Other than that, [31] proposed network intrusion detection in network security laboratory – knowledge discovery in databases (NSL-KDD) dataset using improved genetic k-means algorithm. They reported that the research the improved genetic k-means algorithm is more efficient than the k-means++ approach for intrusion detection. Also, the methods used in this research shows that they performed way better in large dataset compared to small dataset that illustrate lesser accuracy. In addition, there is also research proposed the comparison of intrusion detection by k-means and fuzzy c-means focusing on NSL-KDD dataset. Mainly focus to find incursion in the 20% of NSL-KDD dataset to accurately discover the primary attack categories, such as DoS, user-to-root (U2R), Probe and remote-to-local (R2L) that carried out in MATLAB [32]. They reported that the fuzzy c-means method is capable of detecting network attacks with a maximum attack rate of 45.95% for 28 features. Also, they also stated that to improve the percentage of attacks detected, hybrid variations of clustering algorithms to identify intrusions in a network could be developed. In order to analyze whether the incoming traffic data are normal, [33] used the k-means approach from Spark's machine learning library on knowledge discovery in databases (KDD) Cup99. From this method, 10 abnormal behaviours of 400 thousand traffic data have been detected. For future work, researchers suggested that different methods should be used with KDD Cup99.

Meanwhile, [34] proposed fast k-nearest neighbors (kNN) classifier on cloud environment conducted on CICIDS2017 dataset. They claimed that the fuzzy k-nearest neighbor (FkNN) classifier is a better classifier that requires less detection time while maintaining accuracy compared to traditional kNN for mitigating assaults and reducing concentrating solar power (CSP) economic losses occurred in this research in a shorter time frame. Regardless, they recommended that it be implemented and compared to other classifiers in a real-time context. Other than that, [35] proposed machine learning which kNN is one of them based IDSs by using CICIDS2017

dataset. Based on accuracy, precision, recall, and F1-score, kNN is the best classifier, while Naïve Bayes and quadratic discriminant analysis (QDA) have the worst performance, despite their short training and prediction timeframes. Using kNN classification and k-means clustering techniques, [36] give statistical analysis of the labelled network flow based CIDDS-001 dataset. They claimed that in terms of employed prominent metrics, both kNN classification and k-means clustering perform well separately over the CIDDS-001 dataset. It is stated that different machine learning approaches can be used to enhance the analysis of this dataset. Next, density peaks technique to intrusion detection was also applied by [37] who used a standard dataset, KDD Cup99, to identify attacks more efficiently and introduce density in K-NN. In terms of average accuracy, they believe that this technique outperforms traditional K-NN. Nonetheless, they advised devising a new strategy for dealing with the classification of specific attacks.

On the other hand, [38] improved a detection method based on outlier detection by using modified k-means and K-NN. Based on KDD Cup99 testing, they claimed that the suggested method surpasses other current methods by a large margin, proving its superiority over previous techniques. Next, [39] proposed a hybrid k-means and kNN algorithm to accurately minimise the system's time complexity. The KDD Cup99 data collection is used to develop this model. The accuracy of the kNN algorithm based on k-means in classification of normal and assaults is good, according to experimental results. In addition, the model training time of a K-means and kNN algorithm-based IDS is more relevant in today's massive data amount of network intrusion databases.

Apart from using IDS for security measures, there are other solutions that other researchers have suggested. For example, in [40], the focus of the research is on the security of the SDN network. The researcher proposed an effective and scalable security framework, namely LINK-GUARD which is used for facilitating secure link discoveries in an SDN network. The framework is designed to detect and thwart link fabrication attacks (LFAs), thus reducing the risks of network topology poisoning. From the performance evaluation result indicates that the LINK-GUARD can effectively and efficiently secure topology discoveries against both host-based and switch-based link fabrication attacks, thus promotes a scalable solution for dynamic and large SDN networks.

In [41], the emphasis is also placed on security threats that have the potential to jeopardize the infrastructure of SDN networks. They meticulously discussed all existing security threats and expounds on their impact on SDN networks. However, the researchers also propose a solution to address this security concern by enhancing the TLS protocol itself to secure the communication channel. Furthermore, there is research that specifically concentrates on the risks associated with DDoS attacks on SDN networks.

In [42], researchers concluded that a DDoS attack on an SDN controller can cause the controller to become idle and unavailable to legitimate users. Conclusions are made based on the results of their simulations using Pox and Mininet controllers. However, no mitigation for the attack was proposed. In [43], the researcher implemented techniques for enhancing the security of south bound infrastructure in SDN which includes OpenFlow switches and end hosts. The techniques focus on validation and secure configuration of flow rules in the OpenFlow switches by trusted SDN controller in the domain, securing the flows from the end hosts and detecting attacks on the switches by malicious entities in the SDN domain. However, the research does not provide alert message towards the attack. Meanwhile, in [44] the research presents SDN-based security framework, which automates the monitoring, detection, and mitigation of slow-rate DDoS attacks. The framework is implemented in a physical network that uses equipment from the European Experimental Facility Smart Networks for Industry (SN4I). However, the research only focuses for slow-rate DDoS attacks containing traffic generated using the SN4I facility.

3.Method

Simulation technique was used to carry out this research. Three main components are used to build a complete environment of the simulation namely the controller, the data plane, and the IDS. The controller is the brain of the ecosystem while the data plane acts as a switch emulated by Mininet. IDS is responsible for conducting initial detection of any possible incoming DDoS attacks. For this, Snort was used and installed in the same virtual machine (VM) where Ryu's controller resides [45]. *Figure 4* shows components of snort which are decoder, preprocessor, detection and alert and logging system.

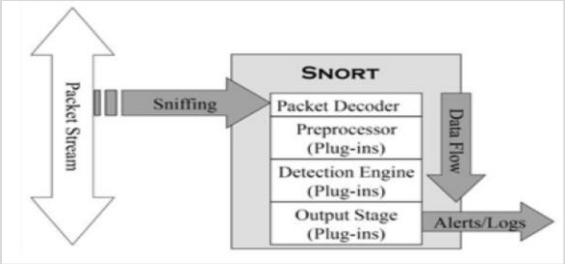


Figure 4 Snort component

Decoder

The first component is decoder. It uses or responsible for forming packets and determine the place and packet size to be fully utilize by next component. However, decoder also able to identify any anomalies header such as invalid dimensions that may contain dangerous packet which then caused alerts to be generated.

Preprocessor

The second component is preprocessor. It uses to take packets and check against a set of plug-ins such as scanner and hypertext transfer protocol secure (HTTPS). These plug-ins or protocols check for certain type of behavior from the packet and forward it to detection component. It is then used to identify and distinguish between a good packet and a bad packet.

Detection

The third component is the detection engine. This component is utilized to examine the data based on the pre-defined sets of rules. If the rules match the data in the packet, they are sent to the alert and logging system, which then categorizes them as malicious packet or data. It's important to note that Snort has specific syntax for its sets of rules, including protocol types, length, or headers. During this phase, the pre-defined sets of rules can be configured to block any suspicious packets, thus preventing them from traversing the network.

Alert and logging system

The final component of Snort is the alert and logging system. When the packet or data is processed by the detection engine and matches a pre-defined set of rules, it triggers an alert for the system administrator to take appropriate action. The process flow of the IDS for monitoring malicious packets is depicted in *Figure 5*.

In addition to Snort, Wireshark is used for traffic analysis. To capture observations and outcomes, five distinct scenarios are simulated. Each scenario utilizes a different type of Ryu application manager based on specific requirements. *Figure 6* illustrates the network topology diagram used for these five scenarios. The distinguishing factor among these scenarios lies in the compromised host and the victim host configurations.

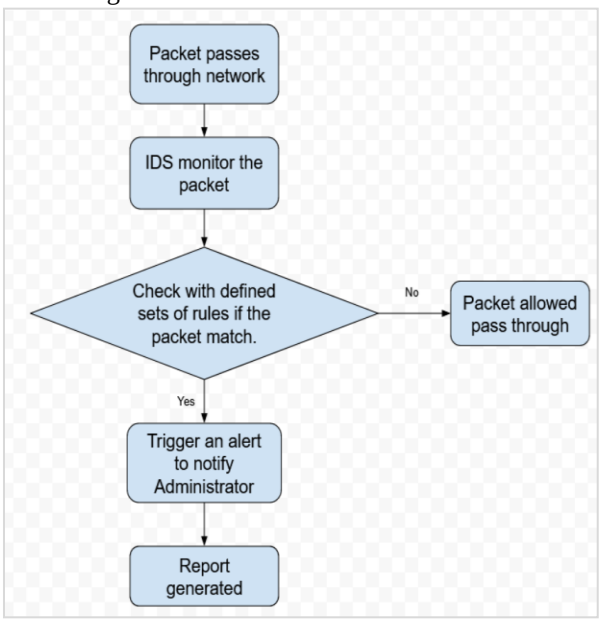


Figure 5 IDS process flow

The following are the 5 scenarios breakdown:

Scenario 1: Baseline scenario

In this scenario, a normal ping test is conducted to assess network reachability. No attacks are launched during this simulation. Traffic is captured and analyzed using Wireshark, and the resulting data serves as a benchmark for comparison with the traffic patterns observed when the network is subjected to an attack.

Scenario 2: 2 hosts under attack without IDS protection.

For this scenario, 2 hosts become the victims and under attack (H4 and H3). 4 hosts have been compromised by attacker and launch internet control message protocol (ICMP) flood towards the 2 hosts.

Scenario 3: 2 hosts and controller attack without IDS protection.

In this third scenario, 2 hosts and controller under attack. Switch Sw1 has been compromised and flood ICMP packet toward the SDN controller. At the same time 2 hosts (H4 and H3) also under attack from 4 compromised hosts.

Scenario 4: 2 hosts under attack with IDS protection

In the fourth scenario, IDS is enabled, 2 hosts become the victims and under attack (H4 and H3). 4 hosts have been compromised by attacker and launch ICMP flood towards the 2 hosts.

Scenario 5: 2 hosts and controller attack with IDS protection

In the fifth scenario, 2 hosts and controller under attack. Switch Sw1 has been compromised and flood ICMP packet toward the SDN controller. At the same time 2 hosts (H4 and H3) also under attack from 4 compromised hosts. However, this time snort IDS is enabled.

Table 1 summarized the 5 scenarios generated for this research. It details out what activities done for each of the scenarios, what Ryu application run during the simulation for the scenario and on which scenario snort enable. *Table 1* summarizes the details on all the 5 simulated scenarios used for testing purpose.

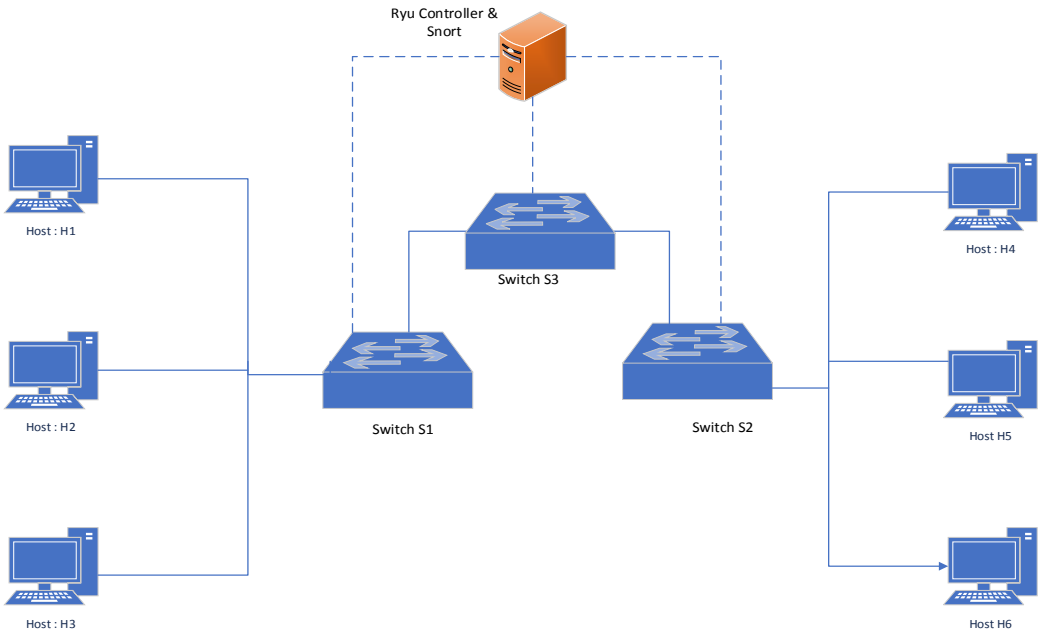


Figure 6 Network diagram used for the simulation

Table 1 Summary of the 5 scenarios

No.	Scenario name	Scenario details	Ryu controller application	DDoS attack launch	Snort
1	Scenario 1: Baseline	Sending a normal ping packet to test reachability. Capturing traffic for use as a benchmark.	DDoS_snort.py	None. Just Normal ping test	Enable
2	Scenario 2: Two (2) hosts under attack without IDS protection	Host H1, Host H2, Host H5 and Host H6 has been compromised and launch DDoS attack on Host H3 and Host H4	DDoS_snort.py	Generate Hping3 generated from Host H1, Host H2, Host H5 and Host H6 flood ICMP packet to Host 3 and Host H4	Disable
3	Scenario 3: Two (2) hosts under and controller attack without IDS protection	Host H1, Host H2, Host H5 and Host H6 has been compromised and launch DDoS attack on Host H3 and Host H4. Switch Sw1 has been compromised and launch attack on controller	simple_switch.py	Generate Hping3 generated from Host H1, Host H2, Host H5 and Host H6 flood ICMP packet to Host 3 and Host H4. Switch Sw1 flood ICMP packet toward controller	Disable

No.	Scenario name	Scenario details	Ryu controller application	DDoS attack launch	Snort
4	Scenario 4: Two (2) hosts under attack with IDS protection	Host H1, Host H2, Host H5 and Host H6 has been compromised and launch DDoS attack on Host H3 and Host H4	DDoS_snort.py	Generate Hping3 generated from Host H1, Host H2, Host H5 and Host H6 flood ICMP packet to Host 3 and Host H4	Enable
5	Scenario 5: Two (2) hosts under and controller attack with IDS protection	Host H1, Host H2, Host H5 and Host H6 has been compromised and launch DDoS attack on Host H3 and Host H4. Switch Sw1 has been compromised and launch attack on controller	DDoS_snort.py	Generate Hping3 generated from Host H1, Host H2, Host H5 and Host H6 flood ICMP packet to Host 3 and Host H4. Switch	Enable

DDoS attack performed using Hping3 packet generator. It is opens source tools for TCP/IP protocol. Throughout the simulation, the same Hping 3 argument is used. *Table 2* summarized the Hping arguments and the descriptions

Table 2 Hping3 argument summary

Hping3 Arguments	Description
--flood	Sending ICMP packet to the victims' machines as fast as possible
--ICMP	Sending ICMP packets
-d	The length of the payload to send in bytes

To enable Snort to identify potential DDoS attacks on the network, two rules are formulated within Snort. The initial rule triggers an alert labeled "ICMP DDoS attack" when Snort detects the transmission of over 100 packets per second.

First rule:

alert icmp any any -> any any (msg:"DDOD:ICMP Flooding Detected";detection_filter: track by_src,count 100, seconds 1; sid:10000002;)

The second rule generates an alert labeled "ICMP flooding detected" when Snort receives a packet with a payload exceeding 100 bytes within a span of 1 second.

Second Rule:

alert icmp any any -> any any (dsize: > 200; msg:"DDoS: Abnormal Data Packet detected"; sid:10000003;)

3.1Scenario 1: baseline network

In this scenario, a network topology is established according to *Figure 7* using Mininet. To construct the topology, execute the following command in Mininet. This identical topology persists throughout the three scenarios. Notably, no ICMP flooding is carried out in this scenario. Every 5 minutes, the command "pingall" is executed to assess the overall network connectivity. Additionally, Host H2 and Host H4 exchange pings every 5 minutes to verify the availability of the respective machines. The data collected within this scenario serves as a benchmark against which the outcomes from other scenarios are compared. The analysis is based on the total number of generated packets and the percentage of packet loss within the network.

Sudo mn – custom ~/mininet/custom/mytopo1.py –topo custom –controller=remote, ip=192.168.16.4.

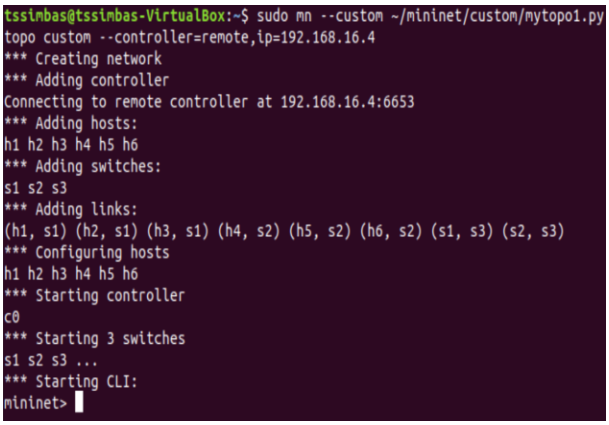


Figure 7 Creating the network topology in Mininet

The ping test indicates failure (see *Figure 8*) since the controller is not yet activated. In the absence of the controller, there is no flow information available to dictate how the packets between the hosts should be forwarded. Consequently, all hosts become

unreachable, as the switches lack the necessary data to determine how to route the traffic. Upon inspecting the dump log files, it's evident that no flows have been established within both switches. As demonstrated in *Figure 9*, neither of the OpenFlow switches possesses any logs pertaining to flow entries that could guide the forwarding of packets.

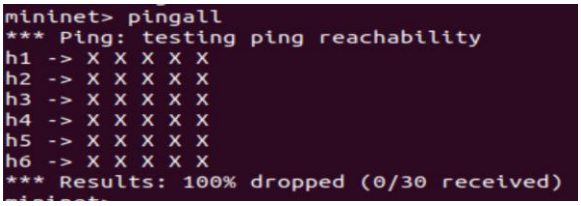


Figure 8 Ping test without enable the Ryu controller

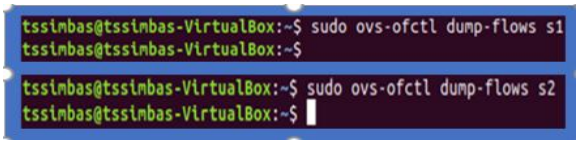


Figure 9 Dump flows logs from switch S1 and S2 without flow information

Subsequently, the Ryu controller is enabled to furnish flow information to the switches, instructing them on how to forward traffic between the hosts within the network. In this research, a Ryu controller application is employed to detect attacks using Snort. This script not only furnishes flow information to the switches but is also equipped to identify potential DDoS attack launches on the SDN network. The controller can be invoked using the following command in the terminal. This Python script integrates Ryu with Snort.

Ryu-manager –verbose app/simple_switch_snort.py

The subsequent step involves enabling the Snort tool. Snort is responsible for detecting potential DDoS attacks and subsequently sending alerts to the Ryu controller (refer to *Figure 10*). The command provided below is utilized to execute the Snort tool.

Sudo snort -c /usr/local/etc/snort/snort.lua -R /usr/local/etc/rules/local.rules \-I enp0s3 -A alert-fast -s 65535 -k none.

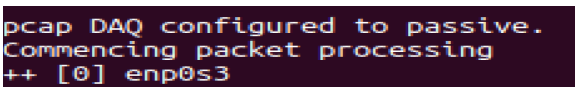


Figure 10 Snort listening on interface enp0s3

Once the Ryu manager has been activated, a normal ping test is conducted to assess network reachability. As depicted in *Figure 11*, the ping test to all nodes is successful, with no dropped packets observed, as indicated by a zero percent (0%) drop rate.

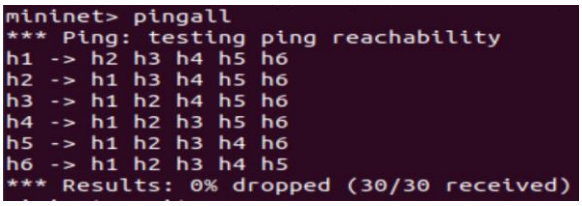


Figure 11 Ping test to check network reachability

3.2Scenario 2: 2 hosts under attack without IDS protection

The second scenario simulates an SDN under attack, albeit without IDS protection. In this scenario, four hosts (H1, H3, H5, and H6) have been compromised, granting attackers access to these hosts. Upon compromise, attackers can initiate a DDoS attack by flooding ICMP packets to the two hosts (H2 and H4). Notably, Snort is not enabled in this scenario, signifying that the SDN network lacks IDS protection against the attackers. *Figure 12* provides an illustration summarizing the particulars of this scenario. As depicted in *Figure 12*, hosts H2 and H4 are chosen as victims, operating under the assumption that these two users are high-ranking government officials whose machines store a wealth of confidential documents essential for decision-making during crucial government meetings. The remaining four victims are standard office workers within the organization, potentially clerks or receptionists, who utilize less secure machines or laptops. These four victims remain oblivious to the fact that their machines have been compromised. They continue to access their machines and carry out their daily tasks without suspicion.

3.3Scenario 3: 2 hosts and controller attack without IDS protection

In this scenario, four hosts (H1, H3, H5, and H6) have been compromised, granting attackers access to these four hosts. To exacerbate the situation, switch Sw1 has also been compromised. With this setup, attackers can launch a DDoS attack by flooding ICMP packets to the two hosts (H2 and H4). Within an SDN network, hosts communicate directly with the controller, making it necessary for the attacker to gain access to switch Sw1 in order to subsequently target the controller. Notably, in this scenario, Snort is not enabled, signifying that the SDN network lacks IDS protection against the attacker. *Figure 13* provides a summary of the particulars of this scenario.

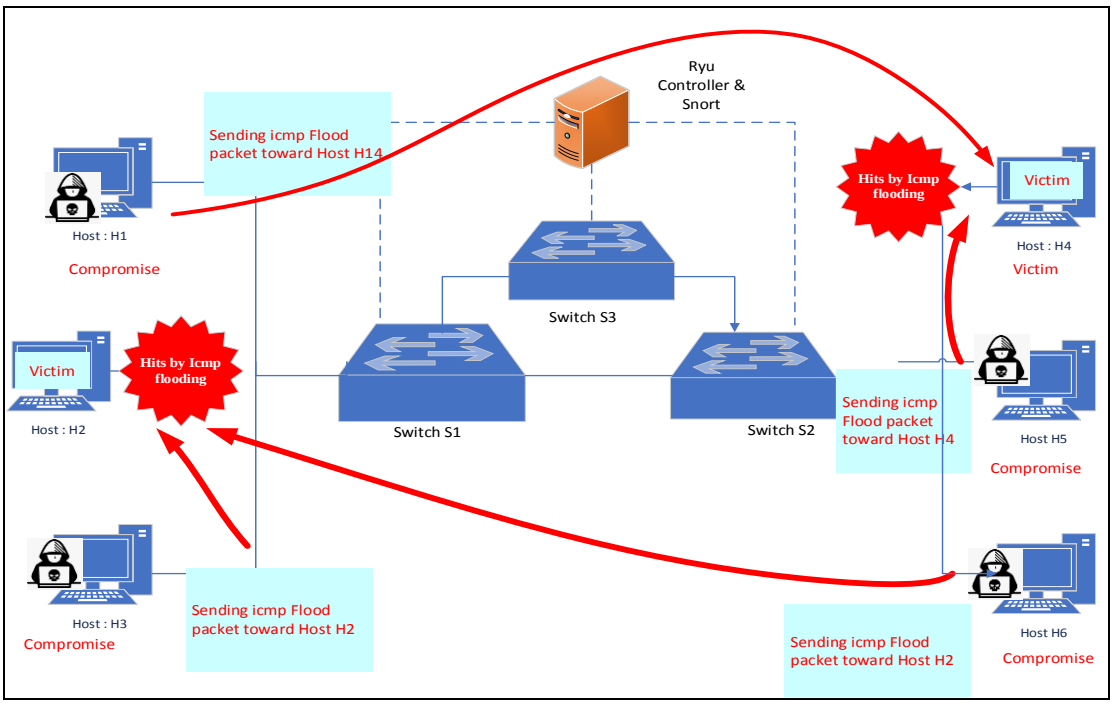


Figure 12 Scenario 2: host H1 and H4 the victim machine flood by ICMP packets

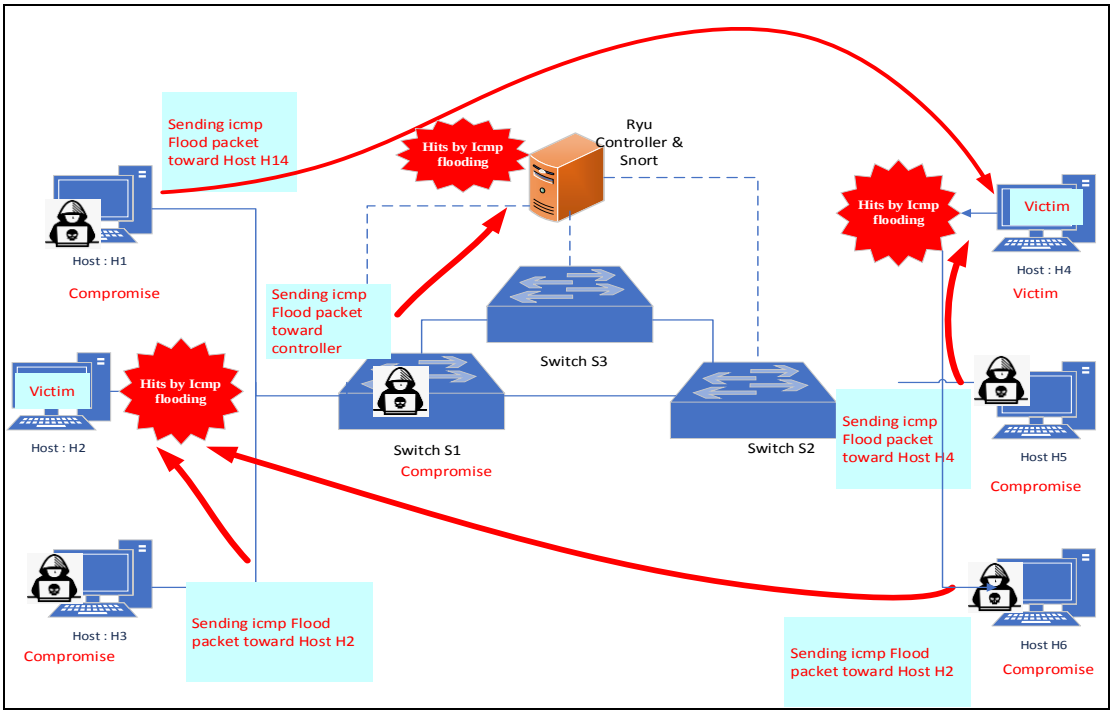


Figure 13 Scenario 3: Host H1, H2 and the controller under attack

3.4Scenario 4: 2 Hosts under attack with IDS protection

This scenario involves simulating an SDN under attack with IDS protection. In this scenario, four hosts (H1, H3, H5, and H6) have been compromised, granting attackers access to these four hosts. Upon compromise, attackers can launch a DDoS attack by flooding ICMP packets to the two hosts (H2 and H4).

Importantly, in this scenario, Snort is enabled to facilitate early detection of any incoming DDoS attack. *Figure 14* summarizes the details of this scenario.

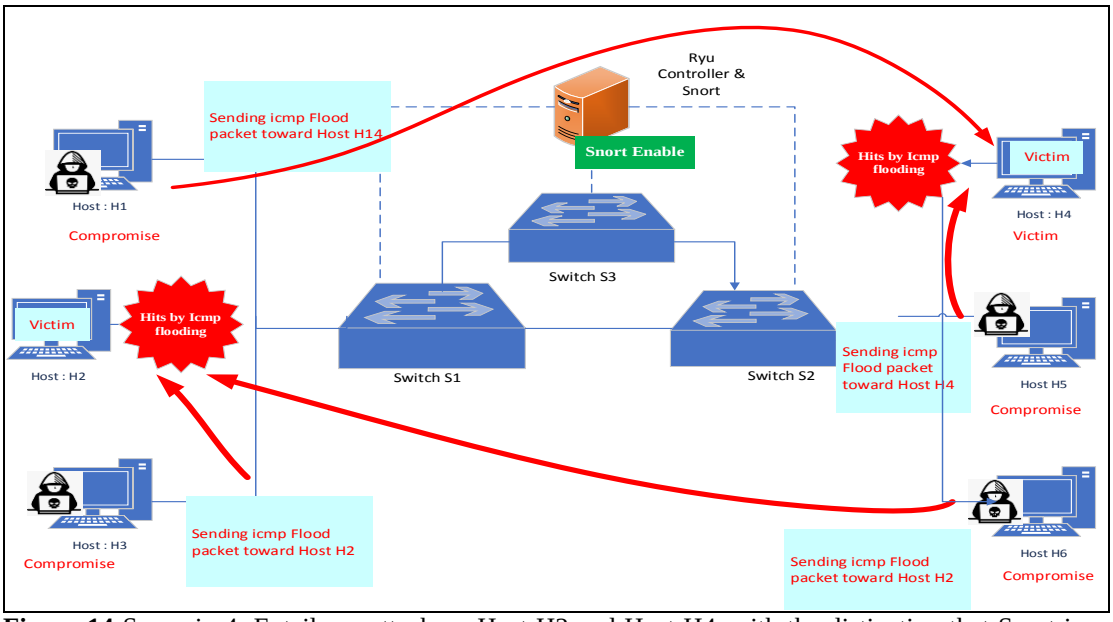


Figure 14 Scenario 4: Entails an attack on Host H2 and Host H4, with the distinction that Snort is enabled in this scenario

3.5Scenario 5: Hosts and controller under attack with IDS protection

In this scenario, all four hosts (H1, H3, H5, and H6) have fallen victim to compromise, and the attacker has also gained access to switch Sw1. With these compromised resources at hand, the attacker can now initiate a DDoS attack by flooding ICMP packets towards the two hosts (H2 and H4). Concurrently, the compromised switch Sw1 unleashes ICMP flood packets towards the controller. However, in this scenario, the Snort IDS is enabled, contributing to early detection capabilities within the SDN network. *Figure 15* offers an illustration of the circumstances in this scenario.

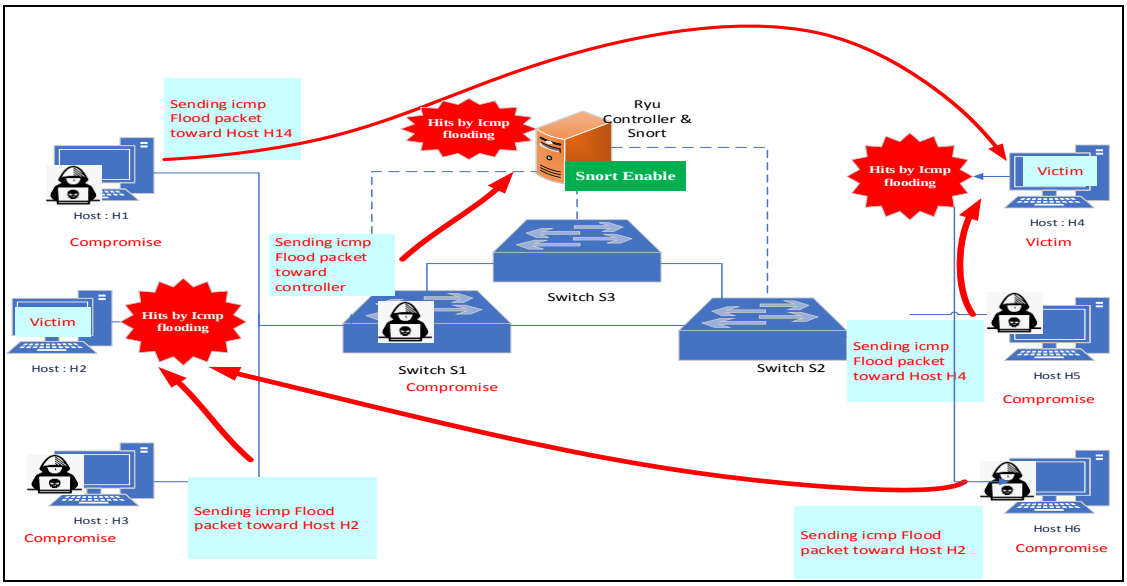


Figure 15 Scenario 5: Snort is enabled for proactive monitoring to detect the attack

4.Results

In this section, we thoroughly discuss all the results and findings gathered from the simulations. The simulations involved conducting ICMP flooding towards victims' machines using compromised sources. Scenario 2 and scenario 3 were executed without activating Snort. This mirrors a real-world situation where certain organizations do not invest in IDS/IPS for their security systems. Such organizations often rely solely on reactive monitoring tools, with the network operations center (NOC) responding to incidents only after receiving alerts.

On the other hand, scenario 4 and scenario 5 were simulated with Snort enabled. The presence of Snort as a proactive monitoring tool empowers the operations team to identify anomalies in detected traffic and initiate investigations before network resources become inaccessible.

4.1 Total number of packets

Since the DDoS attack carried by flooding the victim's machine with massive number of packets, many packets in the network indicate that there is possibility that the network is under DDoS attack. As the purpose of these packets flooding is to overwhelm the resource of the victim's machine and make it unreachable. The analysis done by comparing the total number of packets generated when the machine is under attack with the total packet generated by base network scenario.

4.2 Network availability

Network availability usually related to network uptime. The network resource should be reachable to the client when needed. In this research paper, ping test toward the victim's machine (H2 and H4) to check its availability. 100% packet lost indicates that the host is not reachable.

4.3 Network performance

Network performance pertains to the network's capability to deliver services in alignment with the anticipated and agreed-upon quality of service. Within this simulation, the evaluation of the SDN network's performance was based on packet loss. Given that DDoS attacks are designed to inundate the network with packets that exceed the available bandwidth, the network can become overwhelmed and crash. Consequently, legitimate traffic attempting to access the victim machine is denied due to the absence of network connectivity leading to the machine.

In this research, after each attack occurrence, pingall tests were performed every 5 minutes to assess network performance by quantifying the percentage of packet loss.

4.4 Simulation result

Results from each scenario were collected, documented, and analyzed. The analysis was conducted based on the total number of packets accumulated during the 30-minute interval of each simulation run. This total number of packets serves as an indicator of how many TCP packets were generated throughout the simulation.

A secondary measurement for analysis focused on the network's availability—whether it remained accessible after a certain period. Additionally, network performance was evaluated by assessing the number of packet losses incurred in each scenario.

In the case of scenario 4 and scenario 5, the focus of the results shifted towards the alert messages generated by Snort. This emphasis was placed to gauge the effectiveness of the configured rules within Snort in promptly detecting any incoming DDoS attacks within the SDN network.

4.4.1Scenario 1: Baseline scenario

The results obtained from this scenario serve as a benchmark for the simulation outcomes. This scenario replicates a scenario where all elements within the network generate a normal quantity of traffic. The results from the remaining scenarios are then compared against the findings collected from this benchmark scenario.

The simulation was conducted over a duration of 30 minutes, with pingall tests executed every 5 minutes within the Mininet environment to assess the connectivity of the nodes. The subsequent sections detail the captured and analyzed results, focusing on the total number of generated packets, the percentage of packet loss to gauge network availability and performance, as well as the efficacy of Snort's rules in detecting any suspicious incoming traffic generated by DDoS attacks.

Total number of packets

The results regarding the total number of generated packets are presented in Table 3 for both the Ryu VM and the Mininet VM. This total number of packets represents the standard quantity of received packets within the machines. The generated packets reflect a scenario in which the network is not subjected to any form of attack. The total number of packets generated

in the alternative scenarios may be contrasted with the value provided in this table. If the total number of packets surpasses the indicated figure, it indicates that the network potentially experienced an attacker's compromise.

Table 3 Total number of packet in Ryu VM and Mininet VM for Scenario 1

	RYU VM	Mininet VM
Total packets	8607	8608

Network availability

Total packet lost for the scenario gathered by checking the availability of the victims' host machine, in this case the host H2 and host H4. Host H2 and host H4 ping each other every 5 minutes. This reflects the scenario when legit user wants to initiate a valid connection. Since both host H2 and host H4 are not compromised, these 2 hosts selected for the connectivity testing. *Figure 16* shows total packet lost for both H2 and H4 remain at zero percent (0%) for the rest of the 30 minutes when both H2 and H4 ping each other. It showed network connectivity is there and the resources are available for accessibility.

Network performance

Overall network performance is assessed by employing the "pingall" command within Mininet. This command facilitates the pinging of all the host nodes established in the network. The outcome of the "pingall" test serves as an indicator of the network's overall performance. As illustrated in *Figure 17*, the results of the "pingall" test depict zero percent (0%) packet loss within scenario 1. This outcome signifies that all the hosts in the network exhibit unimpaired connectivity. The absence of packet drops or loss during the execution of the "pingall" command for every 5-minute interval indicates that all network hosts remain accessible.

Snort logs

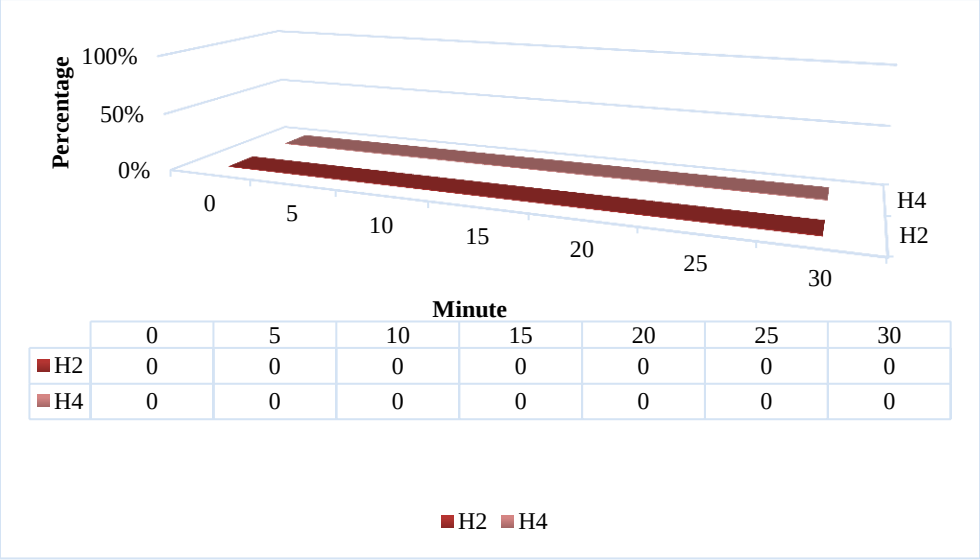
In scenario 1, snort is enabled. The captured logs are to be used as a benchmark for the other scenarios. Based on the result from *Table 4*, the log detected only 1 alert, which is not related to the 2 rules configured to detect the DDoS attack.

Table 4 Snort logs for scenario 1

	Packet received	**Packet analyze**	**Alert detected**
Scenario 1	7103	7103	1

4.2.2 Scenario 2: 2 hosts under attack without IDS protection

In this scenario, 4 hosts have been compromised and flood the victim host (H2 and H4) with traffic. Total number of packets generated in the simulation gathered and compare with the data gathered from scenario 1. Percentage of packet lost gathered, and the data is used to check on the network availability and overall network. The following section are the result captured and analyzed based on the total of number packets generated, percentage of packets lost to observe on the network availability and performance for scenario 2.



	0	5	10	15	20	25	30
H2	0	0	0	0	0	0	0
H4	0	0	0	0	0	0	0

Figure 16 Graph depicting the total packet loss for H2 and H4

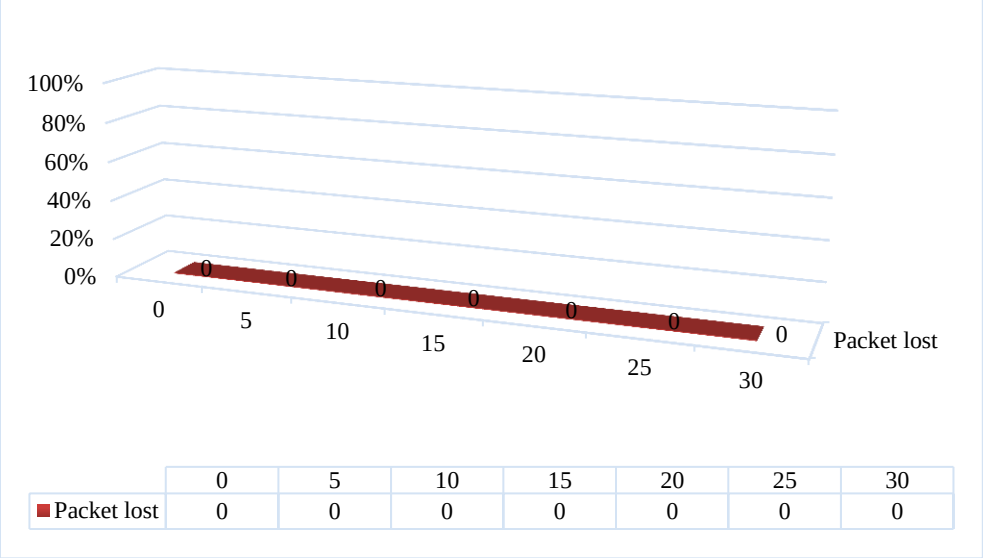


Figure 17 Total packet lost in the network for scenario 1

Total number of packets

Figure 18 represents the differences in the total of packet generated by the 2 scenarios. Comparing the differences of total packet on the Ryu machine, scenario 2 generated 1913 additional packets, meanwhile Mininet VM generated additional of 1885 packets. Huge gap in the differences of total packets generated indicates suspicious activities are currently occur in the network; which in this case attacker is flooding the victims' machine with ICMP packet to execute DDoS attack.

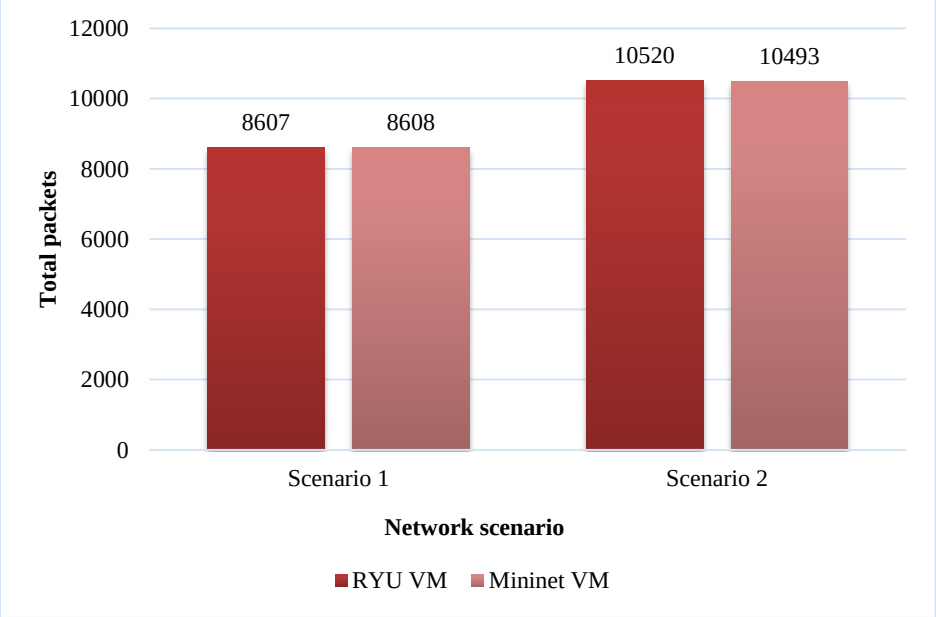


Figure 18 Comparison graph for the total number of packets generated for scenario 1 and scenario 2

Network availability

One of the objectives of the attacker launching DDoS attack is to cut accessibility to the resource, which results in the victim's machine become unavailable. When any legit connection tries to initiate connection to the machine, it becomes unreachable. *Table 5* shows the result of ping test performed to check on the availability of host H2 and host H4; which are the victims' machine in this scenario. Higher percentage

indicates the high number of packets lost from the attack.

Table 5 Packet lost in percentage for Host H2 and Host H4 when under attack

	Minutes						
	0	5	10	15	20	25	30
H2	0	93	83	83	100	91	100
H4	0	100	91	83	100	91	100

Figure 19 illustrates the comparison of packet loss results between scenario 2 and scenario 1. According to the graph, in scenario 1, both hosts (H2 and H4) do not encounter any packet loss, as indicated by a 0% packet loss rate. On the other hand, in scenario 2, both hosts experience 100% packet loss at the twenty (20) and thirty (30) minute marks of the simulation. This signifies complete inaccessibility to both hosts during the 20th and 30th minutes of the simulation, as indicated by their 100% packet loss rate.

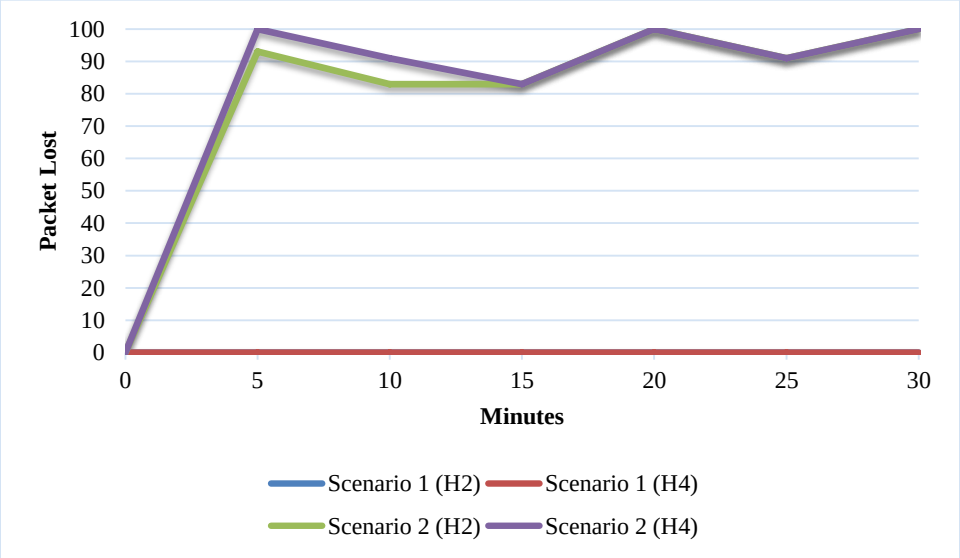


Figure 19 Comparing the result of packet lost for host H2 and H4 for scenario 1 and scenario 2

Network performance

The network performance results were collected using the pingall command, which tests the connectivity to all hosts in the network. The overall network performance was evaluated based on the total packet drop in the network. As demonstrated in *Figure 20*, the packet drop rate in scenario 1 remains consistently at 0% throughout the simulation, indicating no detected packet drops and no observable performance issues in the network.

However, in scenario 2, the results indicate a different pattern. Approximately 5 minutes after the attack is executed, a gradual increase in packet drops is observed, reaching 23%. This trend continues, with the packet drop rate rising further to 59% by the 15th minute. Higher packet drops percentages correlate with more significant network degradation.

4.4.3 Scenario 3: Two (2) hosts and controller under attack without IDS protection

In this scenario, in addition to compromising the four hosts (H1, H3, H5, and H6), the attacker successfully gains access to switch Sw1 and employs it to launch a direct DDoS attack on the controller. In SDN architecture, hosts do not directly communicate with the controller; only switches have communication with the controller. Therefore, to target the controller directly, the attacker must infiltrate any switch within the network. However, the simulation duration is not the full 30 minutes. By the 9th minute, shortly after the attack initiates, both the Ryu VM and the Mininet VM become inaccessible.

Total packets

The total number of packets collected represents the quantity of packets generated during the simulation, which occurred up to the 9th minute after the attack was initiated. Despite the simulation's short duration of less than 9 minutes, the total packet count surpasses the amount generated in both scenario 1 and scenario 2 simulations. *Figure 21* visually displays the substantial disparity in total packet count between scenario 1 and scenario 3. Similarly, when compared with the results from scenario 2, the difference is also considerable, highlighting the impact of the ICMP packet flooding on the network.

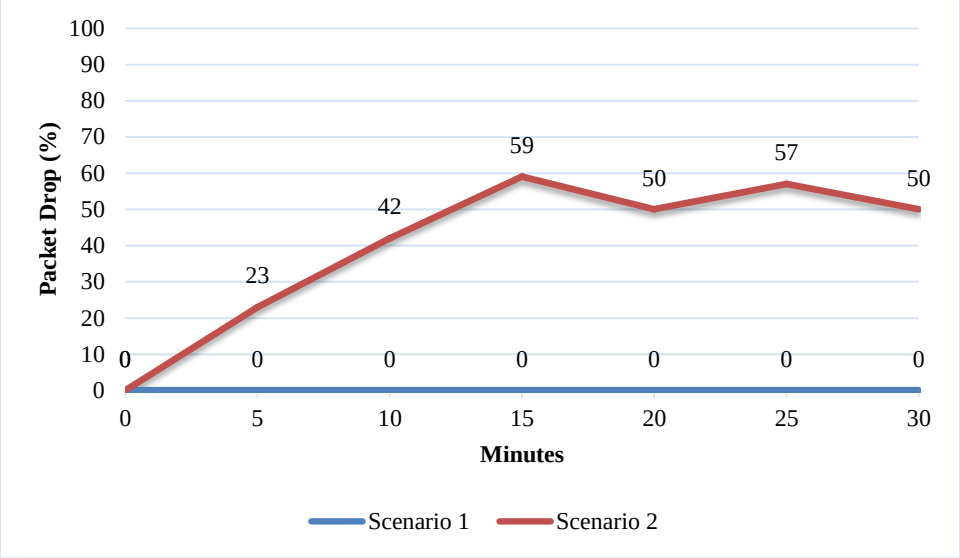


Figure 20 Comparing packet drop result between Scenario 1 and Scenario 2

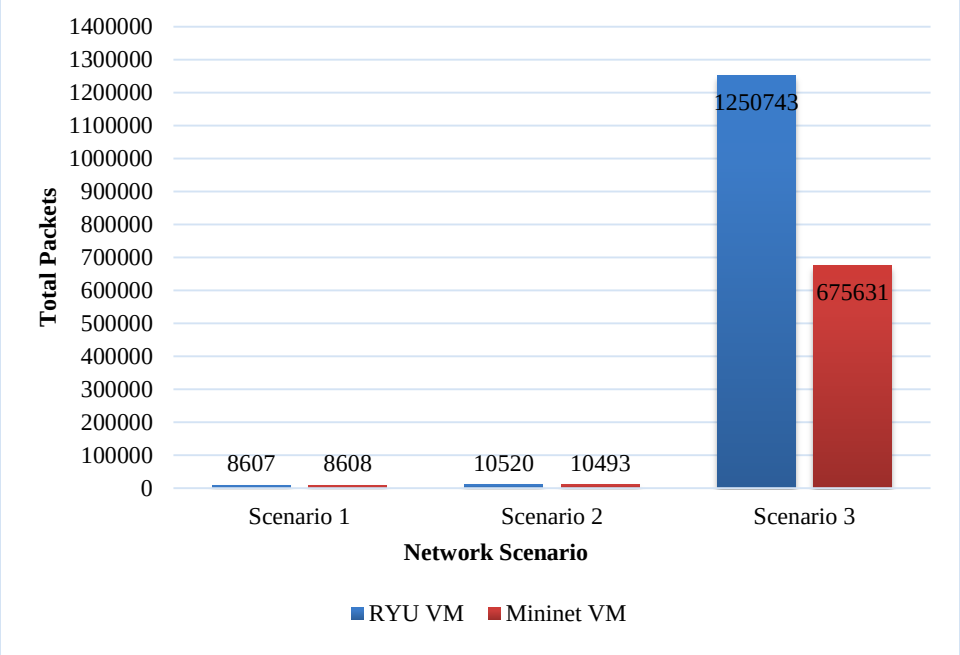


Figure 21 Comparison of total number of packets generated for the 3 scenarios

Network availability

Figure 22 depicts the unresponsiveness of both virtual machines at the 9-minute mark following the initiation of the attack. The simulation commences at 23:05 and concludes at 23:14. At this juncture, both machines are rendered inaccessible, necessitating a reboot to rectify the issue. The ping test results aimed at the targeted hosts (H2 and H4) exhibit a 100% packet loss, which initiates within 5 minutes of the simulation's commencement.

Figure 23 shows the error messages appear on both machines precisely at the 9-minute point after the attack's onset. These error messages surface when the VMs become unresponsive. This scenario mirrors real-world situations where DDoS attacks effectively deplete resources, leading to server inaccessibility.

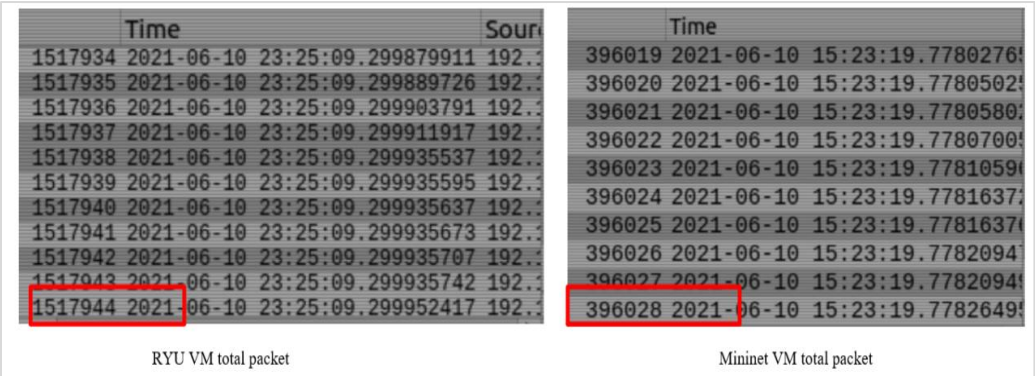


Figure 22 Both Ryu VM and Mininet VM become unresponsive

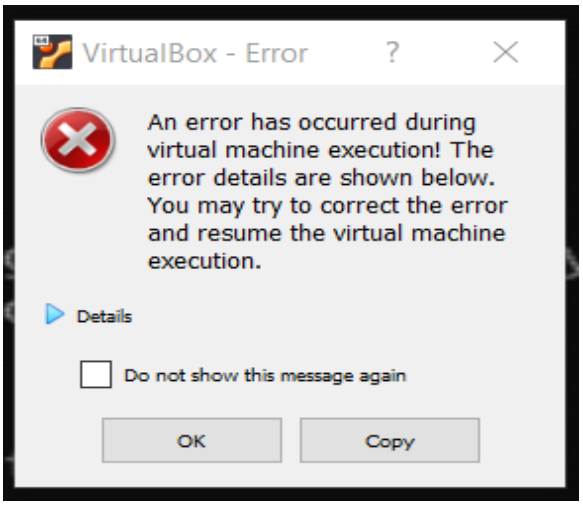


Figure 23 Error message received when both VM become unresponsive

Network performance

Both Ryu VM and Mininet VM become inaccessible once switch Sw1 starts flooding the controller with ICMP packets; results to no packets or traffic can send successfully. When both machines are inaccessible, the network is no longer usable as both the controller and the network node reside in the VM.

4.4.4 Scenario 4: 2 hosts under attack with IDS protection

This scenario is similar with scenario 2, except that the snort is enabled. In this simulation the snort act as IDS which provide proactive monitoring to the SDN network. Results gathered in this scenario focus on the snort log to observe if any alert message detected when the attack starts.

Snort logs

Snort is enabled and listened on interface enp0s3. This interface provides interconnection between Ryu VM and Mininet VM. Only traffic passing through this interface is analyzed and compared for any matches with the 2 configured rules in Snort. These 2 rules are designed to detect suspicious activities with a high likelihood of being caused by a DDoS attack. The Snort for this simulation runs for 30 minutes, and all gathered data is derived from the captured traffic during this 30-minute interval.

In *Figure 24*, only 2 alert messages logged detected from the snort. These 2 alerts do not relate to the 2 rules which configured to detect the DDoS activities in the network. The first alert generated when Mininet starts the topology and initiated the traffic for pingall test for the first time. The traffic below indicates that the OpenFlow switch communicates with controller to get the flow tables. Without the information from the flow tables, the switch does not know how to forward any traffic for the hosts connected to them. The second alert also does not match the 2 rules. Based on the message on the alert this is a normal ICMP traffic.

```
++ [0] enp0s3
[**] [122:15:1] "(port_scan) IP filtered protocol sweep" [**]
[Priority: 3]
06/12-11:46:45.372651 192.168.16.4 -> 192.168.16.5
TCP TTL:64 TOS:0x0 ID:1043 IpLen:20 DgmLen:132 DF

[**] [1:10000001:0] "ICMP Traffic Detected" [**]
[Priority: 0]
06/12-12:07:00.738467 fe80::a00:27ff:fed2:351c -> ff02::2
IPV6-ICMP TTL:255 TOS:0x0 ID:0 IpLen:40 DgmLen:56
Type:133  Code:0  UNKNOWN
```

Figure 24 Alert message detected in snort for scenario 4

Despite the flood of ICMP traffic from the 4 compromised hosts (H1, H3, H5, and H6) into the network, the Snort logs failed to detect any of these

activities. This outcome can be attributed to the OpenFlow protocol's core principle, which confines communication exclusively between OpenFlow switches and the controller. Host-to-host traffic remains internal and does not traverse the enp0s3 interface, used solely for OpenFlow switch communication. The presence of flow tables in OpenFlow switches eliminates the need for the controller to intervene for flow details.

Comparing scenario 1 and scenario 4's total received and analyzed packets, scenario 4's Snort logs reflect a higher volume of received and analyzed packets. These packets encompass diverse types, including ICMP packets and address resolution protocol (ARP) packets. *Figure 25* succinctly presents an overview of the findings from the captured data in these two scenarios. However, despite meticulous analysis, no traffic matching the criteria of the 2 configured rules designed to identify suspicious activity was found. It's worth noting that the two alerts featured in *Figure 24*, extracted from scenario 4's Snort log, do not correspond to these same 2 rules.

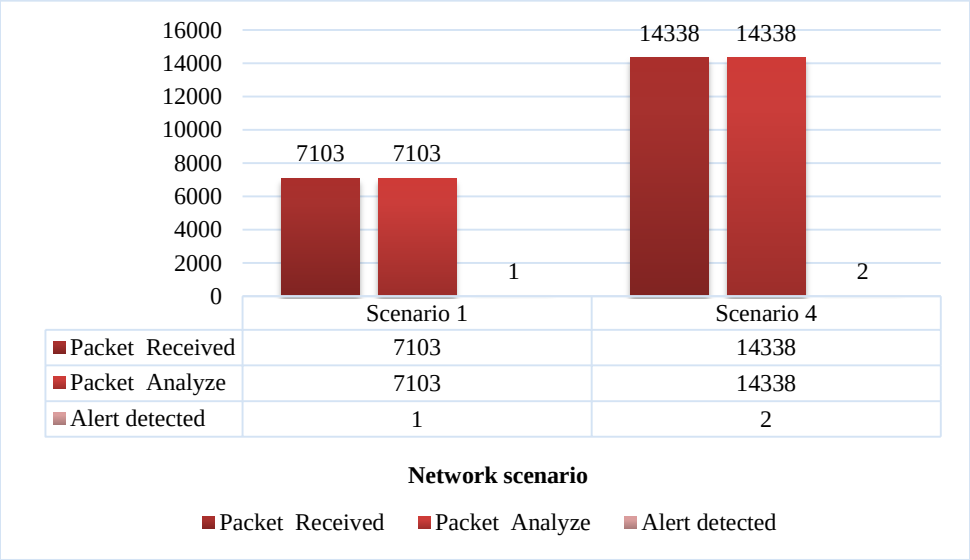


Figure 25 Comparing data from Snort logs between Scenario 1 and Scenario 4

4.4.5 Scenario 5: 2 hosts and controller under attack with IDS protection

In this scenario, the snort is enabled which also act as IDS to protect the infrastructure. This simulates the situation where an organization implements a proactive monitoring tool to protect their network infrastructure. IDS in this simulation acts as proactive monitoring tool that gather or logs any abnormal traffic detected; which is valuable for network analyst to do their troubleshooting. The result gathered in this scenario focus on the message alert received in the snort tool. This is to observe if the rules configured in the snort is capable in providing proactive monitoring by detecting incoming DDoS attack.

Snort logs

Snort, which is installed in the Ryu VM, operates by listening on the interface enp0s3, responsible for communication between the Ryu VM and the Mininet VM. Following the principles of the OpenFlow protocol employed in SDN architecture, communication is restricted to occur solely between OpenFlow switches and the controller. In this specific scenario, the compromised switch Sw1 enables the attacker to directly execute ICMP flooding toward the controller. As the attack commences from switch Sw1, Snort records and logs alert messages that align with the 2 configured rules. *Figure 26* visually represents the alert messages identified by Snort in response to the 2 established rules. The initial alert message, labeled "DDoS: Abnormal Data Packet Detected," aims to identify any packet that surpasses the data packet size set within the rules. Given that the ICMP packets flood the network with a data size of 1500, which exceeds the configured threshold of 200, this alert triggers. The subsequent alert message, labeled "DDoS: ICMP Flooding Detected," is designed to recognize incoming packets from a source generating 100 counts within a single second. Detecting this alert prompts network analysts to exercise caution. It serves as an indicator of ongoing abnormal activities.

While it could potentially be a false alert, it provides a reason for them to initiate an investigation and take appropriate actions.

```
[**] [1:10000003:0] "DDOS: Abnormal Data Packet detected" [**]
[Priority: 0]
06/12-03:39:40.360955 192.168.16.5 -> 192.168.16.4
ICMP TTL:64 TOS:0x0 ID:59 IpLen:20 DgmLen:1528
Type:8  Code:0  ID:15115   Seq:35028  ECHO

[**] [1:10000002:0] "DDOS: ICMP Flooding Detected" [**]
[Priority: 0]
06/12-03:39:40.360955 192.168.16.5 -> 192.168.16.4
ICMP TTL:64 TOS:0x0 ID:59 IpLen:20 DgmLen:1528
Type:8  Code:0  ID:15115   Seq:35028  ECHO
```

Figure 26 Alert messages in Snort match the 2 rules configured

In this scenario, the attacks transpire for a duration of 7 minutes. It is imperative that the flooding initiated by switch Sw1 ceases before reaching the 9-minute mark; failure to do so could render both VMs inaccessible, consequently preventing the collection of logs for subsequent analysis. *Figure 27* facilitates a comparison between the data extracted from Snort logs in scenario 1 and the current scenario.

From the compiled logs, scenario 1 exhibits the detection of a lone alert. However, this alert is not pertinent to the rules configured in Snort for detecting potential DDoS attacks. On the contrary, scenario 5 records a substantial 1,079,718 alerts in total, all of which align with the 2 pre-defined rules within Snort. Notably, scenario 5 also generates a significantly higher total of received and analyzed packets—specifically 743,910 and 722,925—compared to scenario 1's modest total of 7,103 packets received and analyzed.

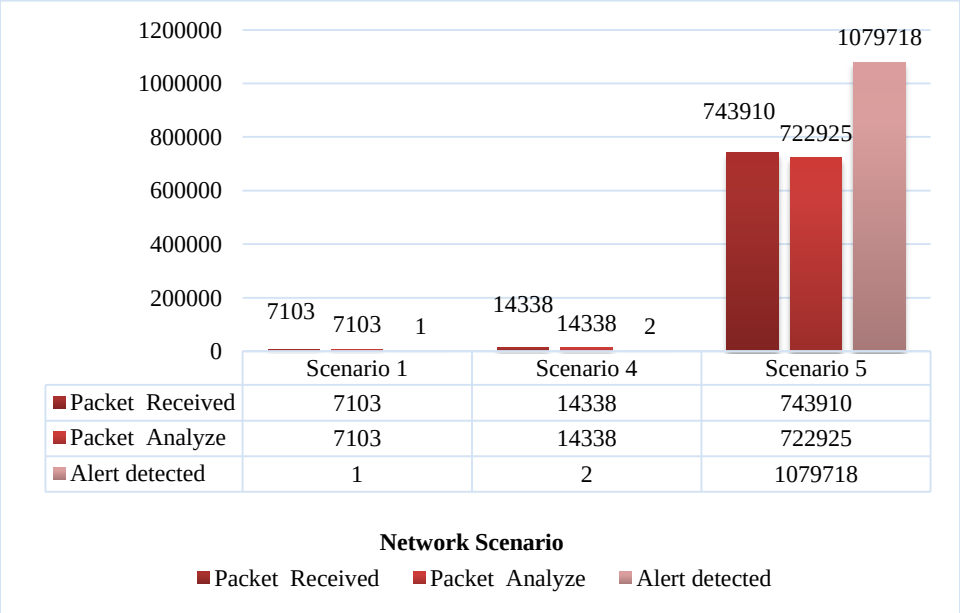


Figure 27 Comparing data from Snort logs between scenario 1 and scenario 5

5.Discussion

Among these five scenarios, the collected data is primarily concerned with the total number of generated packets for each scenario. *Figure 28* provides a summarized visualization of the total packets generated by both the Ryu VM and the Mininet VM.

Examining the outcomes presented in *Figure 28*, it's evident that the total number of packets experiences a rapid escalation in the face of attacks across the first four scenarios. Notably, both scenario 3 and scenario 5 stand out with significantly higher numbers of generated packets when compared to scenario 2 and scenario 4. This discrepancy can be attributed to the compromise of switch S1 in scenario 3 and scenario 5, which subsequently launches ICMP flooding toward the controller. Scenario 3's lifespan extends for just 9 minutes due to the exhaustion of both VM resources, rendering them inaccessible. In parallel, scenario 5 operates for 7 minutes. These findings underscore the observation that network attacks

engender heightened traffic levels. This alignment reinforces the idea that DDoS attacks aim to overwhelm networks and exhaust their resources, often resulting in inaccessibility. Notably, the evidence from scenario 3 underscores the potential for a DDoS attack to incapacitate an SDN network, leading to the unresponsiveness of both the Ryu VM and the Mininet VM.

Beyond packet count, another facet of analysis revolves around network availability, gauged through the percentage of packet loss during ping commands directed at the targeted hosts (H2 and H4). This metric reflects the unavailability of the hosts during an attack. *Figure 29* offers a concise representation of the collected packet loss percentages from the simulations. A 100% packet loss signifies complete unavailability and inaccessibility of the host. Scenario 3 depicts 100% packet loss for both H2 and H4 within the first 5 minutes of simulation initiation. As previously noted, scenario 3's simulation is curtailed to 9 minutes due to the VMs' inaccessibility. Furthermore, network performance is evaluated by measuring the percentage of lost packets—a higher percentage implying greater overall network degradation. This can be attributed to connectivity hiccups on host machines or congestion within network links. The impact of a DDoS attack is evident in the increased packet loss percentages, notably apparent when a compromised switch triggers a DDoS attack on the controller. *Figure 30*, depicting scenario 3 and scenario 5, illustrates the eventual inaccessibility of the machines after approximately 10 minutes. In the interest of data collection, scenario 5's simulation was halted at 7 minutes to facilitate log collection in Snort. Otherwise, both VMs would encounter a fate similar to scenario 3, denoted by the 100% packet loss rate. This underscores the effectiveness of disabling an SDN network through a direct attack on the controller.

Continuing with the analysis, the capability of Snort to detect DDoS attacks targeting the SDN network is explored. For data collection, only three scenarios are activated with Snort. Scenario 1 serves as the baseline, scenario 4 simulates an attack on only two host machines, while scenario 5 replicates a direct attack on the controller. *Figure 30* illustrates the activation of Snort, enabling it to monitor traffic traversing the enp0s3 interface, facilitating communication between the Ryu controller and the OpenFlow switch within the Mininet VM. Any traffic is scrutinized against the parameters of the two pre-configured rules. These rules are engineered to identify traffic patterns characteristic of DDoS attacks.

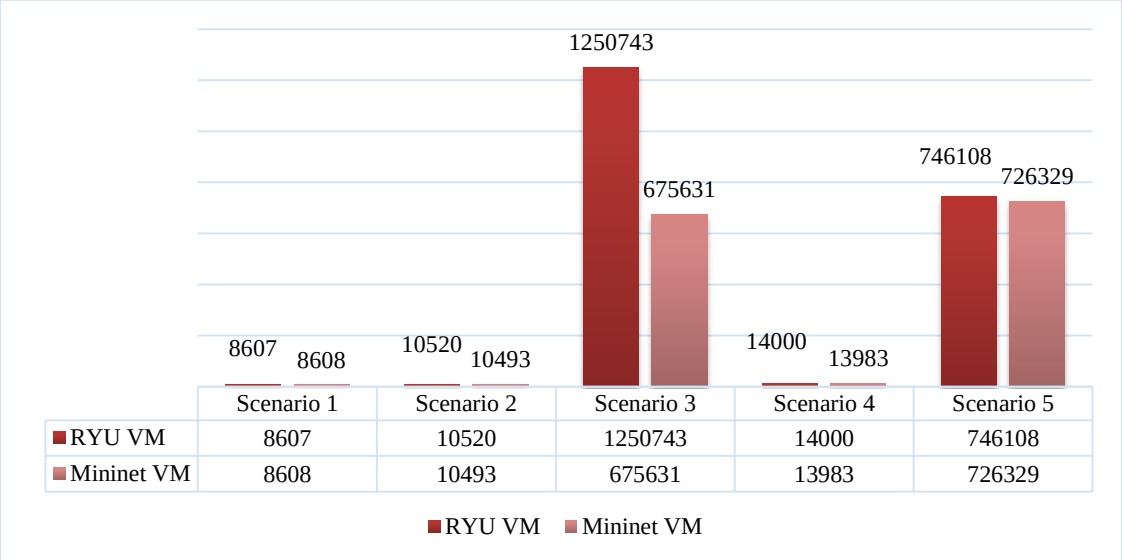
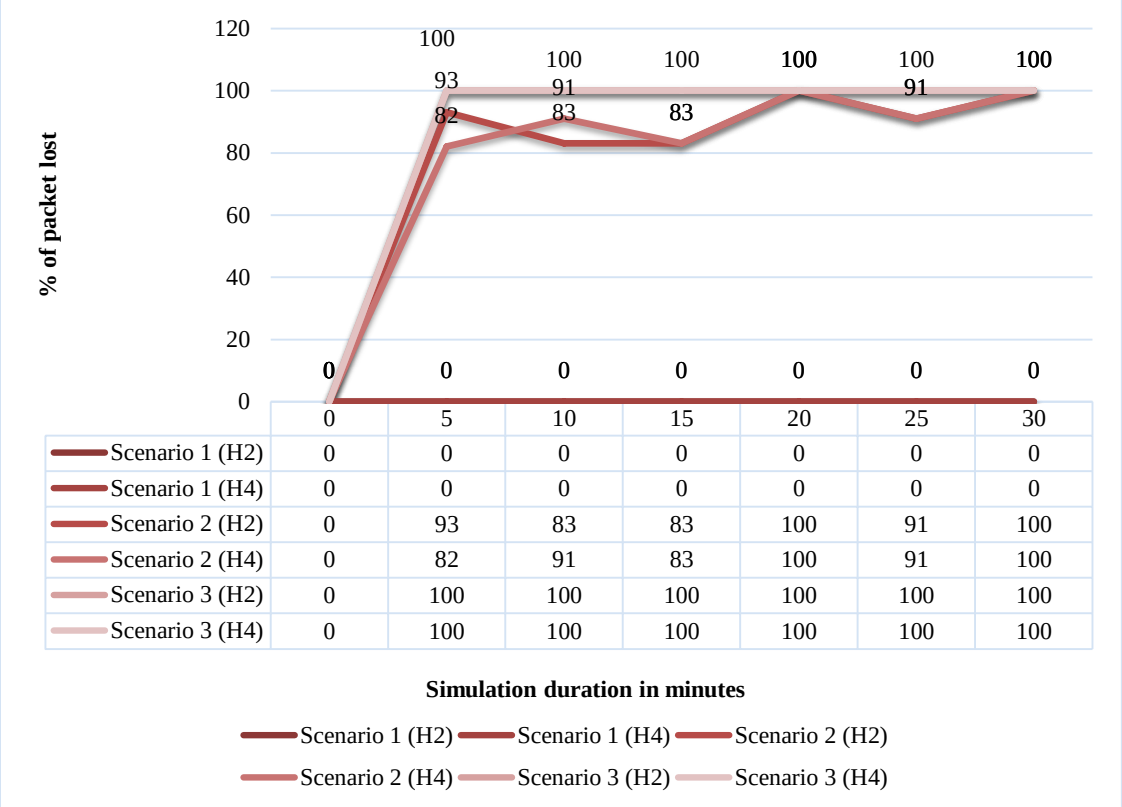
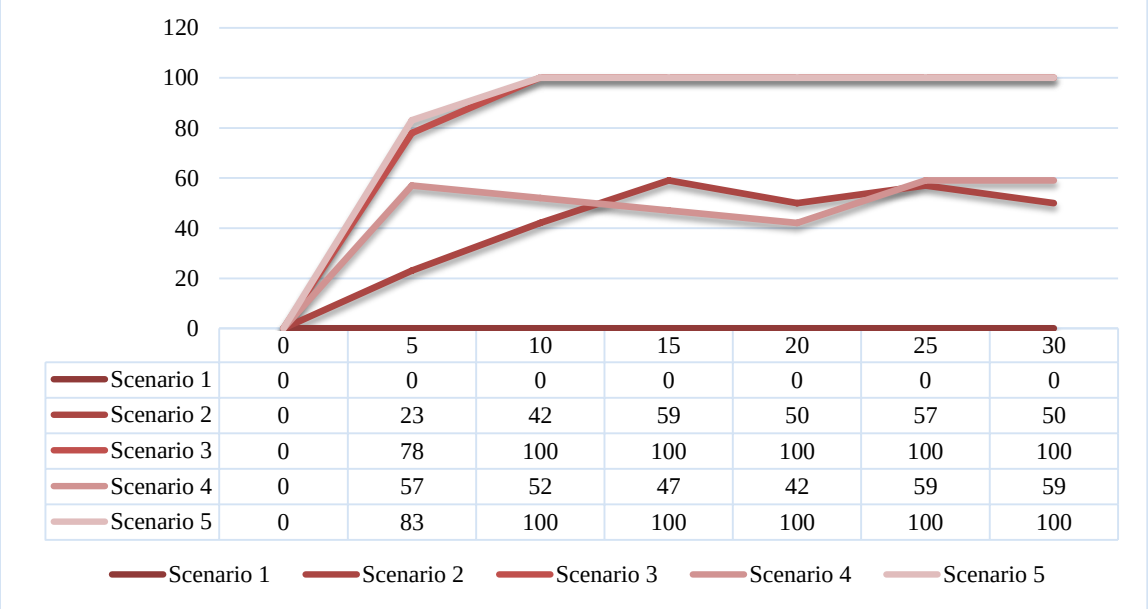


Figure 28 Summarizing total packets generated for all the 5 scenarios



	0	5	10	15	20	25	30
Scenario 1 (H2)	0	0	0	0	0	0	0
Scenario 1 (H4)	0	0	0	0	0	0	0
Scenario 2 (H2)	0	93	83	83	100	91	100
Scenario 2 (H4)	0	82	91	83	100	91	100
Scenario 3 (H2)	0	100	100	100	100	100	100
Scenario 3 (H4)	0	100	100	100	100	100	100

Figure 29 Percentage of packets lost for Host H2 and H4 for 3 scenarios



	0	5	10	15	20	25	30
Scenario 1	0	0	0	0	0	0	0
Scenario 2	0	23	42	59	50	57	50
Scenario 3	0	78	100	100	100	100	100
Scenario 4	0	57	52	47	42	59	59
Scenario 5	0	83	100	100	100	100	100

Figure 30 Percentage of packets lost in the network for all the 5 scenarios

Referring to the observations in *Figure 31*, it's evident that scenario 1 registers a solitary alert message, which, importantly, remains unrelated to any DDoS attack. Even in the context of scenario 4 where the network is under attack, Snort remains unable to identify any traffic indicative of a DDoS attack. These two pre-configured rules are meticulously designed to capture traffic patterns

inherent in DDoS attacks. Yet, the Snort logs in scenario 4 reveal two alerts that hold no connection to these rules designed for identifying DDoS-induced traffic.

This disparity can be attributed to the nature of ICMP flooding in scenario 4, which operates on a host-to-host basis. The flooding does not traverse the enp0s3 interface, responsible for linking the OpenFlow switch to the Ryu controller. In fact, the OpenFlow switch solely interacts with Ryu's controller when requesting missing entries within its flow table. Consequently, in scenario 4, the ICMP flooding circumvents the controller and thus avoids the enp0s3 interface. As a result, Snort remains oblivious to this type of traffic.

Contrasting this, scenario 5 presents a contrasting picture with a substantial volume of alerts detected by Snort. These alerts align seamlessly with the two established rules, which are meticulously crafted to detect incoming DDoS attacks. In this case, the ICMP flooding stemming from the compromised S1 switch directly targets the controller, passing through the communication channel facilitated by the enp0s3 interface. This distinction explains the variance in Snort's performance across the different scenarios.

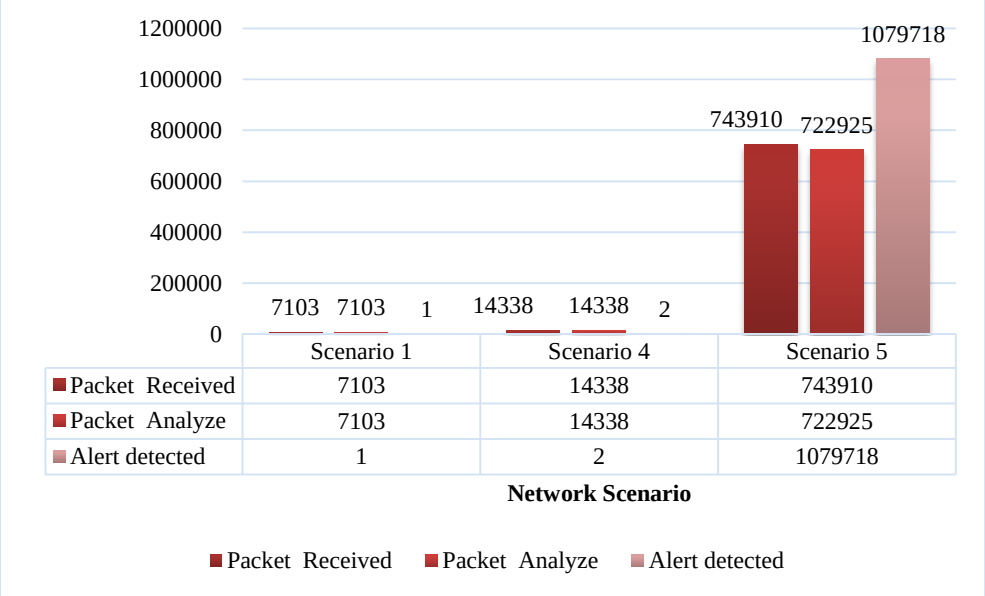


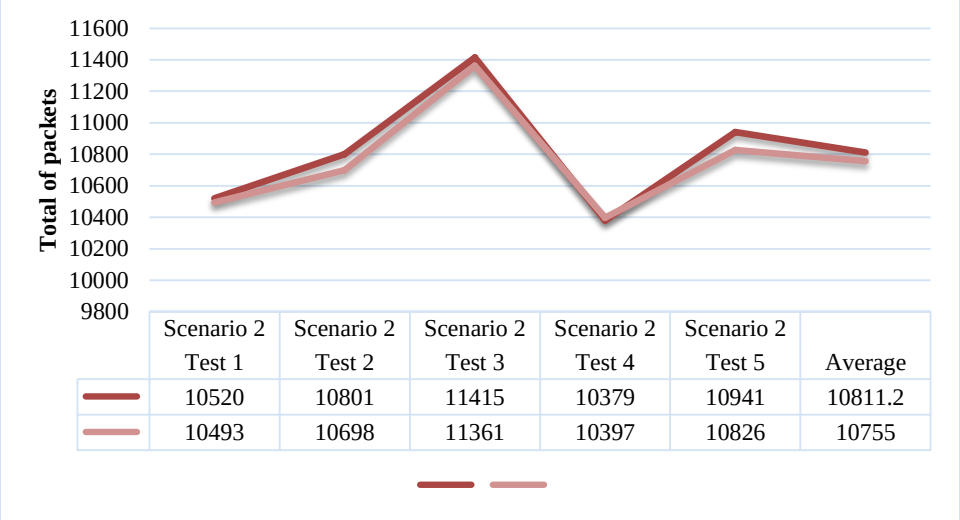
Figure 31 Logs captured from snort for the 3 scenarios

Based on these findings, the incorporation of Snort into the network offers a twofold advantage—it serves as an initial detection mechanism and a proactive monitoring tool for identifying incoming DDoS attacks aimed at the SDN network. This mirrors real-world scenarios where alerts received by Snort proactively warn network administrators about suspicious activities within the network. This early notification provides administrators with a crucial window of time to respond before the network succumbs. This contrasts the situation in scenario 3, where VM 2 lacks Snort, leading to a lack of early warning. By the 9-minute mark, both VMs become unresponsive, resulting in data loss. In stark comparison, scenario 5, which features Snort alerts, grants researchers the opportunity to halt an attack manually before VM inaccessibility sets in. Enabling Snort within the network thus furnishes a proactive monitoring tool that enables early detection and timely response to suspicious activities, empowering network administrators to act.

Further observations were made concerning the pattern of generated packets and packet loss during DDoS attacks. In scenario 2, the simulation is executed five times to assess output variability. The utilization of the hping3 command with the --flood option simulates a DDoS attack by sending packets at maximum speed. Consequently, the total number of packets generated within the network fluctuates, even when the simulation duration remains constant at 30 minutes. *Figure 32* illustrates the output obtained from the five tests conducted in scenario 2. This graph demonstrates the non-consistency of total packet counts across the five tests. Each test, whether on the Mininet VM or Ryu VM, yields a distinct

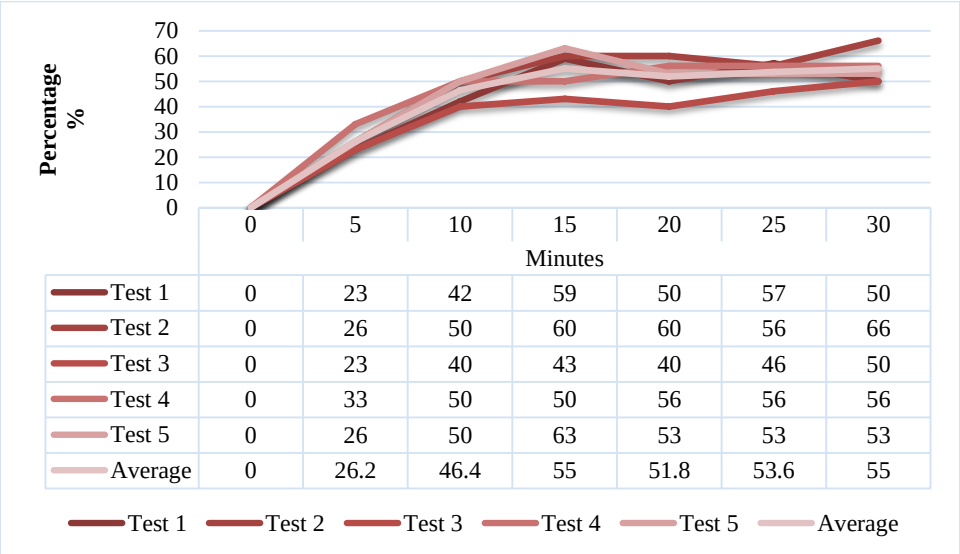
value. For instance, the first test yields 10,493 packets generated on the Mininet VM, while the second test generates 10,698. Likewise, the Ryu VM records 10,520 packets in the first test and 10,801 in the second. This variability is due to the nature of the flood option in the hping3 command, which does not specify a fixed number of packets to be sent. Instead, it floods packets at maximum speed throughout the simulation.

The inherent randomness in the total packet generation caused by the flood option also contributes to fluctuations in the total number of lost packets. Lost packets are observed during network connectivity checks to gauge overall network performance and availability during a DDoS attack. *Figure 33* demonstrates the varying percentage of lost packets across tests when the pingall command is employed to assess network performance under a DDoS attack. The graph illustrates fluctuations in percentage values among the tests, without a discernible pattern due to the random nature of traffic flooding. By comparing the outputs with the average, the results exhibit minimal differences. For instance, in the first test, cumulative packet loss at 5 minutes is 23%, while the average output stands at 26.2%. The discrepancy between these two values amounts to 3.2%.



	Scenario 2 Test 1	Scenario 2 Test 2	Scenario 2 Test 3	Scenario 2 Test 4	Scenario 2 Test 5	Average
	10520	10801	11415	10379	10941	10811.2
	10493	10698	11361	10397	10826	10755

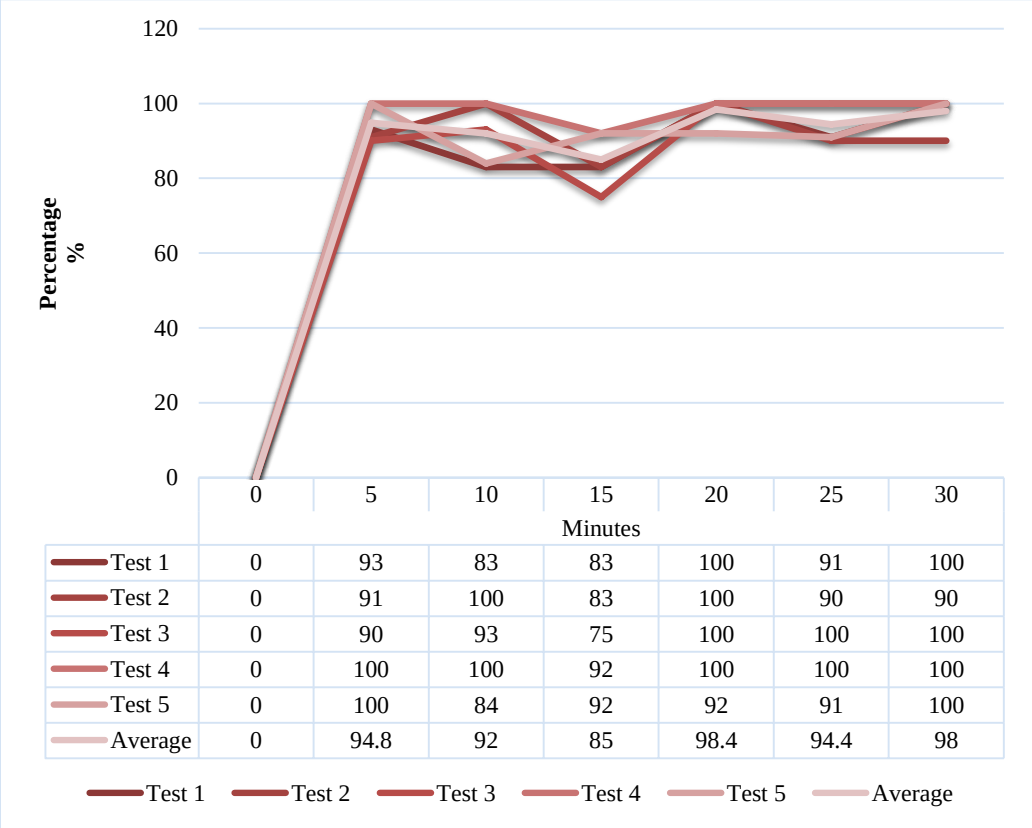
Figure 32 Total number of packets for the 5 tests run under scenario 2



	0	5	10	15	20	25	30
Test 1	0	23	42	59	50	57	50
Test 2	0	26	50	60	60	56	66
Test 3	0	23	40	43	40	46	50
Test 4	0	33	50	50	56	56	56
Test 5	0	26	50	63	53	53	53
Average	0	26.2	46.4	55	51.8	53.6	55

Figure 33 Percentage of packets lost in the network for the 5 tests

The connection test results for both H2 and H4 hosts also illustrate the output of the percentage packet lost when the ping tests performed on these hosts at a difference of every 5 minutes between each other. For example, in *Figure 34* in test 1, the result from the first 5 minutes was 93% and in test 2 the result yield was 91%. When comparing this value with the average output of 94.8%, the difference in scores is insignificant. Likewise for the result taken for connectivity test towards H4, which is depicted in *Figure 35*. For example, for test 1, the result for the first 5 minutes is 100% and the average output for the test at 5 minutes is 93.6%. The difference in scores is 6.4%, which is still considered low.



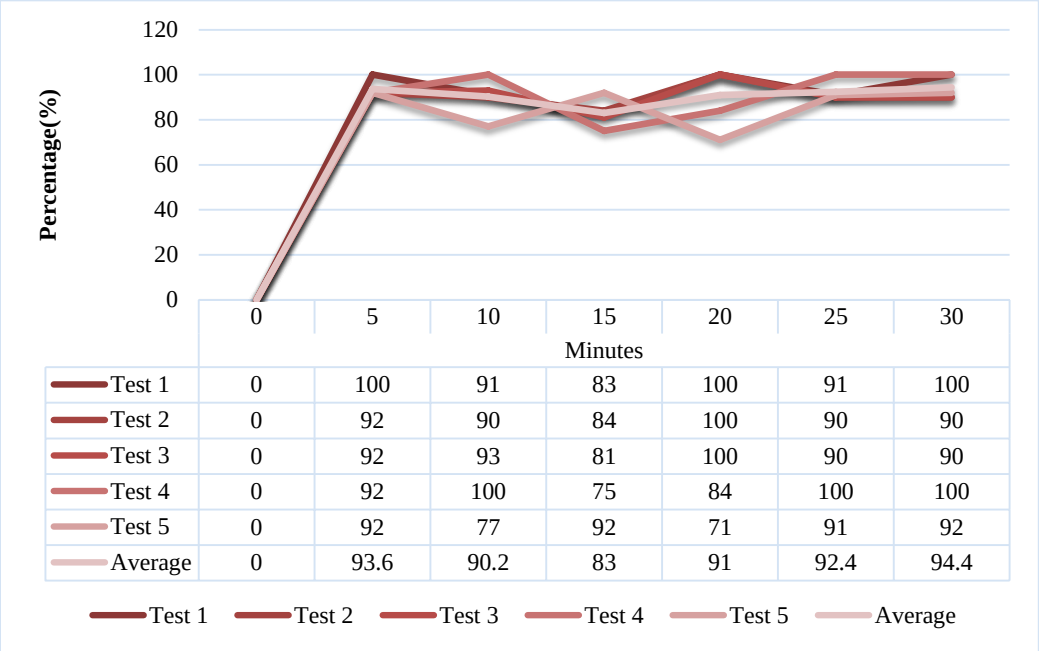
	0	5	10	15	20	25	30
Test 1	0	93	83	83	100	91	100
Test 2	0	91	100	83	100	90	90
Test 3	0	90	93	75	100	100	100
Test 4	0	100	100	92	100	100	100
Test 5	0	100	84	92	92	91	100
Average	0	94.8	92	85	98.4	94.4	98

Figure 34 Percentage of packets lost for connectivity check toward host H2

The utilization of the flood option within the hping3 command introduces a level of uncertainty in the magnitude of flood traffic sent to the network. This characteristic stems from the flood option's behavior of dispatching packets as rapidly as possible without specifying an exact quantity. This intrinsic feature of DDoS attacks further underscores their threat, as the variability in traffic volume makes it challenging to precisely define a threshold for detecting an incoming DDoS attack.

As previously highlighted, the gathered results and subsequent analysis revolve around three key variables. The initial variable pertains to the total number of packets generated across the five scenarios. The second variable gauges the availability of the victim host network during an attack, while the third variable assesses the overall network performance. Alongside these parameters, the efficacy of Snort as an IDS for early detection is also evaluated. The analysis unveils that DDoS attacks result in an upsurge of traffic within the network, substantiating the observation that ICMP flooding heightens network traffic levels. Consequently, the targeted machines (H2 and H4) become inaccessible to other users attempting to establish connections. Furthermore, this influx of traffic negatively impacts the overall network performance, evident through an escalation in packet loss. Nonetheless, this does not render SDN immune to security breaches, as evidenced by its susceptibility to DDoS attacks. This susceptibility becomes apparent through the escalation of packet loss within the network. Additionally, the presence of Snort reinforces

network accessibility through continuous detection and response to incoming attacks. While SDN offers advantages in terms of network management, its vulnerability to certain types of security attacks, such as DDoS, underscores the importance of robust security measures and proactive monitoring tools. A complete list of abbreviations is shown in *Appendix I.*



	0	5	10	15	20	25	30
Test 1	0	100	91	83	100	91	100
Test 2	0	92	90	84	100	90	90
Test 3	0	92	93	81	100	90	90
Test 4	0	92	100	75	84	100	100
Test 5	0	92	77	92	71	91	92
Average	0	93.6	90.2	83	91	92.4	94.4

Figure 35 Percentage of packets lost for connectivity check toward host H4

6.Conclusion

SDN has emerged as the preferred solution for advancing technologies like the Internet of Things (IoT), cloud computing, and data center networks. It effectively addresses limitations found in traditional networks. By decoupling the control plane from the data plane, SDN simplifies switch functionality to packet forwarding, as demonstrated in the simulations within this research paper. While the results underscore Snort's prompt detection of DDoS attacks, there remains scope for enhancing SDN network protection.

Decoupling the network shifts decision-making and computational tasks to a central controller, presenting a potential vulnerability—a single point of failure in the form of the controller. A successful DDoS attack on this controller could incapacitate the entire SDN network. To counter this, an optimal setup should introduce a secondary controller for redundancy. This secondary controller would take over when the primary one is detected as down. Moreover, it is prudent to allocate ample capacity to the controller, ensuring it can manage the high influx of traffic from

OpenFlow switches. While enhancing Snort's

effectiveness in safeguarding an SDN network, coupling it with a firewall is recommended. Incoming alerts regarding potential attacks should guide the firewall's actions, enabling it to block suspicious traffic. In the current simulation, the attack was manually halted by stopping the hping3 process. An alternative approach could involve using a physical setup instead of a simulation. This shift is advocated due to the simulation's efficacy being influenced by the host machine's capacity.

Acknowledgment

The authors express their gratitude and acknowledge the support provided by the Crowd Business and Innovation Research Interest Group, School of Computing Sciences, College of Computing, Informatics and Mathematics, UiTM Shah Alam.

Conflicts of interest

The authors have no conflicts of interest to declare.

Author's contribution statements

Nor Shahniza Kamal Bashah: Conceptualization, investigation, data curation, supervision, writing – review and editing. **Twiene Selynda Simbas:** Data collection, conceptualization, writing – original draft, analysis, and interpretation of results. **Norjansalika Janom:** Study

conception, design, investigation on challenges and draft manuscript preparation. **Syaripah Ruzaini Syed Aris:** Study conception, design, investigation on challenges and draft manuscript finalization.

References

[1] Zhang C, editor. Human security in China: A post-pandemic state. Springer Nature; 2021.

[2] Stanford B, Foster S, Berdud CE. Global pandemic, security and human rights: comparative explorations of COVID-19 and the law. Routledge; 2021.

[3] Hu M. Pandemic surveillance: privacy, security, and data ethics. Edward Elgar Publishing; 2022.

[4] Shaw R, Gurtoo A. Introduction: global pandemic, human security, technology and development. In global pandemic and human security: technology and development perspective 2022 (pp. 1-14). Singapore: Springer Nature Singapore.

[5] Gunaratna RK, Aslam MM. COVID-19 Pandemic: the threat and response. Routledge; 2022.

[6] Kumar S, Gaur MS, Sharma PS, Sagar V. Post pandemic cyber attacks impacts and countermeasures: a systematic review. In international conference on artificial intelligence and smart communication 2023 (pp. 192-9). IEEE.

[7] Bohara B, Bhuyan J, Wu F, Ding J. A survey on the use of data clustering for intrusion detection system in cybersecurity. International Journal of Network Security & its Applications. 2020; 12(1):1-18.

[8] Celesova B, Val'ko J, Grezo R, Helebrandt P. Enhancing security of SDN focusing on control plane and data plane. In 7th international symposium on digital forensics and security 2019 (pp. 1-6). IEEE.

[9] Zhang H, Cai Z, Liu Q, Xiao Q, Li Y, Cheang CF. A survey on security-aware measurement in SDN. Security and Communication Networks. 2018; 2018:1-15.

[10] Khalifa R, El-aasser M. Network security challenges in SDN environments. In 5th international conference on communications, signal processing, and their applications 2022 (pp. 1-6). IEEE.

[11] Ahmed SB, Mohamed YA. An approach for software-defined networks security. In second international conference on electrical, electronics, information and communication technologies 2023 (pp. 1-8). IEEE.

[12] Wang L, Qin Y, Li N. Research on security protection system under multi-party gathering technology of computer big data. In 3rd international conference on electronic technology, communication and information 2023 (pp. 1286-9). IEEE.

[13] https://asset.mkn.gov.my/wp-content/uploads/2020/10/MalaysiaCyberSecurityStrategy2020-2024.pdf. Accessed 27 July 2023.

[14] https://enterprise.verizon.com/content/verizonenterprise/us/en/index/resources/reports/2020-data-breach-investigations-report.pdf. Accessed 27 July 2023.

[15] https://k12cybersecure.com/wp-content/uploads/2021/03/StateofK12Cybersecurity-2020.pdf. Accessed 27 July 2023.

[16] Khairi MH, Ariffin SH, Latiff NM, Abdullah AS, Hassan MK. A review of anomaly detection techniques and distributed denial of service (DDoS) on software defined network (SDN). Engineering, Technology & Applied Science Research. 2018; 8(2):2724-30.

[17] Manso P, Moura J, Serrão C. SDN-based intrusion detection system for early detection and mitigation of DDoS attacks. Information. 2019; 10(3):1-17.

[18] Iqbal M, Iqbal F, Mohsin F, Rizwan M, Ahmad F. Security issues in software defined networking (SDN): risks, challenges and potential solutions. International Journal of Advanced Computer Science and Applications. 2019; 10(10):298-303.

[19] Clouder A. DDoS attack statistics and trend report by alibaba cloud. https://www.alibabacloud.com/blog/ddos-attack-statistics-and-trend-report-by-alibaba-cloud_597607. Accessed 27 July 2023.

[20] Rawal BS, Patel S, Sathiyanarayanan M. Identifying ddos attack using split-machine learning system in 5g and beyond networks. In INFOCOM conference on computer communications workshops 2022 (pp. 1-6). IEEE.

[21] Cai T, Jia T, Adepu S, Li Y, Yang Z. ADAM: an adaptive DDoS attack mitigation scheme in software-defined cyber-physical system. IEEE Transactions on Industrial Informatics. 2023.

[22] Niu M, Feng Y, Sakurai K. A two-stage detection system of DDoS attacks in SDN using a trigger with multiple features and self-adaptive thresholds. In 17th international conference on ubiquitous information management and communication 2023 (pp. 1-8). IEEE.

[23] Sai AD, Tilak BH, Sanjith NS, Suhas P, Sanjeetha R. Detection and mitigation of low and slow DDoS attack in an SDN environment. In international conference on distributed computing, VLSI, electrical circuits and robotics 2022 (pp. 106-11). IEEE.

[24] Yadav AR, Jain AP, Shankar T, Rajesh A, Perumal S, Eappen G. AI based DDOS attack detection of SDN network in mininet emulator. In 2nd international conference on vision towards emerging trends in communication and networking technologies 2023 (pp. 1-4). IEEE.

[25] Dou S, Miao G, Guo Z, Yao C, Wu W, Xia Y. Matchmaker: maintaining network programmability for software-defined WANs under multiple controller failures. Computer Networks. 2021; 192:108045.

[26] Mahmood W, Nasir SD, Waqas I. A research survey on software defined networking (SDN). In proceedings ninth international conference on advances in computing, control and networking 2019 (pp. 1-6).

[27] Sharma PK, Tyagi SS. Improving security through software defined networking (SDN): an SDN based model. International Journal of Recent Technology and Engineering. 2019; 8:295-300.

[28] Sunday UI, Akhibi SD. Application of software-defined networking. European Journal of Computer

Science and Information Technology. 2022; 10(2):27-48.
[29] Andishmand R, Mohammdi H, Mostafavi S. Detection and analysis of DDoS attacks in software-defined networks. Computer Security and Reliability. 2020:1-14.
[30] Bangui H, Ge M, Buhnova B. A hybrid data-driven model for intrusion detection in VANET. Procedia Computer Science. 2021; 184:516-23.
[31] Sukumar JA, Pranav I, Neetish MM, Narayanan J. Network intrusion detection using improved genetic k-means algorithm. In international conference on advances in computing, communications and informatics 2018 (pp. 2441-6). IEEE.
[32] Bhattacharjee PS, Fujail AK, Begum SA. A comparison of intrusion detection by K-means and fuzzy C-means clustering algorithm over the NSL-KDD dataset. In international conference on computational intelligence and computing research 2017 (pp. 1-6). IEEE.
[33] Karataş F, Korkmaz SA. Big data: controlling fraud by using machine learning libraries on spark. International Journal of Applied Mathematics Electronics and Computers. 2018; 6(1):1-5.
[34] Krishna KV, Swathi K, Rao BB. A novel framework for NIDS through fast KNN classifier on CICIDS 2017 dataset. International Journal of Recent Technology and Engineering. 2020; 8(5):3669-75.
[35] Alrowaily M, Alenezi F, Lu Z. Effectiveness of machine learning based intrusion detection systems. In security, privacy, and anonymity in computation, communication, and storage: 12th international conference, SpaCCS 2019, Atlanta, GA, USA, 2019 (pp. 277-88). Springer International Publishing.
[36] Verma A, Ranga V. Statistical analysis of CIDDS-001 dataset for network intrusion detection systems using distance-based machine learning. Procedia Computer Science. 2018; 125:709-16.
[37] Li L, Zhang H, Peng H, Yang Y. Nearest neighbors based density peaks approach to intrusion detection. Chaos, Solitons & Fractals. 2018; 110:33-40.
[38] Sandosh S, Govindasamy V, Akila G. Enhanced intrusion detection system via agent clustering and classification based on outlier detection. Peer-to-Peer Networking and Applications. 2020; 13:1038-45.
[39] Aung YY, Min MM. Hybrid intrusion detection system using K-means and K-nearest neighbors algorithms. In IEEE/ACIS 17th international conference on computer and information science 2018 (pp. 34-8). IEEE.
[40] Al Salti I, Zhang N. LINK-GUARD: an effective and scalable security framework for link discovery in SDN networks. IEEE Access. 2022; 10:130233-52.
[41] Agborubere B, Sanchez-velazquez E. Openflow communications and TLS security in software-defined networks. In international conference on internet of things (iThings) and IEEE green computing and communications (GreenCom) and IEEE cyber, physical and social computing (CPSCom) and IEEE smart data (SmartData) 2017 (pp. 560-6). IEEE.
[42] Muragaa WH, Seman K, Marhusin MF. Simulating DDoS attack in SDN network using POX controller and Mininet emulator. In proceedings of 134th the IRES international conference. 2018 (pp. 39-41).
[43] Tupakula U, Karmakar KK, Varadharajan V, Collins B. Implementation of techniques for enhancing security of southbound infrastructure in SDN. In 13th international conference on network of the future 2022 (pp. 1-5). IEEE.
[44] Yungaicela-naula NM, Vargas-rosales C, Perez-diaz JA, Jacob E, Martinez-cagnazzo C. Physical assessment of an SDN-based security framework for DDoS attack mitigation: introducing the SDN-SlowRate-DDoS dataset. IEEE Access. 2023; 11: 46820-31.
[45] https://snort-org-site.s3.amazonaws.com/production/document_files/files/000/012/147/original/Snort_3.1.8.0_on_Ubuntu_18_and_20.pdf. Accessed 27 July 2023.

Nor Shahniza Kamal Bashah earned her Ph.D. degree in Telematics from the Norwegian University of Science and Technology, Norway. She currently serves as an Associate Professor at the Faculty of Computer and Mathematical Sciences, Universiti Teknologi MARA, Malaysia. Her research pursuits are focused on Mobile and Wireless Communication as well as the Semantic Web.
Email: shahniza@uitm.edu.my

Twiene Selynda Simbas holds a Master's degree in Computer Networking from Universiti Teknologi MARA. She currently holds the position of IT Network Development Advisor at Dell Global Business Center in Cyberjaya, Malaysia. Her research interests center around Cyber Security and Machine Learning.
Email: twinatwork@gmail.com

Dr. Norjansalika Janom is an Associate Professor in the Faculty of Computer and Mathematical Sciences at Universiti Teknologi MARA, Shah Alam. She is a dedicated researcher and a member of the Crowd Business and Innovation Research Interest Group at the university. Her research endeavors primarily revolve around the comprehensive domain of Information Systems Implementation, approached from a socio-technical standpoint. She exhibits a particular interest in exploring Low-Income Communities, Organizational Contexts, and Islamic Contexts.
Email: norjan@uitm.edu.my

Dr. Syaripah Ruzaini Syed Aris is a senior lecturer at the Faculty of Computer and Mathematical Sciences. She holds a Ph.D. in the field of IT Risk Management. Her research pursuits encompass IT Management, Risk Management, Information System Strategic Planning, and Crowdsourcing. Additionally, she offers consultation services pertaining to Information System Strategic Planning and Crowdsourcing. She also finds gratification in delivering training sessions related to crowdsourcing and related areas.
Email: ruzaini@uitm.edu.my

Appendix I

S. No	Abbreviation	Description
1	AI	Artificial Intelligence
2	ARP	Address Resolution Protocol
3	CPSs	Cyber-Physical Systems
4	CSP	Concentrating Solar Power
5	DDoS	Distributed Denial of Service
6	DoS	Denial of Service
7	FkNN	Fuzzy k-Nearest Neighbor
8	ForCES	Forwarding and Control Element Separation
9	HTTPS	Hypertext Transfer Protocol Secure
10	IDS	Intrusion Detection System
11	ICMP	Internet Control Message Protocol
12	IETF	Internet Engineering Task Force
13	IoT	Internet of Thing
14	IoV	Internet of Vehicle
15	kNN	k-Nearest Neighbors
16	NOC	Network Operation Centre
17	NSL-KDD	Network Security Laboratory – Knowledge Discovery in Databases
18	ONF	Open Networking Foundation
19	OPEX	Operational Expenditure
20	QDA	Quadratic Discriminant Analysis
21	R2L	Remote-to-Local
22	SDN	Software Define Network
23	SN4I	Smart Networks for Industry
24	SSNOP	Securing Software-Defined Networks Using OpenFlow Protocol
25	SVM	Support Vector Machine
26	TLS	Transport Layer Security
27	TCP	Transmission Control Protocol
28	U2R	User-to-Root
29	VANET	Vehicular Ad Hoc Network
30	VM	Virtual Machine