Hindawi
Journal of Computer Networks and Communications
Volume 2019, Article ID 8012568, 12 pages
https://doi.org/10.1155/2019/8012568

Research Article

Advanced Support Vector Machine- (ASVM-) Based Detection for Distributed Denial of Service (DDoS) Attack on Software Defined Networking (SDN)

Myo Myint Oo, Sinchai Kamolphiwong, Thossaporn Kamolphiwong, and Sangsuree Vasupongayya

Department of Computer Engineering, Faculty of Engineering, Prince of Songkla University (Hatyai Campus), Hatyai, Songkhla 90110, Thailand

Correspondence should be addressed to Thossaporn Kamolphiwong; kthossaporn@coe.psu.ac.th

Received 10 December 2018; Accepted 4 February 2019; Published 4 March 2019

Guest Editor: Mazdak Zamani

Software Defined Networking (SDN) has many advantages over a traditional network. The great advantage of SDN is that the network control is physically separated from forwarding devices. SDN can solve many security issues of a legacy network. Nevertheless, SDN has many security vulnerabilities. The biggest issue of SDN vulnerabilities is Distributed Denial of Service (DDoS) attack. The DDoS attack on SDN becomes an important problem, and varieties of methods had been applied for detection and mitigation purposes. The objectives of this paper are to propose a detection method of DDoS attacks by using SDN based technique that will disturb the legitimate user's activities at the minimum and to propose Advanced Support Vector Machine (ASVM) technique as an enhancement of existing Support Vector Machine (SVM) algorithm to detect DDoS attacks. ASVM technique is a multiclass classification method consisting of three classes. In this paper, we can successfully detect two types of flooding-based DDoS attacks. Our detection technique can reduce the training time as well as the testing time by using two key features, namely, the volumetric and the asymmetric features. We evaluate the results by measuring a false alarm rate, a detection rate, and accuracy. The detection accuracy of our detection technique is approximately 97% with the fastest training time and testing time.

1. Introduction

Nowadays, networking technologies are gradually developed for advanced infrastructure. With the development of advanced technologies, the explosion of mobile devices, server virtualization techniques, and cloud services are the strongest points in a traditional network architecture. Most traditional network architectures are hierarchical arrangement in a client-server model. Today's applications access different databases and servers in different network domains. Therefore, multiple clients and multiple server cases are expected. Thus, the traffic patterns may not be the same. Enterprise businesses' public and private cloud services want to provide the agility to access applications, infrastructures, and other IT resources on demand. This can be solved by using Software Defined Networking (SDN) to provide a network infrastructure. SDN becomes an important role in overcoming the limitations of a traditional networking. The most obvious thing in SDN is decoupling of the data plane and the control plane. The control plane is the plane that determines where to send the traffic, and the data plane is the plane that executes this decision and actually forwards the traffic. Although SDN has many advantages, some challenging issues that need to be solved still exist. One of the big challenging issues is the SDN security issue. There are many kinds of network attacks on SDN. Among them, Distributed Denial of Service (DDoS) attack is very well known and has the highest impact on SDN [1]. There are varieties of researches for detection of the DDoS attack on SDN network [2].

In this paper, we propose Advanced Support Vector Machine (ASVM) technique as an enhancement of an existing Support Vector Machine (SVM) algorithm to detect DDoS attacks. We have explored three research problems with our proposed technique [3]. The first problem is the extension of the multiclass problem in the Support Vector Machine (SVM) algorithm. If the SVM algorithm is applied in a DDoS attack detection problem on a SDN network, some of the network traffic attributes are multivalue attributes. However, the SVM is originally designed for a binary classification. Therefore, multiclass classification is a big problem for applying SVM. The second problem is the long training and testing time required for the SVM algorithm. The SVM classifier gives a low false-positive rate and a high classification accuracy. However, the SVM algorithm takes more time to train and test for the detection of the attack. The third problem is the efficiency of the SDN enabled centralized network. In previous proposed SDN architectures, the network system used only a single controller. Therefore, using multiple controllers is the most important issue for our proposed network infrastructure. Our contributions are summarized as follows.

We create test cases of the proposed model by using Miniedit and OpenDaylight controllers [4]. In the traffic generation process, we generate normal traffics, UDP flooding DDoS attack traffics [5], and SYN flooding DDoS attack traffics [6]. In the traffic collection process, we collect the traffic from each switch. In the feature generation process, we generate the volumetric features, average number of packets in a flow, average number of flow bytes and the asymmetric features, amount of packet variations in a flow, the variation of flow bytes, and the average duration of traffics in the sampling interval. In the classification process, we propose the Advance Support Vector Machine (ASVM) method. In the evaluation process, we evaluate the classification result by measuring false alarm rate, detection rate, and accuracy.

The paper is organized as follows: in the second section, we survey a number of related works to our proposed method. In the third section, we discuss the theoretical background used by our research work, Software Defined Networking and Distributed Denial of Service (DDoS) attack. In the fourth section, we present the architecture of our proposed system. In the fifth section, we provide the implementation details of the proposed detection system. In the sixth section, we briefly discuss the experimental results. We discuss the performance evaluation part of our results in the seventh section. Lastly, the eighth section concludes our work and some future works.

2. Related Works

There are two kinds of DDoS detection techniques: signature-based detection and anomaly-based detection techniques. The signature-based detection technique uses the network behaviours. The anomaly-based detection uses the machine-learning techniques. Commonly used machine learning techniques for DDoS attack detection include an artificial neural network (ANN), Support Vector Machine (SVM), Fuzzy Logic, Decision tree, Evolutionary algorithm, Navies Bayes, and k-means clustering algorithms. ANN has been used in the detection of known and unknown DDoS attacks research [7], which shows that we can detect the DDoS attack on the SDN controller with a noticeable accuracy and prevent serious damage to the controller. The perceptron neural network was used in [7], and the evaluation results showed that a significant improvement on the detection rate were achieved while a reduction in false alarm rate is also achieved in comparison with the closest previous work. Furthermore, their system was able to maintain the average detection time at an acceptable level. They would investigate an efficient method to mitigate the attack for the future work. Support Vector Machine (SVM) is used to classify the DDoS attack with normal traffic because of its high accuracy and less false-positive rate in [8]. SVM classifier was compared with other classifiers for detection of the DDoS attack and SVM provided an accurate classification than other techniques. DDoS real-time detection and the integration of the traffic pattern built in SVM with SDN controller were their future work. Fuzzy Logic can be used for real traffic detection of the DDoS attack on SDN [2]. The authors have solved the existing problems of the OpenFlow protocol. They proposed Fuzzy Logic-based DDoS mitigation algorithm that deployed multiple criterion for DDoS detection. Their system demonstrated the ability to detect and filter 97% of the attack flows with a false-positive rate of 5%. They would like to extend the OpenFlow protocol to achieve robust and faster performance.

Moreover, the researchers have designed the system to detect DDoS attacks based on a decision-tree technique, and they traced back to the approximate locations of the attacker with a traffic flow pattern-matching technique [9]. Their system could detect the attack with the false-positive ratio of 1.2%–2.4%. They conducted their experiment on the DETER system. Their results indicated that their proposed system was capable of detecting the attacks and tracing back with a high accuracy. Evolutionary algorithms (EAs) for detecting DDoS attack are presented in SDN [10]. The researchers reviewed four types of EAs that widely applied in current SDNs: Genetic algorithms (GAs), Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO), and Simulated Annealing (SA). All four EAs were compared, and the applications of these four EAs in SDNs were categorized. In order to get a good DDoS detection technique, the researchers have provided a better solution of detections using a features analysis [11]. Naive Bayes classifier algorithm was used in order to classify the packets into normal and attack packets. The use of information gain algorithm increases the performance. CAIDA 2008 and CAIDA anonymous trace 2015 datasets were used for their feature selection and classification. A method to detect a DoS attack using clustering technique with the k-means algorithm that available to be modified and developed in many possible ways was used in [12]. By using this algorithm, their result was evaluated on detection rate, accuracy, and false-positive rate. Their method has been evaluated by using DARPA 98 dataset with the satisfying result. In the future, they would like to improve in minimize false-positive rate.

3. Software Defined Networking (SDN)

Software Defined Networking (SDN) is an emergent network architecture where the network control is dynamic, manageable, adaptable, and physically separated from forwarding devices [13]. There are three layers in a SDN architecture, including the infrastructure layer (Data plane), the Control layer (Control Plane), and the Application layer (Management plane), as shown in Figure 1.

The first layer, the infrastructure layer is composed of switches. The major work of these switches is forwarding the incoming packets according to the flow tables. Forwarding decisions can be decided and configured by the control plane through the southbound protocol. The first standard of the southbound protocol is the OpenFlow protocol [14]. OpenFlow is defined from the OpenFlow switch specification published by Open Network Foundation (ONF). The second layer, the control layer, maintains a centralized view of the network and open interfaces. This layer allows applications to control the underlying networking. This layer also provides the interconnection of applications on the top and the bottom of the architecture. The third layer, the application layer is composed of applications managing and securing the underlying network. The application could be running on the controller or the application could communicate through the northbound Application Programming Interface (API) of the controller. There is no standard API for the northbound protocol. The main idea of SDN architecture is the separation of the data plane and the control plane. This network separation has many benefits in terms of the network flexibility and controllability.

Although SDN has many advantages over the traditional network, it faces some challenges. The main challenges of SDN are reliability, scalability, security, and interoperability. Among the challenges, we emphasize on the security of SDN. Each plane of SDN has vulnerabilities. In the data plane, single network devices, switches are quite vulnerable to different kind of attacks such as Denial of Service (DoS) attack, Distributed Denial of Service (DDoS) attack, data modification, repudiation, blackhole attack, and side channel attack. DoS and DDoS are the most popular attacks on the data plane so that the network cannot be accessed by the legitimate users. In the control plane, the controller is the easiest target of DDoS because the first packet of each flow must be sent to the controller, and sometimes it can cause a bottleneck condition. Moreover, some malicious attacks, DoS, blackhole, and fake flow rule generation can also occur at the control plane. In the application plane, there is some vulnerability concerning the DDoS attack, for example, in Smart City application [15]. Good features of SDN offer new opportunities to defeat attacks in cloud computing environments. DoS, DDoS flooding attacks, are the main methods to destroy the availability of cloud computing [16]. Therefore, many researchers propose solutions and countermeasures of DDoS attacks on a SDN network.

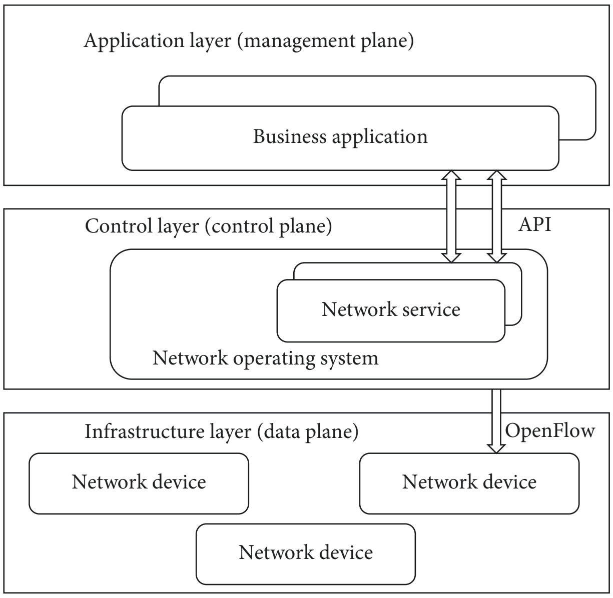


Figure 1: The structure of Software Defined Networking (SDN).

4. Distributed Denial of Service (DDoS) Attack

Distributed Denial of Service (DDoS) attack is a kind of DoS attack that the bombardment of simultaneous data is accessing to the server to hide the availability of resources in the network. According to the state of the internet security, summer 2018 report [17], the largest DDoS attack with a record peak 1.35 Tbps was observed on Wednesday, February 28, 2018. In this kind of DDoS attack, the attackers did not use any botnet network. They use weaponized misconfigured Memcached servers to conduct the DDoS attack. The attack size is more than twice that of Mirai botnet DDoS attack in 2016. This attack originated from thousand autonomous systems (ASNs) across tens of thousands of endpoints. It was an amplification attack using the Memcached-based approach producing 126.9 million packets per second [18]. The DDoS attack can be classified into three basic categories: volume-based attacks, protocol attacks, and application attacks. Under a volume-based attack, the target can be flooded with heavy traffics in order to exhaust its bandwidth. This type of attack can be detected by byte per second. Flooding attacks, User Datagram Protocol (UDP) flood, and Internet Control Message Protocol (ICMP) flood are volume-based attacks. In this research, we have been analyzed volume-based attacks. Under a protocol attack, the resources can be exhausted by exploiting the network protocol. The result of this attack is the unavailable underlying operating system. This type of attack can be detected by packets per second. SYN flood, ping of death, and smurf attacks are protocol attacks. Under an application attack, the application or server can be crashed by exploiting the application layer protocol. This attack can be detected by request per second. Hypertext Transfer Protocol (HTTP) flooding and Slowloris are application attacks [19].

The most common type of DDoS attacks include SYN flooding attack, UDP flooding, ICMP flooding, HTP flooding, ping of death attack, smurf attack, and slowloris attack. The details of each attack are as follow. SYN flooding attack can exploit the weakness of TCP connection sequence, three-way handshake [20]. At first, the host machine receives a synchronized (SYN) message to start the "handshake." The server acknowledges the message by sending an acknowledge (ACK) flag to the first host and then closes the connection. Under a SYN flooding attack, the spoofed messages are sent and the connection does not close, and the service can be shutting down. UDP flooding attack can exploit the session less User Datagram Protocol (UDP) [21]. At first, the attackers send a large amount of UDP packets to random ports on the target, and the target host checks for applications on that port. No listening application on that port is found, so it replies with ICMP destination unreachable packet. This attack can consume more resources even though the host is unreachable. ICMP flooding attack can exploit by consuming a large number of ICMP pings [22]. Under an ICMP attack, ICMP echo packets are frequently sent without waiting for any echo reply, and the target attempts to reply these ICMP echo requests. Therefore, its outgoing bandwidth can be affected. HTTP flooding attack can be exploited by using legitimate GET or POST requests [23]. Although this attack uses less bandwidth than other kinds of DDoS attacks, it can force the server to use its maximum resources. Ping of death attack can exploit IP protocols by sending malicious pings to the system [24]. This attack does not require huge data to bring down the victim; it only needs to exploit the standard protocol. Smurf attack can exploit IP and ICMP protocol by using a malware program called smurf [25]. This attack spoofs an IP address and pings these addresses on a given network using smurf. Slowloris attack can break down the server by having maximum connections with attackers [26]. At first, attackers send partial HTTP requests to the server. The server keeps the connection for these requests, and the result is DoS to legitimate requests.

5. Detection of DDoS Attack on SDN by Using Advanced Support Vector Machine (ASVM)

Under our proposed framework, the DDoS attack will be detected on the SDN network by using the Advanced Support Vector Machine (ASVM) method. The proposed research presents a customizable DDoS defence framework which generates DDoS attack alerts by considering the application's security requirements [1]. Our proposed framework has been motivated by the concept that different applications have different security requirements. From our proposed framework, a DDoS attack detection solution must include a customizable reaction mechanism for generating DDoS attack alerts. Our proposed system leverages the programming and dynamic nature of SDN and implements an adaptive DDoS protection mechanism. Figure 2 illustrates the architecture of the proposed framework.

Attackers or normal users have been sent the packets to the OpenFlow Switches. When the packet arrives at the OpenFlow switch, the packet information will be checked such as the information on the packet header fields including source port, destination port, source IP address, and destination IP address. The information of the incoming packets will be checked against the flow entries, if a match is found then a specified action can be executed. Otherwise, the packet will be sent to the OpenDaylight controller via the southbound API using a packet_in control message. Controllers are connected as a cluster. When the traffics arrived at the OpenDaylight controller cluster, they will be forwarded via the northbound API to the Detection of DDoS attack by ASVM of application layer. The packet will be classified as a DDoS attack traffic or a normal traffic. The components of our proposed framework consist of four modules including the traffic generation, the traffic data collection, the feature extraction, and the classification of attack or normal by ASVM method. Two kinds of flooding-based DDoS attacks and normal traffics are generated. We have collected the traffic data from each OpenFlow switche. The five features have been extracted and classified as DDoS attacks or normal traffics by ASVM method. The graphical representation of these modules can be seen in Figure 3.

6. Traffic Generation

The generation of two DDoS attack traffics and normal traffics is implemented in this work. Two DDoS attacks are UDP flooding attacks and SYN flooding attacks. UDP flooding attack is a type of Denial of Service (DoS) attack in which the random ports on the target's host will be flooded with IP packets using User Datagram Protocol (UDP). Under a UDP flooding attack, first, the victim's IP addresses are determined; then the source port and the destination port are initialized to 80 and 1. Each time, 2000 packets are generated. The packets interarrival time for UDP attack traffics is 0.03 seconds. Scapy, a packet generation tool for computer networks written in python language, is used for generating the packets in this work. For each random source IP address, a packet is created with the source IP and the destination IP using scapy. Scapy can forge or decode packets; Scapy can send the packets on the wire; Scapy can capture the packets; and Scapy can match the requests and the replies. Scapy can also handle tasks like scanning, tracerouting, probing, unit tests, attacks, and network discovery. After the packet is created, it must be sent to the destination IP address within the time interval. The step by step process of the UDP flooding attack on the SDN network can be seen in Figure 4.

SYN flooding attack is a type of DoS attack that exploits the normal three-way handshake procedure to consume the resources on the targeted server and render it unresponsive by using the TCP connection. Under a SYN flooding attack, the victim IP addresses, the victim Port, and the number of packets must be determined. Then, an IP packet with a random source IP and the victim IP will be generated. We also need to create the TCP packet with a random source port, the victim port, 's' flag, packet sequence, and time window. At last, both the IP and TCP

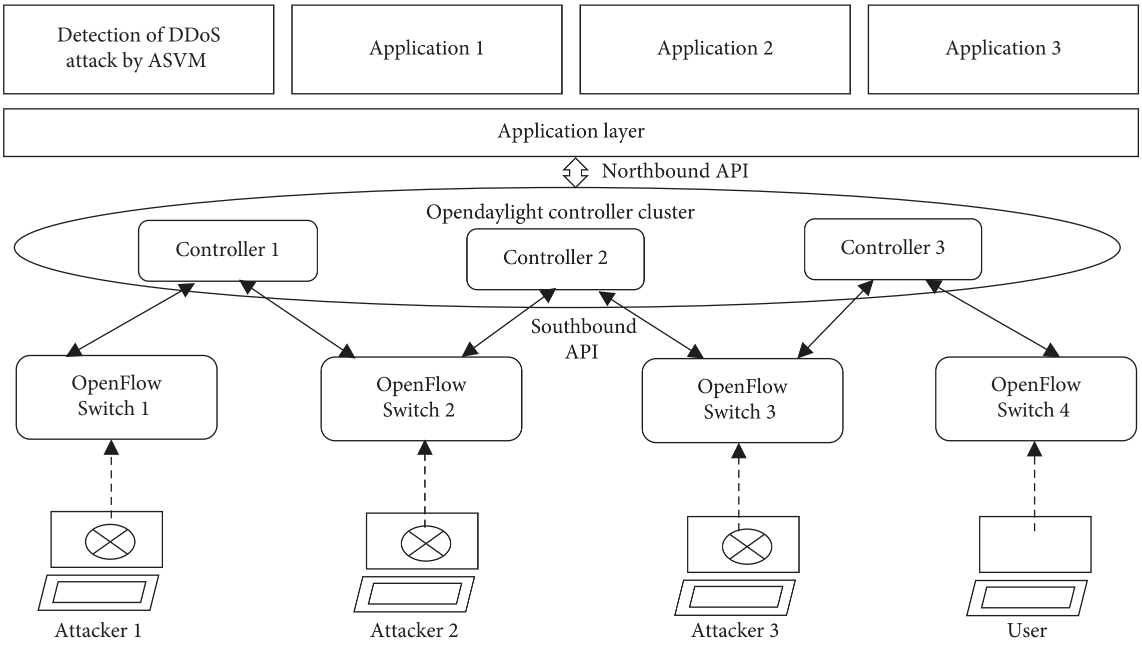


Figure 2: The proposed SDN-based DDoS attack detection framework.

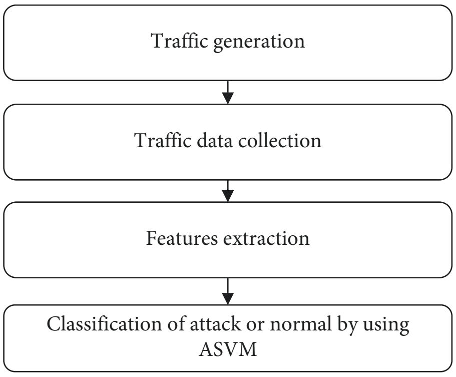


Figure 3: Four modules of our proposed system framework.

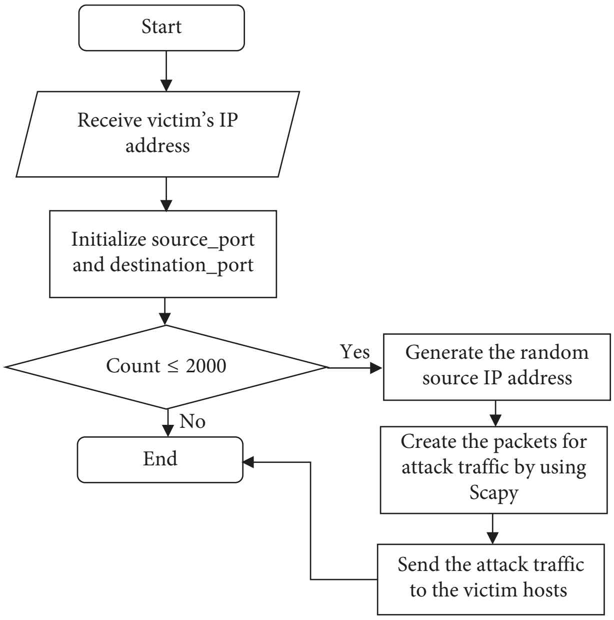

Start
Receive victim's IP address
Initialize source_port and destination_port
Count ≤ 2000
Yes
No
End
Generate the random source IP address
Create the packets for attack traffic by using Scapy
Send the attack traffic to the victim hosts

Figure 4: Step by step process of UDP Flooding Attack.

packets will be sent to the victim host. The step by step process of a SYN flooding attack on the SDN network can be seen in Figure 5. The normal traffics are also generated as shown in Figure 6. For a normal traffic generation, the last number of host's destination IP address must be determined. Each time, 1000 packets are generated because the average number of packets at a normal condition is approximately 1000 packets. The packets interarrival time for normal traffic generation is 0.1 second. The random source IP address is used each time. Scapy is also used for creating the normal traffic packets to be sent to the destination host.

7. Traffic Data Collection

For the detection of a DDoS attack on a SDN network, the traffic data collection is the main part of the system. We can collect the traffic data information through the OpenFlow protocol from the OpenFlow switches. In SDN, the traffic data are stored in the flow table within the OpenFlow switches. When we want to extract the traffic data, the OpenFlow switch responds to the onp_-flow_stats_requst message and periodically sends this request message to the controller. OpenDaylight controller is used in our research to manage and control the data-obtaining period and flow-deleting period within the time interval. We can send the flow request command, "sudo ovs-ofctl dump-flows s1" to each switch in order to collect the traffic flow information of the flow table. An example of the extracted traffic flow information from a switch is shown in Figure 7.

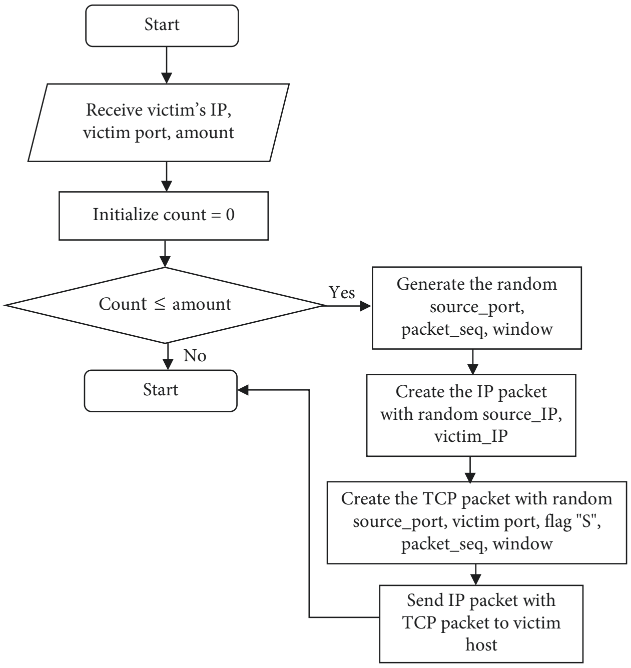

Start
Receive victim's IP, victim port, amount
Initialize count = 0
Count ≤ amount
Yes
Generate the random source_port, packet_seq, window
No
Start
Create the IP packet with random source_IP, victim_IP
Create the TCP packet with random source_port, victim port, flag "S", packet_seq, window
Send IP packet with TCP packet to victim host

Figure 5: Step by step process of SYN Flooding Attack.

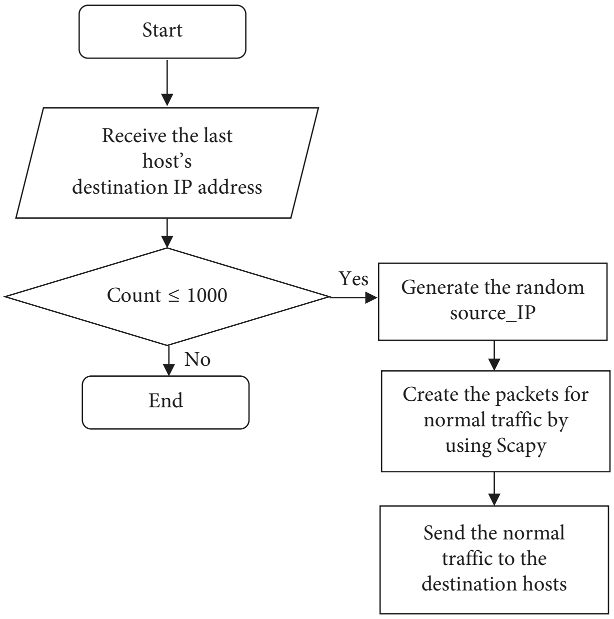

Start
Receive the last host's destination IP address
Count ≤ 1000
Yes
Generate the random source_IP
No
End
Create the packets for normal traffic by using Scapy
Send the normal traffic to the destination hosts

Figure 6: Step by step process of normal traffic generation.

8. Feature Extraction

After collecting the traffic data, the next step is the extraction of traffic features. The collected malicious traffic flows on the SDN network can be analyzed by inspecting various characteristic values of the flow table. When the traffic data from the switch is extracted, we can collect the number of packets that the host is sending, the number of bytes that the host is used, and the duration that takes for sending a packet to or receiving a packet from other hosts.

The nature of the SYN and UDP flooding attack traffics are in a form of normal distribution [27]. For volumetric and asymmetric nature of the traffic patterns, there are five different kinds of traffic features to be analyzed, including average number of flow packets in the sampling interval (ANPI), average number of flow bytes in the sampling interval (ANBI), variation of flow packets in the sampling interval (VPI), variation of flow bytes in the sampling interval (VBI), and average duration of traffics in the sampling interval (ADTI).

ANPI is the sum of the number of flow packets in each flow per total flows at the sampling interval as shown in Equation (1). ANPI is used for a detection of the DDoS attack on the SDN network because the nature of the DDoS attack is sending a large number of packets in order to disable the controller. Therefore, we can detect a malicious traffic by measuring the number of flow packets.

$$\mathrm{ANPI} = \frac{\sum_{i=1}^{\text{total flows}} \text{flow packet}_i}{\text{total flows}}. \tag{1}$$

ANBI is the sum of the number of flow bytes in each flow per total flows at the sampling interval as shown in Equation (2). ANBI is used for a detection of the DDoS attack on the SDN network because most DDoS attackers want to send the packet; they do not consider the data bytes of the packets. Thus, the flow byte measurement can indicate a malicious traffic.

$$\mathrm{ANBI} = \frac{\sum_{i=1}^{\text{total flows}} \text{flow byte}_i}{\text{total flows}}. \tag{2}$$

VPI is the measurement of the standard deviation of the number of flow packets at sampling interval as shown in Equation (3). We can detect the DDoS attack on SDN network by considering the VPI feature because most DDoS attackers randomly create the packets in order to send to the hosts; they do not consider the full data packet, and mostly empty packets are used.

$$\mathrm{VPI} = \sqrt{\frac{\sum_{i=1}^{\text{total flows}} \left(\text{flow packet}_i - \mathrm{ANPI}\right)^2}{\text{total flows}}}. \tag{3}$$

VBI is the measurement of the standard deviation of the number of flow bytes at the sampling interval as shown in Equation (4). We can use the VBI feature in the detection of the DDoS attack on the SDN network because most DDoS attackers do not consider the flow bytes of the packets. Therefore, we can detect malicious traffic by measuring the variation of the flow bytes.

$$\mathrm{VBI} = \sqrt{\frac{\sum_{i=1}^{\text{total flows}} \left(\text{flow packet}_i - \mathrm{ANBI}\right)^2}{\text{total flows}}}. \tag{4}$$

ADTI is the sum of each duration of the SDN traffic per a sampling interval as shown in Equation (5). We can detect a malicious traffic nature by measuring the ADTI feature of the SDN traffic. The DDoS attackers send a large number of packets within a certain interval. Therefore, we can measure

```
mininet> sh ovs-ofctl dump-flows s1
NXST_FLOW reply (xid=0x4):
cookie=0x0, duration=165.700s, table=0, n_packets=8, n_bytes=560, idle_age=20,
in_port=1 actions=FLOOD
```

Figure 7: An example of the traffic flow information from a switch.

the malicious traffic behaviours at each duration of the packet sending and receiving period.

$$\text{ADTI} = \frac{\text{each duration of SDN traffic}}{\text{sampling interval}}. \tag{5}$$

9. ASVM Classification of Attack or Normal Traffic

In our proposed system, the ASVM method is utilized to classify each packet to be attack or normal traffic. The ASVM method is the advanced Support Vector Machine (SVM) algorithm. SVM is a supervised machine learning algorithm that can be used on both classification and regession problems [3]. SVM is widely used in many application areas because of its high accuracy, ability to deal with high-dimensional data, and flexibility in modelling diverse data. SVM is originally used for liner two-class classification problems. In a sample linear two-class classification problem, the assumption is that there are two classes, +1 (positive class) and −1 (negative class). Small letter 'x' denotes a vector with components x_i. The dataset of n points can be shown as

$$D = \{(x_i, y_i)\}_{i=1}^{n}, \tag{6}$$

where x_i denotes the i^{th} characteristic vector in a dataset and y_i is the label associated with x_i. The value of y_i is +1 or −1. The example of linear classification by SVM is shown in Figure 8.

According to Figure 5, there is a straight line separating the vector of class +1 from the vector of class −1. This straight line is denoted as $w \cdot x + b = 0$, where the vector w is called the weight vector and the scalar b is called the bias. The hyperplane of the class label 1 above the straight line is denoted as $w \cdot x + b = 1$ and another hyperplane of the class label −1 below the straight line is denoted as $w \cdot x + b = -1$. When the dataset is linearly separable, this two hyperplanes can be seen as parallel and the distance between them must be as large as possible. The distance between them is calculated as follows:

$$\text{distance between two hyperplanes} = \frac{2}{\|w\|}. \tag{7}$$

Therefore, the distance between the planes must be maximized. As a result, $\|w\|^2/2$ must be minimized. We also need to consider the prevention of the data points from falling into the margin. We need to add the constraint for each "i" either $w \cdot X_i - b \geq 1$ if $y_i = 1$ or $w \cdot X_i - b \leq -1$, if $y_i = -1$. The constraint for each data points need to be lied at the correct side of the margin which is $y_i\ (w \cdot X_i - b) \geq 1$, for all $1 \leq i \leq n$. Therefore, the optimization problem here is minimize $\|w\|^2/2$ subject to $y_i\ (w \cdot X_i - b) \geq 1$, for $i = 1, ., n$. In practice, the data are not linearly separable. There are multiclass. Sometimes, the maximization of margin can cause an error because of a misclassification of the data. In this work, we extend the SVM with Advanced Support Vector Machine (ASVM). We need to consider the slack variables (ξ_i) and the classification error (C). Slack variable is the variable that measures the distance of the point to its marginal hyperplane [28]. The optimal problem is shown in the following equation 8:

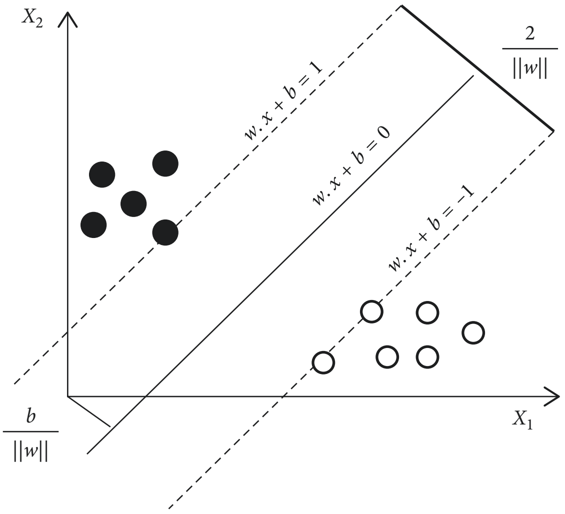


Figure 8: Linear Support Vector Machine (SVM).

$$\begin{aligned} &\text{minimize} && \frac{\|w\|^2}{2} + C\sum_{i=1}^{n}\xi_i, \\ &\text{subject to} && y_i\left(w \cdot x_i - b\right) \geq 1 - \xi\_i, \quad \xi\_i \geq 0. \end{aligned} \tag{8}$$

The classification error, $C > 0$, gives the relative importance of maximizing the margin and minimizing the amount of slack. In a multiclass classification problem, we need to consider the classifier judgment including one-versus-one and one-versus-some. In one-versus-one, the classification pattern is constructed as $n(n-1)/2$. There are two classes. The sample of the first class is trained as a positive sample and the second class is trained as a negative one. All of these classifiers are needed to classify the data in the testing phases. In one-against-some, the classification pattern is constructed such that each class is trained with the remaining $n-1$ classes. One class of the sample is denoted as positive, and all other samples are denoted as negatives. When we make a decision, it is needed to produce a real-valued confidence score. When we use the SVM algorithm in the classification problem, the most important thing is choosing the kernel function. Kernel function $K(x\_n, x\_i)$ takes the dataset into a higher dimension space in order to make it possible to separate the data [29]. The kernel function in this work is of the form

$$(x_n \cdot x_i) \longleftarrow K(x_n, x_i) = (\phi(x_n) \cdot \phi(x_i)), \qquad (9)$$

where x_n is the support vector data with $n = 1, 2, 3, 4, \ldots, N$. The most useful kernel functions of SVM algorithm are a linear kernel function, Radial Basis Function (RBF), sigmoid, and polynomial. Kernel functions are listed in Table 1.

In this system, we have detected UDP and SYN flooding attacks. Nature of both attacks is normal distribution [30]. In this work, linear kernel and OVS (one-versus-some) decision function are used for classifying the DDoS attack and the normal traffics.

10. Experimental Result and Analysis

The experiments in this work are conducted on the Mininet (version 2.3.0d1) emulator in order to create the SDN network topology on an Ubuntu 16.04 VMware. Our VMware is implemented with 2 processors, 4 MB of RAM, and 20 GB (SCSI) of hard disk. There are the varieties of different controllers: Ryu, ONOS, POX, Floodlight, NOX, and OpenDaylight. Among them, the OpenDaylight (version Beryllium) controller is used for controlling the network topology. OpenDaylight is an open source Java-based SDN controller that is supported by VMware, managed by the Linux Foundation [31]. The OpenDaylight controller has a very large platform with a lot of plugins and features. Mininet is a network emulator that runs the collection of end-hosts, switches, routers, and links on a single Linux kernel, and its results are as same as a real network [32]. Most DDoS attacks use at least three hosts, and the number of hosts can be up to approximately one hundred hosts; at least one switch is used, and the number of controllers used can range from one to as much as possible. Our SDN testbed consists of one hundred hosts (h1 to h100), nine switches (s1 to s9), and three controllers (c0, c1, c2). Four subnets are arranged in our testbed. The experiments are set up on Miniedit. Miniedit is a simple GUI editor for Mininet. Figure 9 shows our implemented testbed.

After running the testbed, the network flows have been added to the nine switches. Open Virtual Switch (OVS) and OpenFlow protocol (version OpenFlow13) are used in our testbed. OVS is a production quality, multilayer virtual switch licensed under the open source Apache 2.0 license [33]. We have been the command, for example, in switch s1 as "sh ovs-ofctl add-flow s1 in-port = 1, action = flood" at our testbed terminal. 126 flows are added for nine switches. In our testbed, each traffic type is generated from 100 scenarios. There are three types of traffics including normal traffics, UDP flooding attacks, and SYN flooding attacks. Under a UDP flooding attack scenario, we use at least five hosts to nine hosts as the attacker hosts and four hosts as the victims. Under a SYN flooding attack, four hosts are assigned as attacker hosts and only one victim host. In each scenario, the traffic generation is started first; then the traffic flow information from each switch will be manually collected from each switch. After processing the generation and the collection of traffic data for each scenario, five different traffic features are extracted in order for the ASVM to detect the DDoS attack.

Table 1: Different kernel function.

No.	Kernel function	Formula
1	Linear	$K(x_n, x_i) = (x_n, x_i)$
2	RBF	$K(x_n, x_i) = \exp(-\gamma \|\|x_n - x_i\|\|^2 + C)$
3	Sigmoid	$K(x_n, x_i) = \tanh(\gamma(x_n, x_i) + r)$
4	Polynomial	$K(x_n, x_i) = (\gamma(x_n, x_i) + r)^d$

In this experiment, the sampling traffic collection time for attack traffics and normal traffics is 200 seconds. The result of the first feature and ANPI for normal traffics are shown in Figure 10. The trend of the curve has gradually fluctuated within the sampling time. The ANPI feature of attack traffics are shown in Figure 11. During the attack period, the numbers of packets are growing rapidly. The trend of the curve is fluctuated at first, and sometimes, the value reached the highest point depending on the randomly generated attack traffic packets.

The result of the second feature, ANBI in the sampling interval for normal traffics, is shown in Figure 12. The trend of the curve is fluctuated depending on the number of flow bytes for the normal traffics. The value of ANBI for attack traffics within the sampling time is expressed in Figure 13. The attackers send a large number of packets as fast as possible, but they do not consider the data value. Therefore, the ANBI value of attack traffic is regularly from up to down and sometimes apparently reaches the highest point.

The result of the third feature, VPI for normal traffics is shown in Figure 14. Normally, the variation of the flow packets is relatively unchanged. For the attack case, however, the VPI changes rapidly. The VPI curve trend for attack traffics is shown in Figure 15. When the attacks occur within the sampling time, the variation of traffics has fluctuated, and sometimes, it reaches the highest point.

The result of the fourth feature, the normal traffics of VBI, is shown in Figure 16. The trend of the curve is gradually fluctuated, and sometimes, it reaches the lowest points at sampling time 65 and 169 seconds. When the attack occurs in the sampling time, the attackers did not consider the flow byte values of the sending packets. Therefore, the curve trend gradually grows up and down as shown in Figure 17.

The result of the last feature, ADTI for normal traffics and attack traffics, is shown in Figures 18 and 19, respectively. The curve of both types is the same, but the ADTI value of the attack traffics is apparently greater than that of the normal traffics.

The extracted features from the traffic data have been stored as the feature dataset, namely, SDNtrafficDS. The next step is the classification of these dataset by the ASVM method. The classification process is shown in Figure 20.

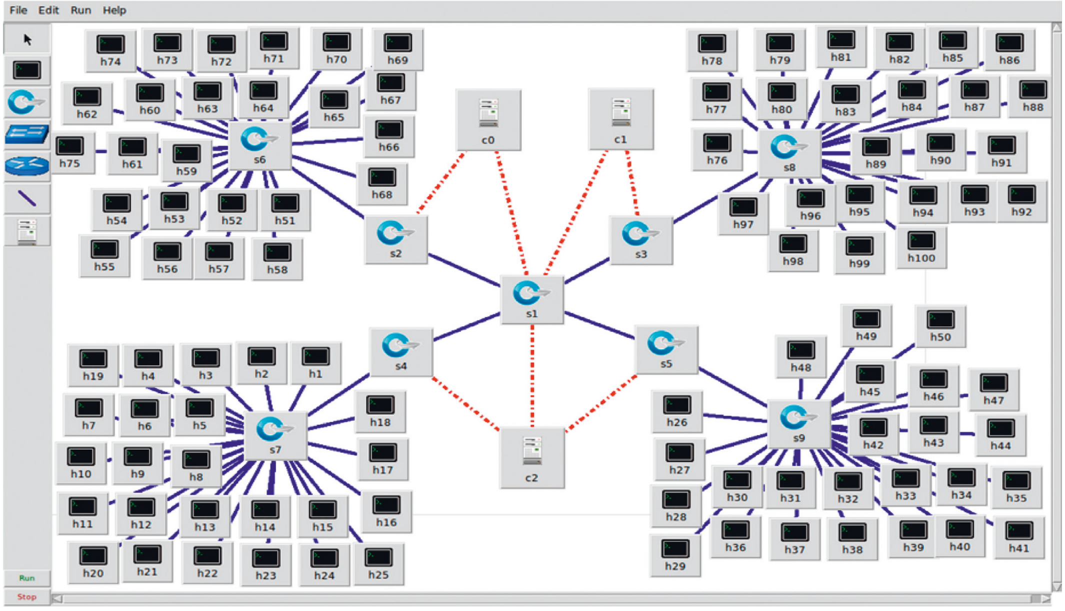


FIGURE 9: SDN testbed for detection of DDoS attack by using ASVM.

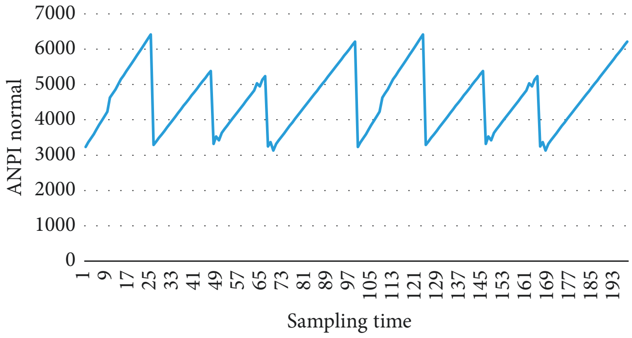


FIGURE 10: Feature of ANPI for normal traffics.

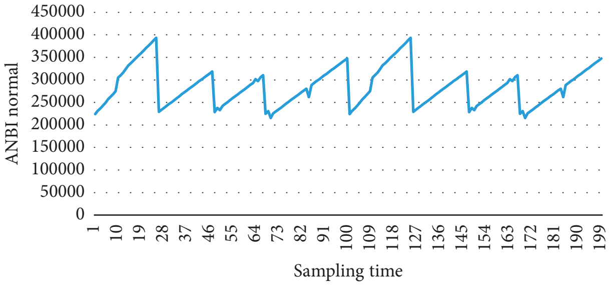


FIGURE 12: Feature of ANBI for normal traffics.

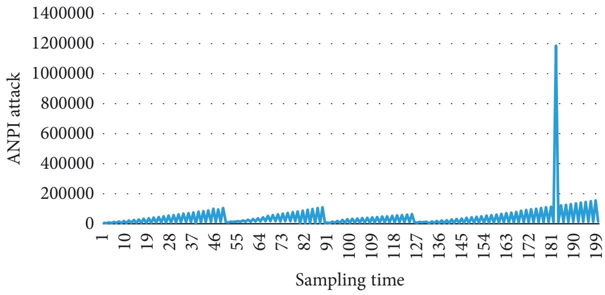


FIGURE 11: Feature of ANPI for attack traffics.

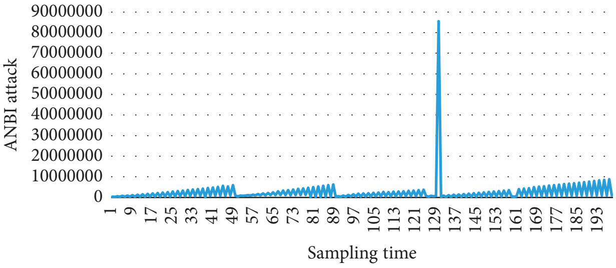


FIGURE 13: Feature of ANBI for attack traffics.

First, SDNtrafficDS is read and the Type field and the last fields is separated. The data is then split into Training DS and Testing DS using a cross-validation method in order to reduce an overfitting problem [34]. Next, the model is produced by ASVM using the Training DS. Linear kernel, OVS decision function, classification error "C" ($C > 0$), and the auto Gamma value are used in our ASVM. After the training process is done, the resulting model is used for classifying the Testing DS. The confusion matrix is used for the performance evaluation of the classification results. The classification report for three classes is generated. Lastly, the accuracy of our proposed classification result from the Training DS and the Testing DS is also generated.

11. Evaluation of Prediction Result

Training DS and testing DS are multidimensional data. We have solved our research's first problem of multiclass

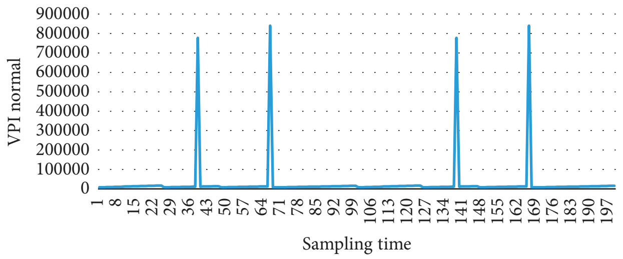


Figure 14: Feature of VPI for normal traffics.

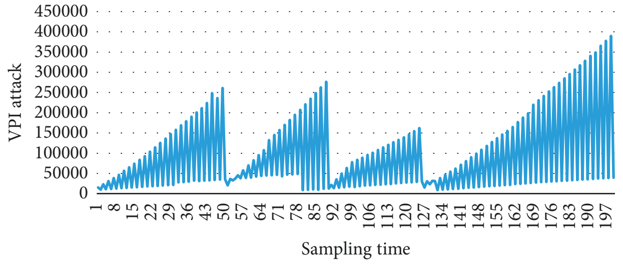


Figure 15: Feature of VPI for attack traffics.

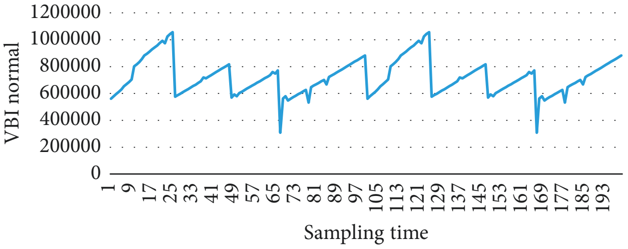


Figure 16: Feature of VBI for normal traffics.

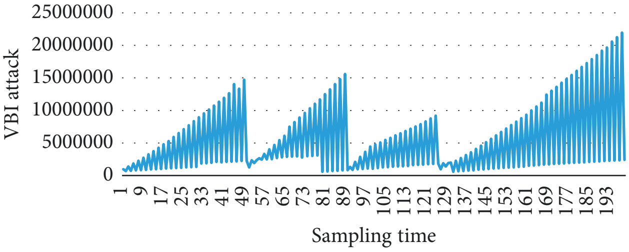


Figure 17: Feature of VBI for attack traffics.

problem extension. The second problem of the long training time and testing time of SVM algorithm has been solved by using the linear kernel with penalty parameter of the classification error term, '*C*,' considering the value of "gamma" and "OVS" decision function shape. False alarm rate, detection rate, and accuracy are used for evaluating our detection result. False alarm rate is the error rate of our detection system that is the incorrect result on a normal behaviour. Thus, less false alarm rate is preferred. Detection rate is the correct rate for detecting the malicious traffics. The higher detection rate is the better system performance. Accuracy is the measurement of the system that correctly classifies both normal traffics and malicious traffics. All three measures are shown in the following equations:

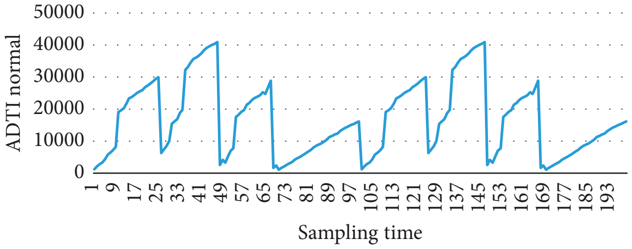


Figure 18: Feature of ADTI for normal traffics.

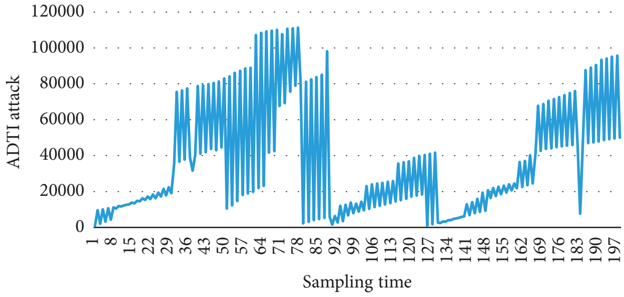


Figure 19: Feature of ADTI for attack traffics.

$$\begin{aligned}\text{false alarm rate} &= \frac{\mathrm{FP}}{\mathrm{TP}+\mathrm{FP}} * 100\%,\\ \text{detection rate} &= \frac{\mathrm{TP}}{\mathrm{FN}+\mathrm{TP}} * 100\%,\\ \text{accuracy} &= \frac{\mathrm{TP}+\mathrm{TN}}{\mathrm{TP}+\mathrm{TN}+\mathrm{FP}+\mathrm{FN}} * 100\%.\end{aligned} \tag{10}$$

True positive (TP) is the amount of network traffics that are correctly detected attack or normal traffic and forwarded. True negative (TN) is the amount of network traffics that are correctly detected and dropped. False positive (FP) is the amount of network traffics that are incorrectly detected and forwarded. False negative (FN) is the amount of network traffics that are incorrectly detected and dropped. In this experiment, we have been trained and tested with the cross-validation method of splitting rate from 10% to 90% of SDNTrafficDS. The experimental result can be seen in Table 2.

According to the experimental results shown in Table 2, the average accuracy of the detection is 0.97, the average false alarm rate is 0.02, and the average detection rate is 0.97. The training time and testing time for each rate are approximately 50 seconds and 55 seconds, respectively.

12. Conclusion

In this paper, we proposed a way to detect two flooding-based DDoS attacks using the proposed advanced SVM method. Nowadays, most researches in the detection of the DDoS attack on SDN network used traditional networking dataset. In this work, a new dataset, SDNTrafficDS,

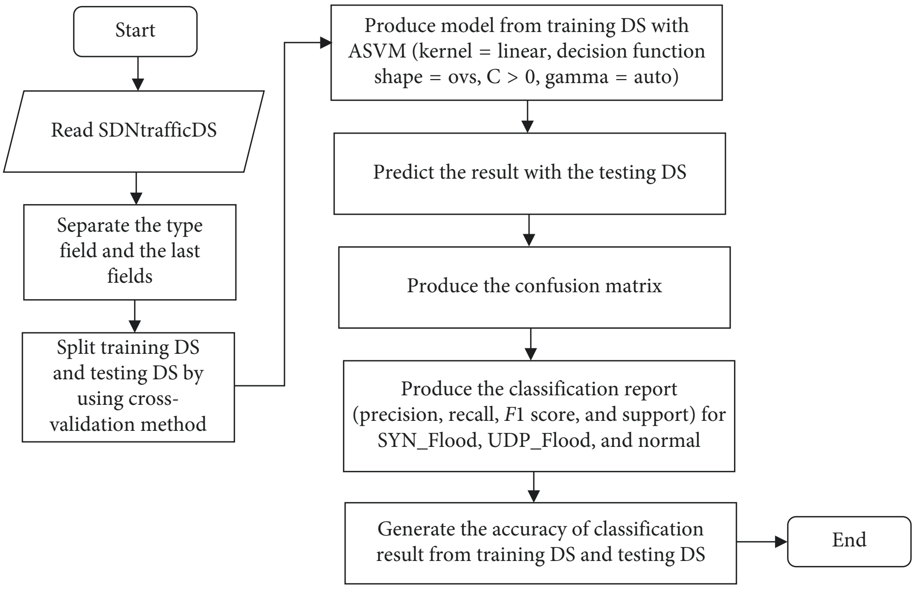

Start
Read SDNtrafficDS
Separate the type field and the last fields
Split training DS and testing DS by using cross-validation method
Produce model from training DS with ASVM (kernel = linear, decision function shape = ovs, C > 0, gamma = auto)
Predict the result with the testing DS
Produce the confusion matrix
Produce the classification report (precision, recall, F1 score, and support) for SYN_Flood, UDP_Flood, and normal
Generate the accuracy of classification result from training DS and testing DS
End

Figure 20: System flow of the proposed classification method.

Table 2: Evaluation of prediction performance of ASVM method.

Split rate	Training data (%)	Testing data (%)	False alarm rate	Detection rate	Accuracy
0.1	90	10	0	1.0	1.0
0.2	80	20	0.06	0.93	0.93
0.3	70	30	0.02	0.98	0.97
0.4	60	40	0.03	0.97	0.96
0.5	50	50	0.01	0.99	0.98
0.6	40	60	0.01	0.99	0.98
0.7	30	70	0.01	0.99	0.99
0.8	20	80	0.03	0.96	0.96
0.9	10	90	0.03	0.97	0.97

is generated and used. Our emulated testbed is conducted using Mininet. In our testbed, one hundred hosts, nine switches, and three controllers are used. The existing researches in the security of SDN network used a single controller in their network setting. In this work, on the other hand, three controllers are used. Although one controller has down because of the attack, another controller can still be used. We used one hundred scenarios for UDP flooding attack and another one hundred scenarios for SYN flooding attacks. Both malicious traffic data and normal traffic data are generated. The SDN traffics from the OpenFlow switches are collected. The volumetric and asymmetric features from the SDN traffics are collected and extracted to create the dataset. Cross-validation method is employed while training and testing the classification model. Linear kernel is used in our SVM algorithm. As a result, the training and testing time is reduced. The parameter of classification error (*C*), gamma value, and decision function shape (OVS) is considered. According to the experimental results, the overall accuracy of the proposed model is at 97%. Our future works include an online detection system for DDoS attack on SDN network. In addition, other attack planes of SDN layer must also be considered. Moreover, we would like to mitigate the DDoS attack using the lightweight method.

Data Availability

We have used our own dataset by using Mininet emulator. Our dataset is available at https://my.pcloud.com/publink/show?code=XZYm5P7ZXWd1JwSha2XTmPMtkfv2wzdXp5my.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

The authors would like to express their gratitude to their scholarship program. This work was supported by the Higher Education Research Promotion and the Thailand's Education Hub for Southern Region of ASEAN Countries Project Office of the Higher Education Commission. Furthermore, the authors would like to thank all colleagues from the CNR Lab, Department of Computer Engineering at the Prince of Songkla University.

References

[1] M. Myint Oo, K. Sinchai, and K. Thossaporn, "The design of SDN based detection for distributed denial of service (DDoS)

attack," in *Proceedings of the 21st International Computer Science and Engineering Conference*, Antalya, Turkey, August 2017.

[2] T. Dang-Van and H. Truong-Thu, "A multi-criteria based software defined networking system Architecture for DDoS-attack mitigation," *REV Journal on Electronics and Communications*, vol. 6, no. 3-4, 2016.

[3] T. Evgeniou and M. Pontil, "Support vector machines: theory and applications," *Machine Learning and Its Applications: Advanced Lectures*, vol. 2049, pp. 249–257, 2001.

[4] S. Badotra and J. Singh, "Open daylight as a controller for software defined networking," *International Journal of Advanced Computer*, vol. 8, no. 5, 2017.

[5] S. Kolahi, K. Treseangrat, and B. Sarrafpour, "Analysis of UDP DDoS flood cyber-attack and defense mechanisms on Web Server with Linux Ubuntu 13," in *Proceedings of the 2015 International Conference on Communications, Signal Processing, and their Applications (ICCSPA'15)*, London, UK, June 2015.

[6] R. Bani-Hani and Z. Al-Ali, "SYN flooding attacks and countermeasures: a survey," in *Proceedings of ICICS*, Beijing, China, 2013.

[7] F. Gharvirian and A. Bohlooli, "Neural network based protection of software defined network controller against distributed denial of service attacks," *International Journal of Engineering*, vol. 30, no. 11, pp. 1714–1722, 2017.

[8] R. T. Kokila, S. Thamarai Selvi, and G. Kannan, "DDoS detection and analysis in SDN-based environment using support vector machine classifier," in *Proceedings of the 2014 Sixth International Conference on Advanced Computing (ICoAC)*, Chennai, India, December 2014.

[9] Y. Chi Wu, H. Tseng, W. Yang, and R. Hong Jan, "DDoS detection and traceback with decision tree and grey relational analysis," in *Proceedings of the 2009 Third International Conference on Multimedia and Ubiquitous Engineering*, Qingdao, China, June 2009.

[10] L. Linxia, V. C. M. Leung, and L. Chin-Feng, "Evolutionary algorithms in software defined networks: techniques, applications, and issues," *ZTE Communications*, vol. 15, no. 3, 2017.

[11] N. Anandshree Singh, K. Johnson Singh, and T. De, "Distributed denial of service attack detection using naive bayes classifier through info gain feature selection," in *Proceedings of the International Conference on Informatics and Analytics*, Pondicherry, India, August 2016.

[12] M. I. W. Pramana, Y. Purwanto, and F. Yosef Suratman, "DDoS detection using modified K-means clustering with chain initialization over landmark window," in *Proceedings of the 2015 International Conference on Control, Electronics, Renewable Energy and Communications (ICCEREC)*, Bandung, Indonesia, August 2015.

[13] K. Benzekki, A. El Fergougui, and A. Elbelrhiti Elalaoui, "Software-defined networking (SDN): A survey," *Security and Communication Networks*, 2017.

[14] S. Kazuya, S. Kentaro, T. Nobuyuki et al., "A survey on OpenFlow technologies," *IEICE Transactions on Communications*, vol. E97.B, no. 2, pp. 375–386, 2014.

[15] N. Zakaria Bawany and J. A. Shamsi, "Application layer DDoS attack defense framework for Smart city using SDN," in *Third International Conference on Computer Science, Computer Engineering, and Social Media (CSCESM)*, Thessaloniki, Greece, May 2016.

[16] N. Dayal, P. Maity, S. Srivastava, and R. Khondoker, "Research trends in security and DDoS in SDN," *Security and Communication Networks*, vol. 9, no. 18, pp. 6368–6411, 2016.

[17] A. Akamai, "State of the internet/security," *SOTI*, vol. 4, no. 5, 2018.

[18] A. Akamai, *Memcached Reflection Attacks: A NEW era for DDoS*, Akamai Technologies, Cambridge, MA, USA, 2018.

[19] S. Acharya and N. Tiwari, "Survey of DDoS attacks based on TCP/IP protocol vulnerabilities," *IOSR Journal of Computer Engineering*, vol. 18, no. 3, pp. 68–76, 2016.

[20] M. Bogdanoski, A. Risteski, and T. Shuminoski, "TCP SYN flooding attack in wireless networks," in *Proceedings of the Conference: Innovations on Communication Theory, INCT*, Istanbul, Turkey, October 2012.

[21] S. H. Mujtiba and G. R. Beigh, "Impact of DDoS attack (UDP flooding) on queuing models," in *Proceedings of the 2013 4th International Conference on Computer and Communication Technology (ICCCT)*, Allahabad, India, September 2013.

[22] H. Harshita, "Detection and prevention of ICMP flood DDOS attack," *International Journal of New Technology and Research (IJNTR)*, vol. 3, no. 3, pp. 63–69, 2017.

[23] A. Verma and D. Kumar Xaxa, "A survey on HTTP flooding attack detection and mitigating methodologies," *International Journal of Innovations and Advancement in Computer Science*, vol. 5, no. 5, 2016.

[24] F. Yihunie, A. Eman, and A. Odeh, "Analysis of ping of death DoS and DDoS attacks," in *Proceedings of IEEE Long Island Systems*, Applications and Technology Conference (LISAT), Farmingdale, NY, USA, May 2018.

[25] S. Rajneet, "A study of DoS & DDoS-smurf attack and preventive measures," *International Journal of Computer Science and Information Technology Research*, vol. 2, no. 4, 2014.

[26] T. Lukaseder, G. Shreya, and K. Frank, "Mitigation of flooding and slow DDoS attacks in a software-defined network," in *Proceedings of Cryptography and Security*, Santa Barbara, CA, USA, August 2018.

[27] L. Tauber, "Introducing the normal distribution in a data analysis course: specific meaning contributed by the use of computers," in *Proceedings of the ICOTS 6 : the Sixth International Conference on Teaching Statistics*, Cape Town, South Africa, 2002.

[28] F. Tang, P. Tiňno, P. A. Gutiérrez, and H. Chen, "The benefits of modelling slack variables in SVMs," *Neural Computation*, vol. 27, no. 4, 2015.

[29] M.-F. Balcan, A. Blum, and S. Vempala, "On kernels, margins, and low-dimensional mappings," *Lecture Notes in Computer Science*, in *Proceedings of the 15th International Conference on Algorithmic Learning Theory*, Padova, Italy, October 2004.

[30] Y. Ahuja and S. K. Yadav, "Multiclass classification and support vector machine," *Global Journal of Computer Science and Technology Interdisciplinary*, vol. 12, no. 11, 2012.

[31] S. Asadollahi, B. Goswami, and A. M. Gonsai, "Implementation of SDN using OpenDayLight controller," in *Proceedings of the International Conference on Recent Trends in IT Innovations-Tecáfe*, vol. 52, no .2, India, April 2017.

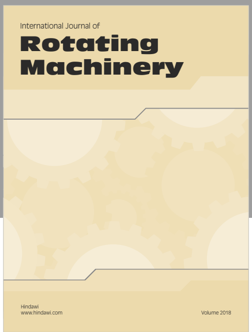
[32] F. Keti and S. Askar, "Emulation of software defined networks using mininet in different simulation environments," in *Proceedings of the 6th International Conference on Intelligent Systems, Modeling, and Simulation*, Kuala Lumpur, February 2015.

[33] B. Pfaff, "Open vSwitch," 2014, http://www.openvswitch.org//support/slides/brkt.pdf.

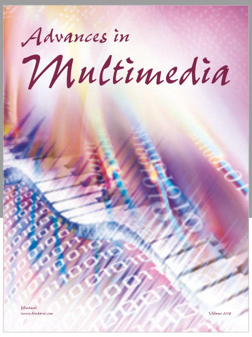
[34] A. Moore, "Cross-validation for detecting and preventing overfitting," 2001.

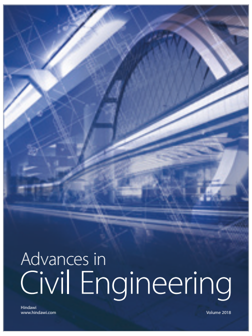
Journal of
Engineering

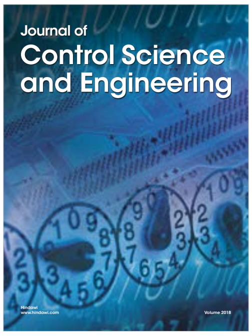
The Scientific
World Journal


International Journal of
Rotating
Machinery

Journal of
Sensors

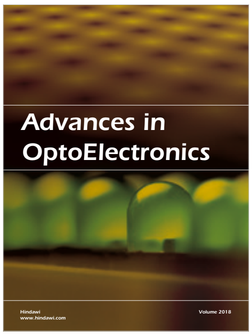

Advances in
Multimedia


Advances in
Civil Engineering


Journal of
Control Science
and Engineering

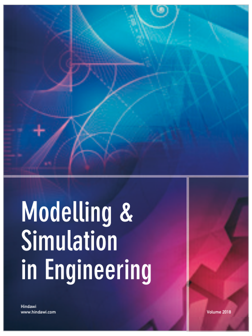
Journal of
Robotics

Journal of
Electrical and Computer
Engineering

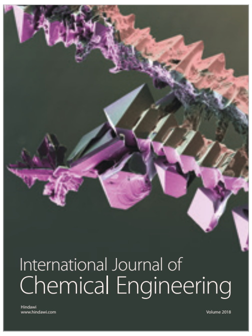

Advances in
OptoElectronics

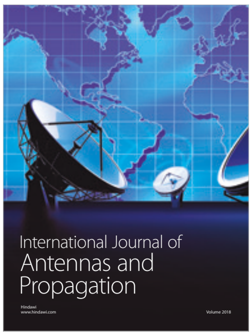
VLSI Design

International Journal of
Navigation and
Observation

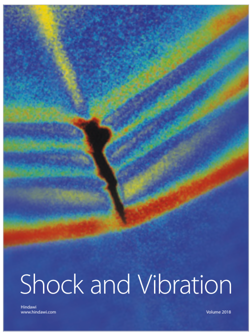

Modelling &
Simulation
in Engineering

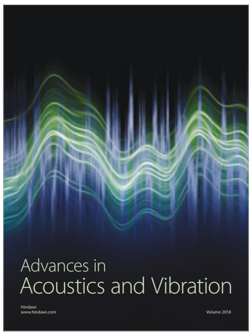
International Journal of
Aerospace
Engineering


International Journal of
Chemical Engineering


International Journal of
Antennas and
Propagation

Active and Passive
Electronic Components


Shock and Vibration


Advances in
Acoustics and Vibration