I. J. Computer Network and Information Security, 2021, 3, 15-27
Published Online June 2021 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijcnis.2021.03.02

Machine Learning-based Distributed Denial of Service Attacks Detection Technique using New Features in Software-defined Networks

Waheed G. Gadallah
Faculty of Computers and Information, Assiut University, Assiut, Egypt
E-mail: waheedgameel65@aun.edu.eg

Nagwa M. Omar
Faculty of Computers and Information, Assiut University, Assiut, Egypt
E-mail: n_omar@aun.edu.eg

Hosny M. Ibrahim
Faculty of Computers and Information, Assiut University, Assiut, Egypt
E-mail: hibrahim@aun.edu.eg

Received: 29 January 2021; Revised: 20 February 2021; Accepted: 07 March 2021; Published: 08 June 2021

Abstract: Software-Defined Networking is a new network architecture that separates control and data planes. It has central network control and programmability facilities, so it improves manageability, scaling, and performance. However, it may suffer from creating a single point of failure against the controller, which represents the network control plane. So, defending the controller against attacks such as a distributed denial of service attack is a valuable and urgent issue. The advances of this paper are to implement an accurate and significant method to detect this attack with high accuracy using machine learning-based algorithms exploiting new advanced features obtained from traffic flow information and statistics. The developed model is trained with kernel radial basis function. The technique uses advanced features such as unknown destination addresses, packets inter-arrival time, transport layer protocol header, and type of service header. To the best knowledge of the authors, the proposed approach of the paper had not been used before. The proposed work begins with generating both normal and attack traffic flow packets through the network. When packets reach the controller, it extracts their headers and performs necessary flow calculations to get the needed features. The features are used to create a dataset that is used as an input to linear support vector machine classifier. The classifier is used to train the model with kernel radial basis function. Methods such as Naive Bayes, K-Nearest Neighbor, Decision Tree, and Random Forest are also utilized and compared with the SVM model to improve the detection operation. Hence, suspicious senders are blocked and their information is stored. The experimental results prove that the proposed technique detects the attack with high accuracy and low false alarm, compared to other related techniques.

Index Terms: Software-Defined Networking, Distributed Denial of Service, Machine Learning, Support Vector Machine.

1. Introduction

In Conventional networks (CNs), the control plane is built inside the device itself besides the forwarding plane. Data routing and decision-making functions for switching or routing are completed inside the same device. Consequently, the complexity of the network device is increased which affects its performance. Also, necessary configurations for each device to function properly are done according to the device's vendor. In such a case, the configuration of each device is set individually, which causes a waste of time and effort, especially when performing network maintenance and scalability [1,2].

Software-Defined Networking (SDN) provides a new networking paradigm that changes the computer network infrastructure to solve some problems that face CNs. SDN architecture separates the forwarding plane from the controlling plane for the network layer devices (routers). It replaces the network layer device with an SDN-supported device called OpenFlow (OF) switch or Open vSwitch (OVS) which is considered the forwarding device in SDN architecture [3]. SDN increases the simplicity of the network layer device since its main function is to forward packets

not to execute routing functions. However, the implementation of the network logic or control plane is done in a major component named SDN controller which does not exist in CNs infrastructure. Accordingly, within SDN architecture, the forwarding device receives its forwarding table and other configurations from the SDN controller [4].

There are two key distinguishing properties for SDN architectures [2].

1. Decoupling of Control plane and data plane: Implementing the control plane into a basic component which is the controller instead of each network device reduces the complexity and the cost of network devices compared with CN devices in which the control plane is distributed among all core devices. Network operators do not need to configure and manage each network device individually. Instead, the controller can manage and control all network devices easier since it has a perfect view and global knowledge of the network and its behavior as a whole [2,5].
2. Network programmability: SDN provides sophisticated and suitable application programming interfaces (APIs) that facilitate developing networking applications and software programs. Applications, for instance, traffic engineering and Quality of Service (QoS) are easy to be deployed programmatically through simple programming commands and lines of codes. SDN facilitates providing and applying new configurations, troubleshooting procedures, and management operations. In contrast, CN devices are proprietary and closed which complicates the development of such network applications [2,4].

Although SDN has many benefits as mentioned previously, it has some security challenges that threaten the proper performance of the network. Security challenges in SDN may include the protection of different interfaces such as northbound APIs and southbound APIs against network attacks like packet sniffing and man-in-the-middle attacks. The forwarding devices also should be secured so that they are not controlled by different attacks like viruses and Trojan horses. They should not represent severe security vulnerabilities for other network components. Besides, SDN has a single point of failure problem against the controller. If the SDN controller drops, the network will not perform its necessary functions, hence, the entire network will drop as well. Therefore, it is much important to secure the SDN controller against severe attacks such as the distributed denial of service (DDoS) attack [5].

The DDoS attack can be classified as one of the main threats that may attack the controller and make it unavailable [6]. For that, it is very urgent and embarrassing that we face the problem of DDoS attacks and that we work to solve it. In a DDoS attack, the main attacker tries to hijack and control many devices called handlers or slaves that host and execute the attacking software. Handlers can compromise multiple agents called zombies that execute the attacking software and produce packet floods targeted to the victims. A large number of packets with spoofed source IP addresses are sent through the network to attack the controller through the network. When a packet reaches a forwarding element, usually, it will not find a match with any entry of its flow tables since the packet has a spoofed source IP address. Hence, the forwarding device will send a Packet-In message, including the packet (or only its headers) to the controller [6,7]. This behavior results in the delivery of many messages to the controller. This traffic of DDoS spoofed messages can bind and fully exhausting the resources of the SDN controller. This will cut the connection between the forwarding devices in the network and the controller, causing it unavailable for the newly arriving legitimate messages. In such a situation, the entire SDN architecture may lose its controlling plane since the controller is considered the main control plane unit in SDN architecture [8]. Consequently, the whole network will fail and will not work properly. Therefore, it is an important issue to secure a controller and make it always available.

Many approaches are adopted to detect SDN DDoS attacks such as entropy [9], traffic pattern analysis [9], and machine learning (ML) methods [9]. ML is considered a strong and common approach to detecting SDN DDoS attacks. It has many popular methods that are used to create a model trained using data set and then the generated model may be used for detecting the attack. Naive Bayes [10] is a method that depends on the Bayes theorem in completing its calculations. It is fast, but features must be independent [10]. Support Vector Machine (SVM) [10,11] is an efficient ML classification algorithm. SVM uses a dataset to perform training, with each record labeled with one of two classes. It extracts fields from training records. SVM creates a classifier that can categorize a new data record. SVM selects the extreme vectors (support vectors) that support creating the hyperplane. It is fast and preferable for classification into two labels, complex problems, and data with large dimensions. Also, it requires less memory space and less prone to the overfitting problem. However, choosing a convenient kernel is complex, besides, SVM results may be unclear [10,11]. K-Nearest Neighbor (KNN) [10] is another ML method. It classifies the instance by considering its neighbors. The specified class is chosen due to being the most frequent among its K (an integer number) nearest neighbor. On the advantages side, KNN implementation is easy, effective, and robust. On the disadvantages side, it suffers from the high cost of its calculations, and K's value must be determined. The Decision Tree technique [10] is an algorithm that takes data of attributes along with its classes and generates a set of rules that may be used to identify the data. It creates a model that looks like a tree form. This method is simple and can deal with data that is either numerical or categorical. Unfortunately, the generated tree can be complex, and unstable [10]. Random Forest [10] is an ensemble method that may be used with decision trees. These decision trees may be created while the training process and the result may be regression or classification. This algorithm reduces the classifier's overfitting problem. However, it is slow and its implementation is difficult [10].

There are existing solutions that try to solve the DDoS attack problem in SDN. Some of them use entropy such as

the system introduced in [6] which uses a threshold to calculate an entropy value. The main limitation of this method is the lack of adaptability and being not effective for all network states. On the other hand, some existing solutions use traffic pattern analysis to implement a detection technique as done in [12]. The authors in [12] proposed a solution based on Self-Organizing Maps (SOM) and collected various flow statistics and information from the data plane to create a detection model. The main limitation of this work is that it does not handle the condition of tracking the effect of the attacking packets on the controller. In particular, packets in a DDoS attack usually contain spoofed or zombie-originated source IP addresses. Therefore, large numbers of packets are sent to the controller to make suitable decisions, which may flood and drop the SDN controller. However, some introduced works focus on using ML algorithms, which prove to give the best detection results. They construct a classifier that detects DDoS attacks in SDN such as the technique introduced in [13]. The method uses KNN to create an ML-based model that can detect the attack in SDN architecture. This technique suffers from taking a lot of time for detecting the attack, which is not efficient.

The major contribution and research objectives of our paper are to solve the problem of DDoS attacks in SDN by detecting it with high accuracy and low false alarm rates and then mitigating its serious negative effects. In addition to that, it is very important to increase the efficiency of the detection method. Therefore, through this paper, we hope to implement a detection technique based on ML to solve a DDoS attack problem against the controller in SDN using new features that are new and effective to the best knowledge of the authors. Features are obtained from traffic information and statistics. ML-based algorithms such as Naive Bayes, KNN, Decision Tree, and Random Forest, in addition to SVM, are compared and used to create a classification model. The new features are utilized to train the model. Once the classifier is constructed, it may be used by the SDN controller for DDoS attack detection. Thanks to the proposed model, the DDoS attack is detected more accurately. As a mitigation procedure, the introduced technique blocks hosts which send the attacking packets, and stores various information and statistics about them. Additionally, the controller installs new flow rules that drop any packet that has the same flow as the packet flow, to avoid future attack attempts.

The paper has section 2, which introduces the related work. The proposed work and methodology are explained in section 3. The obtained results and comparisons with related works are proposed in section 4. Section 5 provides the conclusion of the work.

2. Related Work

This section describes the most recent techniques that are proposed to detect DDOS attacks against the SDN controller.

The authors in [12] proposed a solution based on SOM. They depended on gathering some flow statistics from forwarding devices. The parameters used for training SOM are considered from the switch perspective, not the controller view. They include the average number of packets in the flow, average bytes in the flow, average duration in the flow, percentage of pair flows, the growth of a single flow, and growth of single ports [12]. This technique suffers from severe limitations. It violates SDN principles since it builds intelligence in the data plane. Besides, the efficiency of the technique has to be enhanced.

Another system was proposed aiming at detecting attacks of DDoS in SDN controlling plane depending on computing a value of entropy [6]. A threshold was selected experimentally and used by the authors of this system. The approach has severe limitations and may not be reliable for all conditions. For instance, when most or all of the devices in the network are attacked, the entropy would have a small value, so this method will fail to detect the attack [6]. Therefore, the provided technique suffers mainly from a lack of stability and adaptability to network state changes.

A technique that detects Denial of Service (DoS) attack, which runs on top of the NOX controller is presented in [14]. The introduced technique is called Sguard and has two modules. The first module is the access control module. This module uses authorization information to take protective steps to locate the genuine source of a packet. Information is collected, such as source medium access control (MAC) address, source IP address, source port number, and switch ID. Sguard searches for the data obtained against hash table entries. This module permits normal packets, but, rejects malicious traffic. The classification operation is the second module. This module executes an artificial neural network algorithm, which is SOM to decide the class of network traffic, using a feature vector. The introduced technique is lightweight, and effective in detecting the attack. However, it suffers from the large training time, which has to be reduced. Also, its detection accuracy is modest and has to be improved [15].

A detection method of the DoS attacks in SDN, which has two stages is implemented by the authors in [16]. In the first stage, the packet rate is computed using the statistics of a flow gathered by the controller. The ML-based model detects the attack in case the packet rate is greater than a predefined threshold. This approach is effective in detecting the attack. However, it suffers from severe limitations. It takes a long duration of time to detect the attack since the approach includes two stages. Also, its detection accuracy needs improvement.

The authors in [17] provided a method for detecting anomaly SDN streams in SDN architecture. They applied their methodology to DDoS attack anomaly detection. They used the DPTCM-KNN method [17] as the core algorithm of the flow classification and attack detection. The introduced technique is effective, but, the obtained detection accuracy and classification performance need to be improved.

A technique focused on the classification of traffic that defends against DDoS attacks is introduced in [18]. The

authors use function virtualization in SDN to reduce effects caused by DDoS attack traffic. The detector has three main components, which are a trigger of attack detection, detection of an attack, and attack backtracking. The algorithm is efficient. However, it has to improve its time complexity, which is very large. Besides, the attack backtracking module needs to be tested in case of a high traffic load. For that, the technique lacks stability and being fit to different network conditions [18].

A technique that detects a DDoS attack in SDN is introduced in [13]. The algorithm depends mainly on K-means++ [13] and Fast K-Nearest Neighbors (K-FKNN). The modular detection system is implemented in the controller. The experimental results indicate that the proposed technique is efficient and that its detection is essentially stable. However, it takes long durations of time to classify and detect the attack as well as it puts a high load on the SDN resources [13].

A detection method against DDoS attacks in SDN is implemented in [19]. The proposed system has two parts. Through the first part, normal traffic and DDoS attack traffic were analyzed on the SDN environment with the dataset. In the second part, filtering, wrapping, and embedded-based feature selection methods were applied to get the most effective features on ML model classification. The decreased feature dataset gotten by the feature selection methods was input to each classifier, hence the classifier's accuracy could be traced. However, the introduced technique suffers from a severe limitation since it needs to enhance its detection accuracy and performance [19].

In [20], a DDoS attack detection along with a mitigation method in SDN is implemented. The system is called LSTM-FUZZY. Three phases complete the detection system: characterization, anomaly detection, and mitigation. The modular design of the system allows the maintenance and adaptation of other techniques to characterize traffic, detection, and mitigation of anomalies in SDN. However, the proposed system suffers from limitations since it does not consider other network vulnerabilities. Also, the work lacks testing of more different topologies [20].

The authors in [21] present a method that uses Snort Intrusion Detection System (IDS) and deep learning-based model. This paper exploits information that is based on sFlow and adaptive polling [21]. This technique gives promising results, although its detection performance and accuracy still have to be improved.

The authors in [22] presented a time and space-efficient technique to detect SDN DDoS attacks. The method tracks characteristics of identifying hosts, which are compromised by the attack origin. Different traffic statistics are used to identify abnormal behavior. After that, the method uses hash functions in switches and stores integer variables for each host. Hence, a threshold is set and used to classify compromised and normal hosts. The technique is efficient since it saves time and space [22]. However, it has limitations such as violating the SDN standard since it implements logic into switches. Also, the authors have to improve the detection performance and accuracy of the algorithm.

3. Proposed Work and Methodology

In the proposed work, a DDoS attack detection technique is implemented based on ML in the controller in the SDN environment. We use some efficient ML-based algorithms such as Naive Bayes, KNN, Decision Tree, and Random Forest, in addition to SVM to create classification models. The technique uses advancing new traffic-based flow features to create the model, which can classify SDN flow packets as normal or a DDoS attack. Features are gathered from traffic flow headers and statistics.

The methodology has the following steps:

3.1. Generating normal and DDoS attack traffic

The proposed algorithm begins with creating packets that have the necessary header fields. Some of the generated packets are normal traffic packets and others are a type of DDoS attack traffic. The headers, besides the resulted flow statistics, can be used as features to detect a DDoS attack. Some of the features are new and others are from existing studies. The advancing new features are the following:

1. The destination IP address: In the case of a normal traffic packet, the destination IP address is usually well known to the controller. But, for DDoS traffic, it can be unknown. The attacker can plan to transmit a large number of packets in a short time duration that is directed to unidentified hosts. Hence, the controller receives more packets asking for service. In this case, the SDN controller's resources may be hanged down, which makes the controller not available to valid Packet-In messages. We can trace the time duration between the arrivals of sequential packets, which all have unknown destination addresses. When the values of these inter-arrival times are small enough, this will indicate a DDoS attack.
2. The inter-arrival time between successive packets: In the normal case, the value of this parameter would be relatively large (e.g. 0.1 seconds or more) since, within the normal state, the traffic is usually not large and not overwhelming the network. In the opposite case, for a DDoS attack, there is a flooding of packets that are transmitted over the network targeting its components especially its controller. For this reason, the inter-arrival time between every two successive packets is usually small (such as 0.01 seconds) in the case of DDoS traffic.
3. Type of service (ToS) header field: This attack tends to specify the type of service header field of the attack packets to high values such as the value ranges from 4 to 7. This procedure may enable these packets to be

served in higher precedence than other normal traffic packets.

4. Transport layer protocol (TLP) header field: A DDoS attack may use the transport layer protocol header field which is one of the network layer headers to send DDoS attack traffic. The attacker specifies this value with null instead of using a protocol such as "TCP", "UDP", or "ICMP". The attacker acts this behavior to enable the attack packet to avoid security procedures implemented within "TCP" and "UDP". When the target host exposes this attack, its resources would be frequently exhausted. This DDoS attack is called IP null attack [23].

The features used from related works include the following:

1. Average payload length of packets (L_{avg}): The size of the payload part of the packet can be used as a feature for the existence of a DDoS attack when it is small such as 120 bytes [12]. To increase its load efficiency and harmfulness, a DDoS attack usually creates small-sized packets, so that it can create the largest possible number of packets in a small interval of time [12]. We compute the average payload lengths of the packets that have the same source IP address and are sent within a specific interval of time using the following equation.

$$L_{avg} = \sum_{i=1}^{N} \frac{L_i}{N} \tag{1}$$

where, L_{avg} is the average payload length of packets with the same source IP address, L_i is the payload length of packet *i*, and *N* is the number of flows.

2. Average number of packets of flow (*ABF*): The DDoS attack usually tries to hide its identity on the internet by spoofing the source IP address. In this case, the source IP addresses of the attack packets vary continuously, which complicates the mission of tracking the attack's original address. According to the definition of a flow, variation in the source IP address field would result in generating a new flow. Hence, in the case of a DDoS attack, the average number of packets that belong to a specific flow in the switch would be small (e.g. 3 packets) [12]. *ABF* is calculated using equation (2).

$$ABF = \sum_{i=1}^{N} \frac{M_i}{N} \tag{2}$$

where, *ABF* is the average number of packets in a specific flow, M_i is the number of packets in flow *i*, and N is the number of flows.

The previous features are used to create a dataset that includes 50,000 records for normal traffic in addition to 50,000 records for attack traffic.

3.2. Proposed model

The dataset is divided as 70 % of it is utilized to train the classifier and the other 30 % are used to test it. It is mainly structured as a table, which has the number of created packets as the number of rows, and the number of features plus the classification label as the number of columns. There are six features in addition to one label, which shows either normal or DDoS attack conditions.

Many efficient ML-based methods are customized and used to create classification models. For instance, we use SVM, which seeks to obtain the best dataset's generalization. Therefore, it looks for the optimal hyperplane that can accomplish that [24,25]. It creates a model that may determine whether a new data entity belongs to some class or not. Assume having a dataset S . $S = \{(X_1, y_1), ..., (X_1, y_1)\}$ where, $x_i \in R_n$ and $y \in \{-1, +1\}$

The x_i is the entered transformed vector and y_i is the output. SVM has only two class labels which are represented as -1 and +1.

SVM designs the hyperplane H, which is optimal, so that it can decouple the input vectors into two different entities. The hyperplane (H) is described by equation (3).

$$x_i \in R^n : (\vec{w} : \vec{x}) + b = 0, \vec{w} \in R^n, b \in R \tag{3}$$

SVM should look for the hyperplane, which results in the largest possible distance between training samples as depicted in equation (4).

$$f(\vec{x}) = sign(\vec{w} : \vec{x}) + b \tag{4}$$

SVM considers that the data can be separated into two categories. Therefore, the optimal hyperplane may be mixed by the following inequality.

$$y_i\{(\vec{w},\vec{w})+b\} \geq 1, s.t.i = 1,...,n \tag{5}$$

Hence, the problem of optimization may be represented as in Equation (6).

$$\min imization \frac{1}{2}(w^T, w) s.t. y_i(w.x+b) \geq 1 \tag{6}$$

Inversely, whether data can not be separated, the optimization is shown by equation (7).

$$\min imization \frac{1}{2}(w^T, w) + c\sum_{i=1}^{n} \xi_i s.t y_i(w.x+b) + \xi_i \geq 1; \xi_i \geq 0 \tag{7}$$

where, ξ is the slack operand which enhances choosing the best hyperplane, and c is the cost magnitude which represents the regularization factor. The user may select the value of c on trial.

We utilize a linear Support Vector Classifier (SVC) with a radial basis function (RBF) as a kernel to determine the optimal decision border that can separate 6-dimensional space into two classes. The functionality of the RBF kernel for two entities X_1 and X_2 is to calculate the closeness between them. It is described in equation (8).

$$K(X_1 + X_2) = \exp(-\frac{\| X_1 - X_2 \|^2}{2\sigma^2}) \tag{8}$$

where, σ^2 is the variance, ($X_1 - X_2$) is the Euclidean distance between X_1 and X_2.

RBF Kernel is well-known since it is similar to KNN Algorithm. It has the merits of KNN and solves the problem of needing a high space cost. While the training operates, SVM's RBF Kernel has to maintain only the support vectors instead of the dataset as a whole. RBF Kernels have two hyperparameters, 'C' for SVM and 'γ' for the RBF Kernel. Here, γ is inversely proportional to σ as in equation (9).

$$\gamma \alpha \frac{1}{\sigma^2} \tag{9}$$

In addition to SVM, methods such as Naive Bayes, KNN, Decision Tree, and Random Forest are also utilized and compared. SVM is the preferred one to be used later for testing operations since it has the highest accuracy. The proposed model for DDoS detection using SVM is expressed in Fig. 1.

3.3. Detection process

Our DDoS attack detection technique is built into the controller. The controller receives an OF packet from the OF switch. First of all, the source IP address of the ongoing flow packet is extracted and compared against IP addresses in a blacklist. If this address exists, the packet will be dropped and valuable statistics such as the number of times this source address sent a packet, the time stamp, and other valuable information are logged and taken into account. Conversely, if there is no match between the source address and any element of the blacklist, the packet headers along with the necessary statistics are used as features, which are entered as input to the classifier for testing purposes. If the classifier decides that the incoming packet is normal, this packet will be accepted and served in a normal manner. On the other side, if the newly arrived packet is classified as a DDoS attack, the source IP address will be added to a blacklist even if it is the first time for the host with that source IP address to be classified as an attacking device. Also, packet information is recorded, so any suspicious packet that arrives with the same source IP address is discarded. Besides, various traffic flow statistics are logged, such as packet arrival time stamps, as well as the number of packets sent from the same host. Fig. 2. shows an overall detection architecture. Algorithm 1 describes the proposed technique. The description of each component of the proposed detection architecture is as follows.

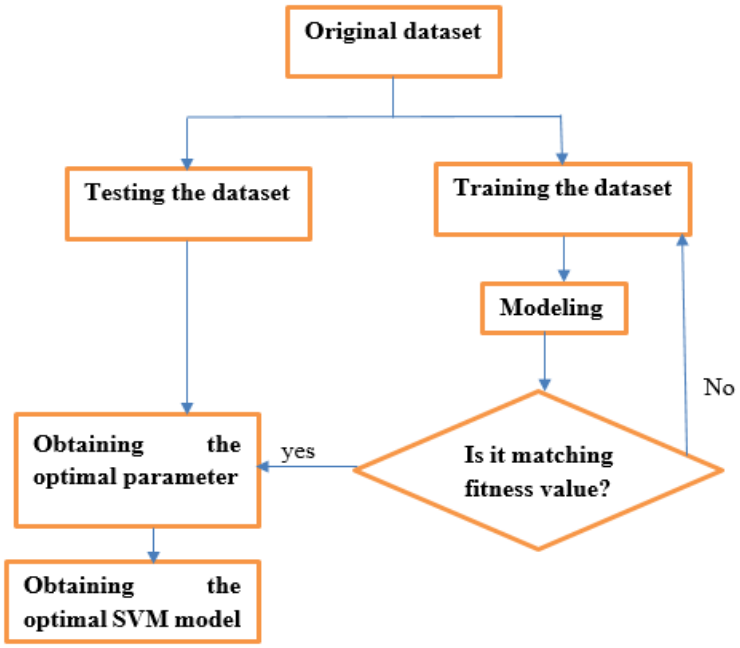

Fig.1. Flow diagram of the proposed SVM model

Packet-in message receiver: This component is responsible for receiving Packet-in messages from OF switches.

Headers reader: It reads information of various headers of the incoming packet.

Features extractor: This component applies calculations on the packet and also various traffic statistics to extract features.

SVM model: It runs the classifier on the obtained features to determine either the attack exists or does not.

Normal service module: This component runs whether the traffic is normal, to the packet is served by the controller in a normal manner.

Mitigation module: It means blocking the attack source, logging its information, and installing flow rules to drop all incoming packets in the future with the same flow.

Algorithm 1 Proposed DDoS Detection Technique

Input:

Output: Attack Detection

```
1. while true  do
2.     Packet-In message is sent  by an OF switch to the controller
3.     Extracts source IP address from the packet
4.     If (source IP address matches an entry in DDoS attack's blacklist)
       then
5.       Log necessary statistics about the packet and its source IP address
6.       Set flow rules to drop the packet, and send them to suitable switches
7.     else
8.       Store packet arrival time for upcoming calculations
9.       Extract packet's destination IP address, port numbers, transport
         layer protocol, type of service, and payload length headers
10.      Obtain features by performing necessary calculations using
          information extracted from headers and traffic information
11.      Input features to the SVM model
12.      If (The model detects a DDoS attack) then
13.        Add the source IP address to the blacklist
14.        Log necessary statistics about the packet and its source IP
           address
15.      else
16.        Serve the packet normally and store necessary statistics about it
17.      end if
18.    end if
19.end while
```

4. Simulation and Result Analysis

4.1. Experimental and simulation setup

The experiments were carried out on a PC, which is running Ubuntu 18.04.2 LTS OS. The system has a CPU of Intel@ Core i7 CPU E7500 @ 3.4GHZ x 2 and a 7.9 GiB memory.

SDN simulation involves a variety of controllers like Pox [26], Ryu [26], and OpenDaylight [26]. In our case, the Pox controller is the preferred option. Pox controller is implemented in python. POX is a common choice for SDN researchers since it is fast, lightweight, and can be customized for a specific use. Pox is the upgraded version of its predecessor NOX controller [27].

The network emulator used for this research is Mininet [28], which is considered the standard network emulator for the SDN environment. Using Mininet, a tree-type network with depth 2, which has 9 switches, and 64 hosts were created as illustrated in Fig. 3. For forwarding elements, OVS [3] was used. For packet generation, scapy [29] is used.

Scapy is a very effective packet generation, search, sniff, strike, and packet forging tool. There were two kinds of traffic generated: normal and attack traffic. Scapy is used also for spoofing the source IP address. Nine hosts are compromised to send the attack traffic. The remaining hosts send normal traffic packets.

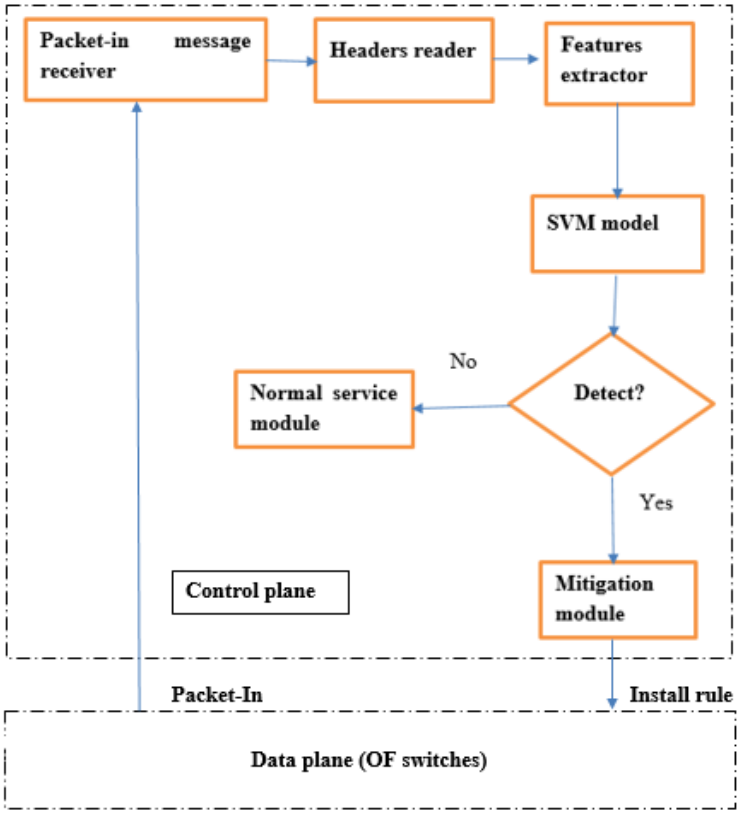

Fig.2. Flow diagram of the proposed detection architecture

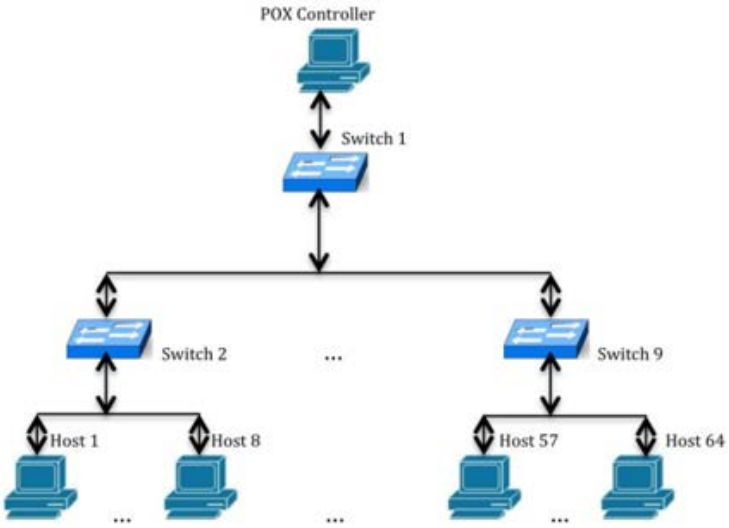

Fig.3. The network topology

4.2. Result analysis

To track the technique's performance, we should consider the following.

a) True Positive (TP): it is the number of attack packets that are truly considered as an attack.
b) True Negative (TN): it is the number of normal packets that are truly classified as normal traffic.
c) False Positive (FP): it is the number of normal packets that are considered by fault as attack traffic.
d) False Negative (FN): it is the number of attack traffic packets that are classified by fault as normal.

Classification accuracy: it is the fragment of the number of correctly classified packets to the aggregate number of packets.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{10}$$

Detection rate, hit rate, recall, or true positive rate (TPR): it is the fraction of the number of attacking packets that are truly classified as an attack to the aggregate number of attack packets.

$$TPR = \frac{TP}{TP + FN} \tag{11}$$

False-positive rate (FPR) or False alarm rate: it is the fraction of the number of normal packets that are considered by fault as an attack to the aggregate number of normal packets.

$$FPR = \frac{FP}{FP + TN} \tag{12}$$

Precision: it is the fraction between the numbers of attacking packets that are correctly classified as attack and the total number of packets that are classified as attacking packets.

$$\Pr ecision = \frac{TP}{TP + FP} \tag{13}$$

F1 score, F-score, or F_measure: it tracks the accuracy based on the combination of precision and recall.

$$F\_measure = \frac{2 \times \Pr ecision \times recall}{\Pr ecision + recall} \tag{14}$$

SDN researchers always seek to implement security techniques that can get the highest possible detection accuracy,

detection rate, precision, and f_measure. Conversely, the false alarm rate is preferred to be as small as possible.

The work began by launching 70,000 packets originated from hosts in the simulated SDN network. Different packet headers and valuable traffic statistics were extracted and used in various calculations to get features. Therefore, the dataset has been created consisting of the obtained features. The dataset was input into ML-based algorithms to create classification models. Based on experimental results, SVM gives the best accuracy among the other ML-based technologies used. Table 1. depicts the detection accuracy of various algorithms.

Table 1. Detection accuracy of various algorithms

Algorithm	Detection accuracy (%)
SVM	99.84
KNN	98.96
Decision Tree	99.26
Naive Bayes	77.64
Random Forest	99.19

After that, simulating the detection process was done. Through the detection operation, 30,000 packets were launched from hosts in the network such that 14987 DDoS attack packets were sent from 9 compromised hosts and 15,013 normal packets from the remaining hosts. Based on experimental results, there are 14982 true negative packets. Additionally, there are 17 false-negative packets, 31 false-positive packets, and 14970 true-positive packets. Hence, various performance metrics were used to measure the performance of the introduced technique as a function of the research objectives. Therefore, the detection accuracy, detection rate, false alarm rate, and f_measure were calculated. To complete achieving the research objectives, the compromised hosts were blocked and their information was logged by the SDN controller as a mitigation mechanism to decrease the harmful effects of the DDoS attack.

Table 2. as well as Fig. 4., Fig. 5., and Fig. 6. show comparisons of accuracy, detection rate, and false-positive rate for SOM [12], entropy technique implemented in [6], Sguard [14], Two-stage approach proposed in [16], DPTCM-KNN [17], RL-RF [18], K-FKNN [13], detection systems implemented in [19], LSTM-FUZZY [20], detection technique implemented in [21], detection technique implemented in [22], as well as the technique introduced in this paper. It can be shown from the results that the proposed technique in this paper gives the best results compared with other techniques.

Table 2. Accuracy, detection rate, and false-positive rate comparisons of different algorithms

Algorithm	Accuracy (%)	Detection rate (%)	FPR (%)
SOM	NA	98.86	0.52
Entropy [6]	96	NA	NA
Sguard	NA	99.77	NA
Two-stage approach [16]	99.25	NA	0.26
DPTCM-KNN	98.2	96.98	6
RL-RF	99.54	NA	0.7
K-FKNN	NA	97.5	NA
Detection technique [19]	98.3	97.73	NA
LSTM-FUZZY	99.71	99.87	0.25
Detection technique [21]	91	83	4
Detection technique [22]	NA	96	0.7
The proposed technique	99.84	99.88	0.21

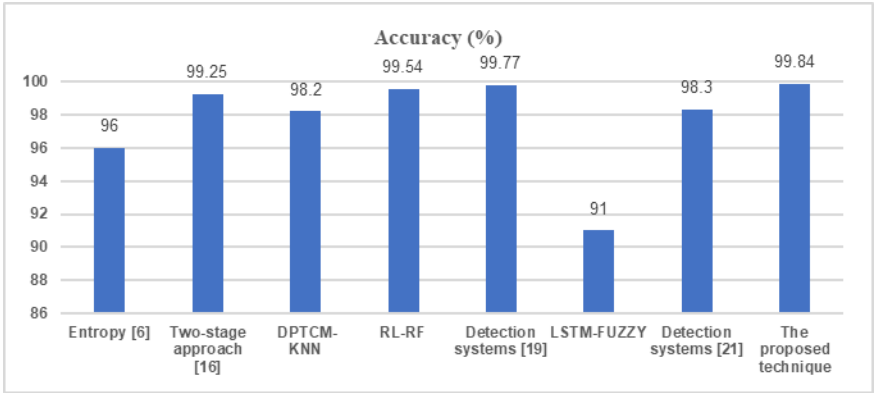

Fig.4. Accuracy comparison of different algorithms

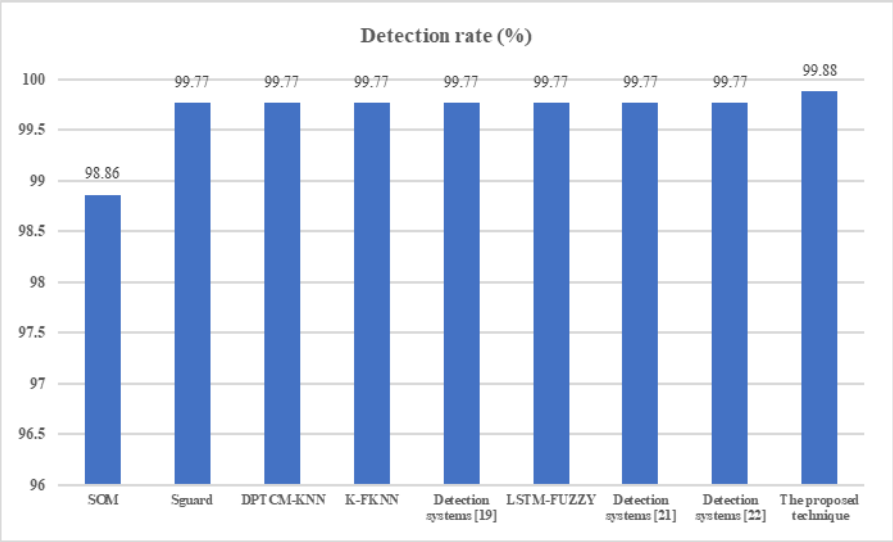

Fig.5. Detection rate comparison of different algorithms

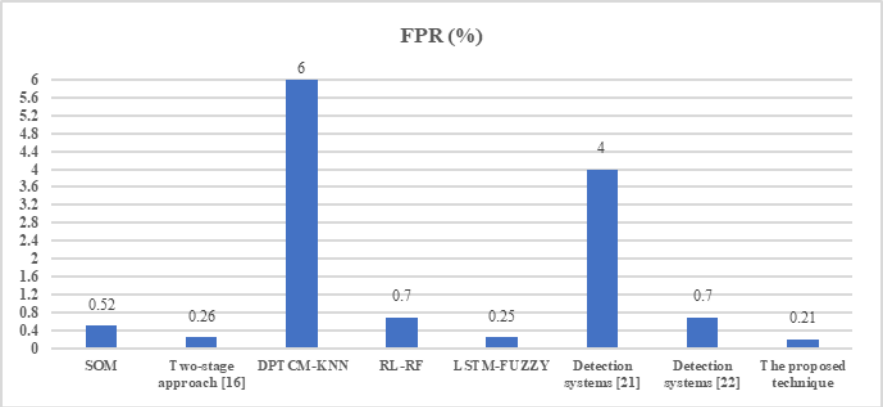

Fig.6. False-positive rate comparison of different algorithms

Table 3. and Fig. 7. show f_measure comparisons among K-FKNN, detection systems implemented in [19], LSTM-FUZZY, detection technique implemented in [21], and the proposed technique. It can be shown from the results that the proposed technique in this paper gives the best results compared with other techniques.

Table 3. F_measure comparison of different algorithms

Algorithm	F_measure (%)
K-FKNN	97.65
Detection systems in [19]	97.7
LSTM-FUZZY	99.8
Detection systems in [21]	88.1
The proposed technique	99.83

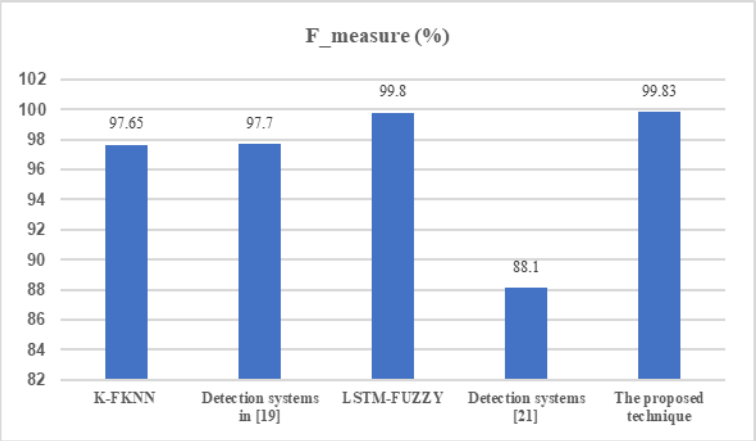

Fig.7. F_measure comparison of different algorithms

5. Conclusion and Future Work

The DDoS attack can be classified as the most severe attack that threatens the functionality and security of SDN architecture. It may target dropping the SDN controller, which is the main component in SDN infrastructure since it represents the centralized control plane of the network. In this paper, new features are used to construct an ML-based model that can detect a DDoS attack against the SDN controller. The new features are extracted from packet headers in addition to the traffic statistics. The introduced work adds to the field of SDN security since the experimental results show that the proposed work can detect the attack efficiently. The effectiveness of the introduced detection method can be justified due to the high obtained detection accuracy and low false-positive rate. Besides mitigating the effects of the attack by blocking attacking devices and logging their important information. In the future, we look forward to using ML to implement a detection technique against a DDoS attack in the data plane and its forwarding devices. It is also important to work on providing an ML-based method to detect a DDoS attack in SDN architectures which has multiple controllers in the control plane. Also, it is important to detect other types of attacks that make use of different SDN architecture vulnerabilities.

References

[1] Barabash, Oleg and Kravchenko, Yuri and Mukhin, Vadym and Kornaga, Yaroslav and Leshchenko, Olga, "Optimization of Parameters at SDN Technologie Networks., " International Journal of Intelligent Systems \& Applications, vol. 9, no. 9, 2017.
[2] A. Abdou, P. C. van Oorschot and T. Wan, "Comparative Analysis of Control Plane Security of SDN and Conventional Networks," IEEE Communications, vol. 20, pp. 3542-3559, 2018.
[3] Open Vswitch. 2018, [Online]. Available at: http://openvswitch.org
[4] J. van, L. M. van Adrichem and A. Kuipers, "Scalability and Resilience of Software-Defined Networking: An Overview," arXiv preprint, 2014.
[5] H. Zhang, Z. Cai, Q. Liu, Q. Xiao, Y. Li, and C. F. Cheang, "A Survey on Security-Aware Measurement in SDN," Security and Communication Networks, 2018.
[6] S. Mousavi and M. St-Hilaire, "Early detection of DDoS attacks against SDN controllers," International Conference on Computing, Networking and Communications, pp. 77—81, 2015.
[7] C. Douligeris and A. Mitrokotsa, "DDOS attacks and defense mechanisms: classification and state-of-the-art," Computer Networks 44, vol. 44, pp. 643-666, 2004.
[8] L. Barki, A. Shilling, N. Meti, Narayan and M. M. Mulla, "Detection of Distributed Denial of Service Attacks in Software Defined Networks, " Intl. Conference on Advances in Computing, Communications and Informatics, pp. 2576-2581, 2016.
[9] N. Z. Bawany, J. A. Shamsi, and K. Salah, "DDoS Attack Detection and Mitigation Using SDN: Methods, Practices, and Solutions," Arab J Sci Eng, vol. 42, pp. 425–441, 2017.
[10] Classification Algorithms in Machine Learning. 2018, [online]. Available at: https://medium.com/datadriveninvestor/classification-algorithms-in-machine-learning-85c0ab65ff4
[11] Seyyid Ahmed Medjahed, Mohammed Ouali, Tamazouzt Ait Saadi, Abdelkader Benyettou,"An Optimization-Based Framework for Feature Selection and Parameters Determination of SVMs", International Journal of Information Technology and Computer Science, vol.7, no.5, pp.1-9, 2015.
[12] J. Ashraf, and S. Latif, "Handling intrusion and DDoS attacks in software-defined networks using machine learning techniques, " IEEE National Software Engineering Conference, pp. 55-60, 2014.
[13] Y. XU, H. SUN, F. XIANG, AND Z. SUN, “Efficient DDoS Detection Based on K-FKNN in Software Defined Networks”, IEEE Access, vol.7, pp. 160536 - 160545, 2019.
[14] Wang, Tao and Chen, Hongchang, "SGuard: A lightweight SDN safe-guard architecture for DoS attacks," IEEE China Communications, vol. 14, pp. 113-125, 2017.

[15] Han, Tao and Jan, S. R. Ullah, Z. Tan, "A comprehensive survey of security threats and their mitigation techniques for next - generation SDN controllers," Concurrency and Computation: Practice and Experience 32, vol. 32, 2020.
[16] Latah, Majd and Toker, Levent, "A novel intelligent approach for detecting DoS flooding attacks in software-defined networks," International Journal of Advances in Intelligent Informatics, 2018.
[17] H. Peng, Z. Sun, X. Zhao, S. Tan, and Z. Sun, "A detection method for anomaly flow in software defined network," IEEE Access, vol. 6, pp. 27809 - 27817, 2018.
[18] C. Xu, Hui, Y. Wu, X. Guo, and W. Lin, "An SDNFV-Based DDoS Defense Technology for Smart Cities," IEEE Access, vol. 7, pp. 137856 - 137874, 2019.
[19] Polat, Huseyin and Polat, Onur and Cetin, Aydin, "Detecting DDoS Attacks in Software-Defined Networks Through Feature Selection Methods and Machine Learning," Sustainability, vol. 12, p.1035, 2020.
[20] Novaes, Matheus P and Carvalho, Luiz F and Lloret, Jaime and Proen{\c{c}}a, Mario Lemes. 2020. Long Short-Term Memory and Fuzzy Logic for Anomaly Detection and Mitigation in Software-Defined Network Environment. IEEE Access, vol. 8, pp. 83765--83781
[21] Ujjan, Raja Majid Ali and Pervez, Zeeshan and Dahal, Keshav and Bashir, Ali Kashif and Mumtaz, Rao and Gonz{\'a} lez, J. 2020. Towards sFlow and adaptive polling sampling for deep learning based DDoS detection in SDN. Elsevier's Future Generation Computer Systems, vol. 111, pp. 763--779
[22] Ali, Sarwan and Alvi, Maria Khalid and Faizullah, Safi and Khan, Muhammad Asad and Alshanqiti, Abdullah and Khan, Imdadullah. 2020. Detecting ddos attack on sdn due to vulnerabilities in openflow. IEEE 2019 International Conference on Advances in the Emerging Computing Technologies (AECT), pp. 1-6.
[23] 35 Types of DDoS Attacks Explained. 2018, [online]. Available at: https://javapipe.com/blog/ddos-types/
[24] The Mathematics Behind Support Vector Machine Algorithm (SVM). 2018, [online]. Available at: https://www.analyticsvidhya.com/blog/2020/10/the-mathematics-behind-svm/.
[25] Thuy Nguyen Thi Thu, Vuong Dang Xuan, "Supervised Support Vector Machine in Predicting Foreign Exchange Trading", International Journal of Intelligent Systems and Applications, Vol.10, No.9, pp.48-56, 2018.
[26] Mahmood Z. Abdullah, Nasir A. Al-awad, Fatima W. Hussein, " Evaluating and Comparing the Performance of Using Multiple Controllers in Software Defined Networks", International Journal of Modern Education and Computer Science, Vol.11, No.8, pp. 27-34, 2019.
[27] T. Koponen, J. Pettit, B. Pfaff, M. Casado, N. McKeown, S. Shenker, N. Gude, "NOX: Towards an Operating System for Networks," Computer Communication Review, vol. 38, pp. 105-110, 2008.
[28] Mininet. 2018, [Online]. Available at: http://mininet.org
[29] Scapy. 2018, [Online]. Available at: http://www.secdev.org/projects/scapy

Authors' Profiles

Waheed G. Gadallah born in Egypt on 1st Jun 1993, did his B.Sc in information technology from faculty of computers and information, Assiut university, Egypt, in 2015. He is working as Teaching Assistant at the Information Technology Department, Faculty of Computers and Information, Assiut University, Assiut, Egypt, from 2016 to now. He passed the pre-Master studies in information technology, in 2017. He has research works in progress in his research field.

Dr. Nagwa M. Omar received the B.Sc., M.Sc., and PhD degrees in Compute Engineering from the Faculty of Engineering, Assiut University, Assiut, Egypt, in 1999, 2002, and 2008 respectively. She worked as Assistant Professor at the Information Technology Department, Faculty of Computers and Information, Assiut University, Assiut, Egypt, from 2009 to 2016. She is working as Associate Professor at the Information Technology Department, Faculty of Computers and Information, Assiut University, Assiut, Egypt from 2016 to now.

Prof. Hosny M. Ibrahim received the B.Sc., and M.Sc. degrees in Electrical Engineering from the Faculty of Engineering, Assiut University, Assiut, Egypt, in 1973, and 1977 respectively. He received the Ph.D. degree in Electrical Engineering from Iowa State University, Ames, Iowa, U.S.A. in 1982. He was the Dean of the Faculty of Computers and Information, Assiut University, Assiut, Egypt from September 2002 to August 2011. He was the head of the Information Technology Department, Faculty of Computers and Information, Assiut University, Assiut, Egypt from July 2010 to May 2015. He is currently Professor at the Information Technology Department, Faculty of Computers and Information, Assiut University, Assiut, Egypt.

How to cite this paper: Waheed G. Gadallah, Nagwa M. Omar, Hosny M. Ibrahim, "Machine Learning-based Distributed Denial of Service Attacks Detection Technique using New Features in Software-defined Networks", International Journal of Computer Network and Information Security(IJCNIS), Vol.13, No.3, pp.15-27, 2021. DOI: 10.5815/ijcnis.2021.03.02