Hindawi
Mobile Information Systems
Volume 2022, Article ID 1169035, 13 pages
https://doi.org/10.1155/2022/1169035

*Research Article*

# An Efficient Method for Online Detection of DRDoS Attacks on UDP-Based Services in SDN Using Machine Learning Algorithms

**Mitra Akbari Kohnehshahri, Reza Mohammadi, Hatam Abdoli, and Mohammad Nassiri**

*Computer Department, Engineering Faculty, Bu-Ali Sina University, Hamedan 6517838695, Iran*

Correspondence should be addressed to Hatam Abdoli; abdoli@basu.ac.ir

Received 4 February 2022; Revised 5 April 2022; Accepted 6 May 2022; Published 26 May 2022

Academic Editor: Jianhui Lv



With advances in mobile devices and systems and the emergence of new ideas such as cloud computing and big data, as well as the tremendous growth in the number of network users, the need to modify the current network architectures has been very much in the foreground in recent years. One of the promising solutions to overcome these challenges is software-defined networking (SDN). SDN is a unique innovative architecture in which network control and traffic flows are independent of each other and planned directly. The SDN's focused view of networks is more comprehensive than other methods, which is why SDN is more efficient in coping with malicious attacks including amplification attacks. The response to amplification of distributed denial of service (DDoS) attacks is larger than the request. In an amplification attack, the attacker fakes the victim's address as the source address and the responses are forwarded to the victim instead of the attacker. This is why these attacks are more difficult to discover in traditional networks, while the focused method of SDN can contribute to the detection of such attacks. There are different methods for detecting these attacks, one of which is to use machine learning (ML) algorithms. In line with this, the present paper is aimed at the detection of distributed reflection denial of service (DRDoS) attacks using ML algorithms. Simulation was performed by the use of ML algorithms, and the findings suggest a significant improvement in the detection of DRDoS attacks in comparison with previous methods.

## 1. Introduction

Growing innovation in network applications and reduction in the costs of network operators has resulted in the idea of software-defined networking (SDN). In this model, the network becomes more intelligent and controllable, which facilitates the innovation and management of the network. The main difference with conventional networks is the separation of the control plane from the network devices. Separating these two planes, a centralized control becomes possible, which offers a comprehensive view of the network. Such a view of SDN enables network administrators to control and manage a large number of network devices, network topology, security policies, and routing automatically and dynamically by means of high-level languages [1]. The advantages of separating data and control units in SDN are as follows.

Centralized control allows for integrated management and control by a central controller. Central control is a logical concept, and the control unit can be implemented in a distributed manner [2, 3]. Network devices are managed and controlled in an integrated and centralized manner, which can increase the productivity of network resources [4].

Dynamicity of the configuration helps to easily add new features to the network infrastructure without updating all network devices [2, 3].

Due to the integrated network management in SDN, the network performance is optimized. By using information about the current status of the existing resources, we can utilize methods that make optimum use of network resources in order to improve network efficiency [5].

Workload is balanced. The network controller can be set in a way that, based on the current status of the network, it

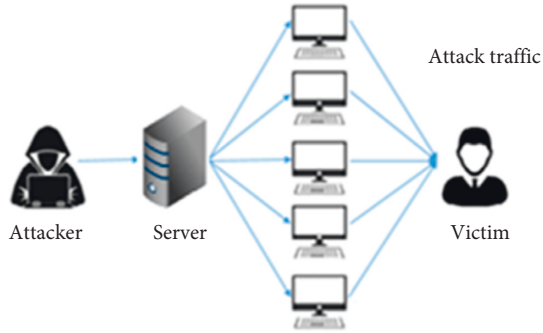


Figure 1: A DDoS attack.

makes the data unit forward the traffic in a balanced manner to prevent congestion on the network and make optimum use of the resources [2, 3].

The required parameters are provided. The users of a network may need certain parameters of quality of service based on the traffic they produce. Thus, to reach an acceptable level of productivity among users, we can develop a program that performs this operation on the network [6].

Error tolerance is increased. By monitoring the network equipment, the controller can detect errors and determine alternative routes if needed. Furthermore, by calculating and announcing alternative routes, the traffic can be instantly forwarded to these routes in the event of error occurrence [2, 3].

Another advantage is that the complexity is decreased. As SDNs are designed at the level of software and do not depend on hardware manufacturers, innovative ideas can be easily implemented in the network [4]. With the increasing growth in networks, security has turned into a major challenge for organizations, governments, and important persons. One type of security threat is denial of service (DoS) attacks which are operated through a single source. The aim is to block the access of authorized users to a certain network resource or make a victim unavailable. Thus, the server can no longer respond to the authorized users. Performing successful DoS attacks on today's powerful systems is not possible through a single system. Moreover, an attack from multiple sources is far more difficult to trace than an attack from a single system. In distributed denial of service (DDoS) attacks, the attacker floods the target with millions of requests per second, which renders the host incapable of offering services to the authorized users. The attackers usually exploit the existing vulnerabilities to attack the target. When the favorable conditions for the attack exist, the attacker can perform a large integrated attack on a target. Ordinary defense mechanisms can easily neutralize single-source attacks. One of the aims of the research in the field of detecting DDoS intrusions has been to devise a comprehensive defense mechanism against such attacks.

DDoS attacks are one of the main security challenges which constitute the largest part of all security threats. Figure 1 illustrates the general structure of a DDoS attack. A DDoS attack attempts to disrupt the network services for various purposes. It uses public services as reflectors in order to increase network traffic [7, 8]. As mentioned regarding distributed reflection denial of service (DRDoS) attacks, the attackers use amplification attacks to intensify the effects of the attack. Therefore, the notion of amplification factor (AF) was introduced to measure the effect of DRDoS attacks. AF is divided into two types: packet amplification factor (PAF) and bandwidth amplification factor (BAF) which are defined by equations (1) and (2), respectively.

$$\mathrm{BAF} = \frac{\mathrm{Len\,(udp\ payload)\ reflector\ to\ victim}}{\mathrm{Len\,(udp\ payload)attacker\ to\ reflector}}, \quad (1)$$

$$\mathrm{PAF} = \frac{\mathrm{Number\ of\ packet\ \ reflector\ to\ victim}}{\mathrm{Number\ of\ packet\ attacker\ to\ reflector}}. \quad (2)$$

BAF is the volume in bytes of the data sent from the amplifier to the victim divided by the volume into bytes of the data sent from the attacker to the amplifier. PAF is the number of packets sent from the amplifier to the attacker divided by the number of packets sent from the attacker to the amplifier. Based on the protocol used, amplification DDoS attacks can be classified as TCP-based or UDP-based attacks. The AF of some of these attacks is listed in Table 1 [9].

As can be seen in Table 1, there are several protocols that cause high values of BAF in UDP-based attacks. For this reason, UDP is the main protocol of DRDoS attacks. As shown in Figure 2, the response produced by these attacks has a greater size than the request. This type of attack fakes the source IP and makes it difficult to filter troublesome packets [7]. The attacker uses UDP-based protocols to disrupt the networks and servers related to network services. In this type of attack, a server which is used as a reflector produces a large amount of data in response to the requests. This server hides the attacker's identity from the victim [10, 11].

These attacks can be detected using SDN technology which adopts a comprehensive, integrated method to networks.

Machine learning (ML) techniques are widely used in intrusion detection systems. Thus, these techniques are effectively used in the detection of DDoS attacks in SDN. In general, ML techniques distinguish between normal and destructive traffic flows according to certain features of the traffic [12].

Table 1: The amplification factor for UDP-based protocol [10].

| Protocol | BAF | PAF | Scenario | Description | Port (s) |
|---|---|---|---|---|---|
| SNMP v2 | 6.3 | 1.00 | GetBulk request | Monitoring network-attached devices | 161 |
| NTP | 556.9 | 3.84 | Request client statistics | Time synchronization | 123 |
| DNS | 54.6 | 2.08 | ANY lookup at author | Domain name resolution | 53 |
| NetBIOS | 3.8 | 1.00 | Name resolution | Name service protocol of NetBIOS API | 137 |
| SSDP | 30.8 | 9.92 | SEARCH request | Discovery of UPnP-enabled hosts | 1900 |
| CharGen | 358.8 | 1.00 | Character generation request | Legacy character generation protocol | 19 |
| QOTD | 140.3 | 1.00 | Quote request | Legacy "quote-of-the-day" protocol | 17 |
| BitTorrent | 3.8 | 1.58 | File search | BitTorrent's Kademlia DHT impl. | Any |
| Kad | 16.3 | 1.00 | Peer list exchange | eMule's Kademlia DHT impl. | Any |
| Quake 3 | 63.9 | 1.01 | Server info exchange | Games using the Quake 3 engine | 27960 |
| Steam | 5.5 | 1.12 | Server info exchange | Games using the steam protocol | 27015 |
| ZAv2 | 36.0 | 1.02 | Peer list and cmd exchange | P2P-based rootkit | 164XY |
| Sality | 37.3 | 1.00 | URL list exchange | P2P-based malware dropper | Any |
| Gameover | 45.4 | 5.39 | Peer and proxy exchange | P2P-based banking Trojan | Any |

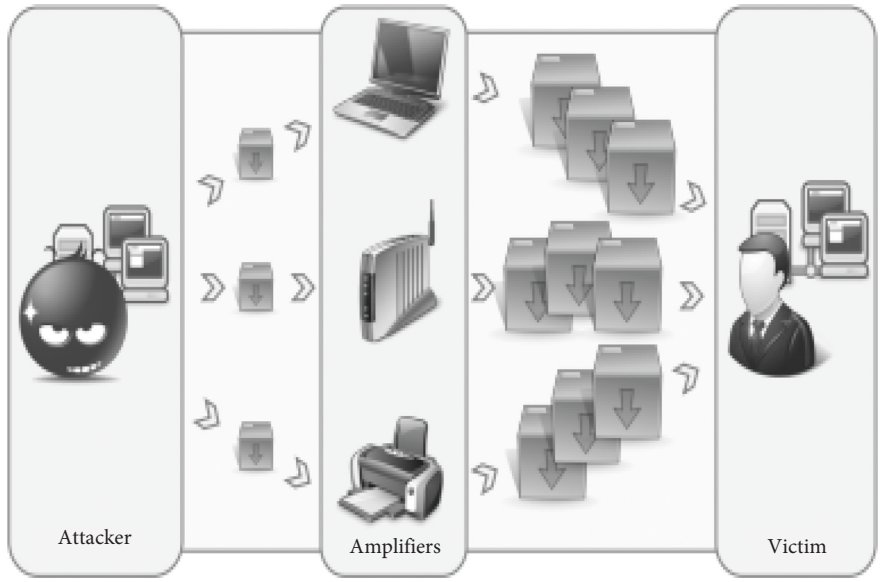


Figure 2: A DRDoS attack [10].

ML methods use techniques for detecting anomalies in a network which may be based on models, statistical and mathematical computations, unsupervised ML, or supervised ML. In fact, any system designed for the detection of network anomalies does this task by collecting the traffic and extracting some kind of information from it. The ML method tries to make a distinction between normal and abnormal traffic patterns [13]. Characteristic of ML techniques is that data are easily available and progress can be seen quite early. They are also useful for solving problems in the functionality and management of the network. The ML techniques for solving fundamental problems in the network include traffic prediction, routing and classification, density control, resource and error management, and network security [14]. As a whole, ML can improve the network's performance through experience. The aim of ML is to design computer systems that learn by experience and are able to adapt to the surrounding environment [15]. The basic procedures of all ML systems are similar. First, the algorithm is provided with training data. The algorithm is actually responsible for learning and looking for different patterns in the data. After finding the patterns, the algorithm devises a model that can be stored in the memory. Afterward, the system can use the models to predict the behavior [16]. As shown in Figure 3, ML algorithms can be classified as supervised learning, unsupervised learning, and reinforcement learning.

To this end, the present paper is aimed at online detection of amplification DDoS attacks in an SDN using ML algorithms.

The paper is organized as follows. In the first section, we shall explain some basic concepts including SDN, attacks, DRDoS attacks, and ML algorithms. The second section reviews the previous works regarding the attacks performed

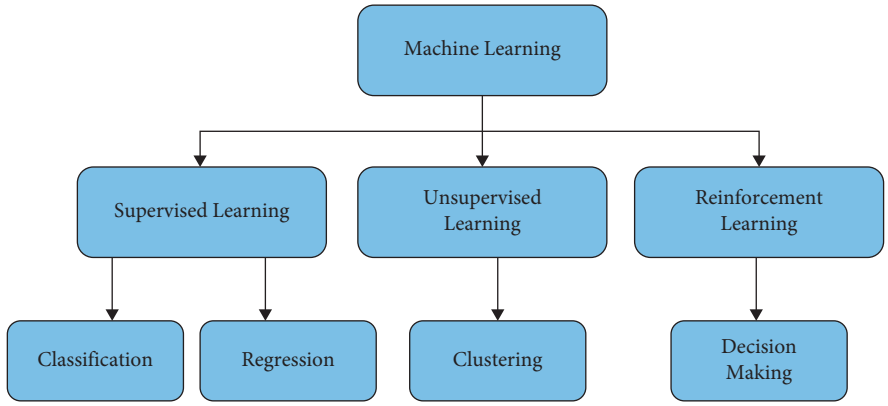


Figure 3: Typology of machine learning.

on SDN. Section 3 introduces the proposed method for online detection of amplification DDoS attacks using ML algorithms. Finally, Section 4 describes the implementation of the proposed method and compares its model with the other methods. The final section makes some conclusions.

## 2. Related Works

The emergence of SDN has caused novel ideas in this field. After the introduction of SDN, researchers began to investigate the capabilities of these networks. Detection of attacks was one of the key topics that received much attention.

As mentioned in Section 1, DDoS attacks are among the major challenges and vulnerabilities that this young network architecture is facing. Amplification attacks are one of the most serious and frequent DDoS attacks. To obtain an in-depth knowledge of the problem, we reviewed the previously proposed method for defending against DDoS attacks, particularly amplification attacks. Previous works in this field have studied some types of attack through the lens of ML algorithms. We will compare them based on whether they were conducted in the context of traditional networks or SDN.

Santos et al. [13] evaluated four different ML algorithms with the aim of examining the precision and speed of classification of attacks and normal traffic in the vulnerable part of SDNs including controller, flow table, and the bandwidth between switch and controller. The algorithms used in this study include support vector machine (SVM), neural network (NN), decision tree (DTree), and random forest (RF). The emulator used was Mininet, and the controller was POX. In these emulations, the RF algorithm was the most precise ML algorithm in terms of their focus on the vulnerable points of the SDN, followed with a small difference by the DTree algorithm. The shortest processing time belongs to the DTree algorithm. Rahman et al. [17] compared four ML algorithms, namely, J48, RF, SVM, and K-nearest neighbor (K-NN). J48 had the performance in terms of training time and testing time.

The paper used hping3 tool in Python to produce normal traffic and simulate ICMP and TCP flood DDoS attacks. Chen et al. [18] studied the detection of amplification DDoS attacks using ML algorithms in SDN. This paper examined the precision of detection of the SVM algorithm in different time intervals.

Gao et al. [9] conducted research into the detection of amplification DDoS attacks in traditional networks. This study focused on DNS protocols. It sought to detect amplification DDoS attacks using PAF and BAF values. Filho et al. [19] addressed the online detection of DoS and DDoS attacks. They evaluated different datasets and finally proposed CIC-DoS, CICIDS2017, and CSE–CIC–IDS2018 customized datasets. The datasets of different attacks were generated using several tools, i.e., hping3, GoldenEye, hulk, and slowhttptest. This study obtained different precision values with the ML algorithms including RF, DTree, logistic regression (LR), stochastic gradient descent (SGD), perceptron, and AdaBoost with different numbers of features.

Another study by Nanda et al. [20] addressed four ML algorithms, namely, BayesNet, C4.5, NaiveBayes, and DTree. The average prediction precision for BayesNet was found to be 91.68. The aim of this paper was to evaluate the precision and speed of processing. Irom Meite et al. [21] sought to detect amplification DNS attacks in traditional networks using four ML algorithms including DTree, multilayer perceptron (MLP), Naïve Bayes, and SVM. The detection precision of the DTree algorithm was greater than the other algorithms.

## 3. The Proposed Method

As can be seen in Table 2, the mentioned methods do not provide all the requirements for efficient online detection of amplification DDoS attacks in SDN. For this reason, we decided to incorporate all these items into our proposed method. The importance of amplification DDoS attacks was described above. Our methodology is aimed at online detection of amplification DNS attacks. For implementation, we prefer SDN due to their integrated perspective which consists of data plane and control plane. In the data unit, we

Table 2: A summary of the previous studies.

| Source | Online | Offline | SDN-based | DNS amplification |
|---|---|---|---|---|
| [13] | ✗ | ✓ | ✓ | ✗ |
| [17] | ✗ | ✓ | ✓ | ✗ |
| [18] | ✗ | ✓ | ✓ | ✓ |
| [9] | ✗ | ✓ | ✗ | ✓ |
| [19] | ✓ | ✗ | ✗ | ✗ |
| [20] | ✗ | ✓ | ✓ | ✗ |
| [21] | ✗ | ✓ | ✗ | ✓ |
| Proposed method | ✓ | ✗ | ✓ | ✓ |

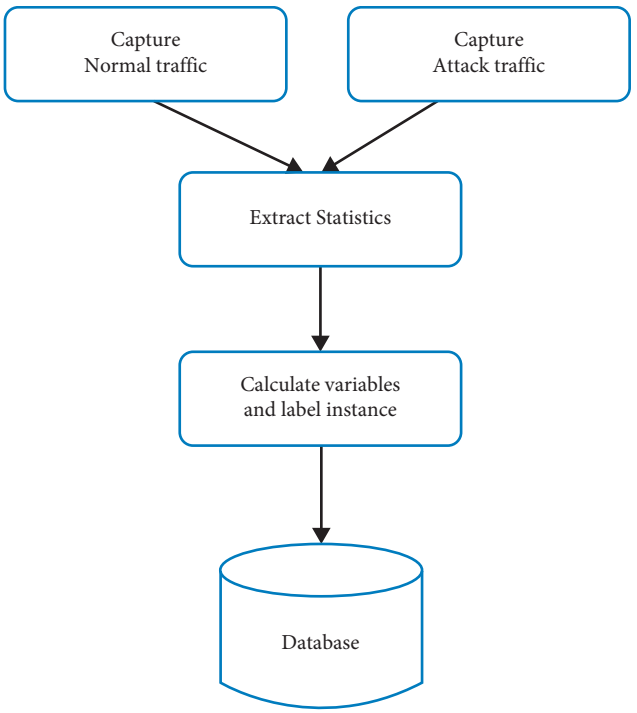


Figure 4: Combination of normal and malicious DNS traffic.

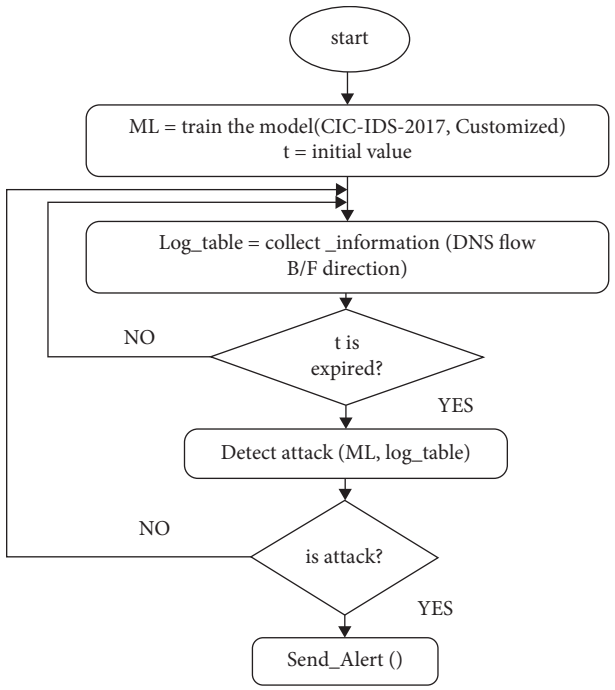


Figure 5: Flowchart of the proposed method.

assume a topology for generating online traffic. The network requires a controller which must be managed by the network on the data plane. On the control plane of an SDN, statistical information about the network links is collected by switches and transmitted to the controller.

In normal DNS traffic conditions, the volume of responses is more than the volume of requests, which is opposed to the attacking conditions where the volume of requests multiplies. Next, the appropriate ML algorithms are selected. The algorithms are trained based on the existing dataset, and then, the model is developed. Finally, online traffic is produced and the controller collects the information about the traffic in intervals of 20 seconds. The ML algorithms will make use of the developed model to distinguish malicious traffic from normal traffic.

As mentioned earlier, the aim of this paper is to detect amplification DNS attacks with the help of the controller's integrated perspective as well as ML algorithms which are modeled based on several attributes that have been extracted from the dataset according to their importance. Our proposed method makes use of five ML algorithms including DTree, RF, SVM, gradient boosting classifier (GBC), and AdaBoost. In this study, the proposed dataset for the detection of DDoS attacks was CICIDS2017 [22] for normal DNS traffic. For malicious traffic, amplification DNS attacks were produced using Scapy tool [23]. CIS-IDS2017 consists of several sets of different types of attack traffic and normal traffic. As can be seen in Figure 4, due to the existence of normal DNS traffic in this dataset, we only extracted the normal traffic from this dataset and combined it with the generated malicious traffic.

The majority of amplification DNS attacks can be detected through the volume and number of response packets. These two attributes were extracted using different algorithms. Figure 5 shows how the proposed method works. First, the ML algorithms are trained in an offline manner using CIS-IDS-2017 as well as the generated malicious dataset. In the determined timespan, statistical information about the DNS response and request flows is collected. When the time ends, attacks can be detected via the trained

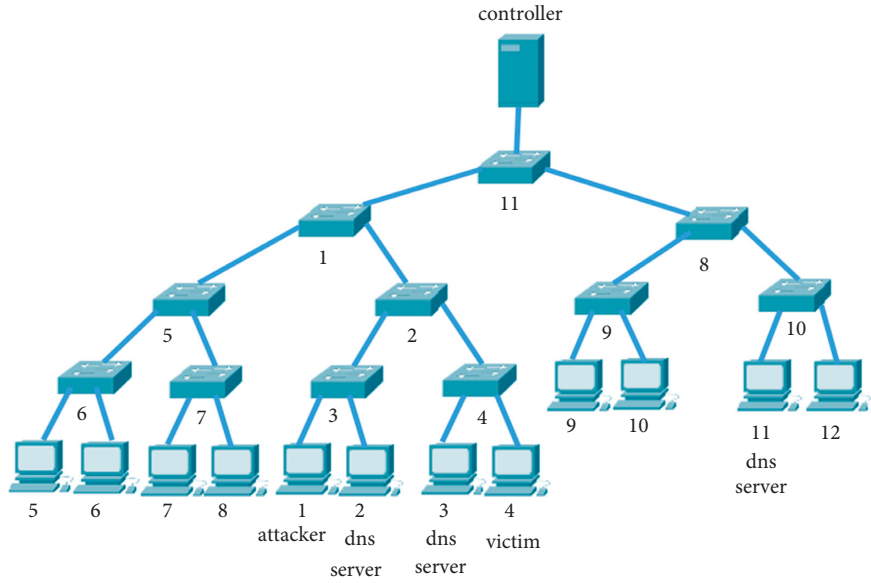


Figure 6: Topology of the online attack traffic.

Table 3: Simulation settings.

| Environment | SDN |
|---|---|
| Emulator | Mininet |
| Controller | Ryu |
| ML algorithms | AdaBoost, DTree, RF, SVM, GBC |
| Number of attributes selected from the dataset | 12 |
| Number of switches in the topology | 11 |
| Traffic type | Normal DNS traffic, malicious DNS traffic, and iperf traffic |
| Intervals of preparing reports on the network | 20 seconds |
| Type of attack detection | Online (simultaneously with traffic generation) |

ML algorithms and the statistical information about the flows. On detecting an attack, an alert is declared and the system returns to the information collection step.

## 4. Performance Evaluation

We implemented the control unit using Ryu controller [24], which is a tool written in Python and is well known among academic researchers. The data unit was simulated using Mininet [25].

As shown in Figure 6, the first step in designing the simulation scenario is to consider a system composed of 11 switches and 12 hosts. To investigate the performance of the proposed method in different conditions, we consider three different modes for the number of attackers and victims in the topology. With these different conditions, we can do more experiments and examine more results. In the first scenario, the topology consists of an attacker and one victim, in the second scenario the topology consists of two attackers and two victims, and in the last scenario the topology consists of three attackers and two victims. In all three scenarios, the number of DNS servers is three. The traffic generated in the topology includes amplification DNS attack, normal DNS traffic, and iperf traffic.

Table 4: Selected features in the proposed method. The average importance of each attribute in the proposed method as obtained by the learning algorithms is presented in Figure 7.

| Number | Features |
|---|---|
| 1 | Total requested packets |
| 2 | Total response packets |
| 3 | Total volume of requested packets |
| 4 | Total volume of response packets |
| 5 | Maximum volume of response packet |
| 6 | Minimum volume of response packet |
| 7 | Average volume of response packets |
| 8 | Standard deviation of the volume of response packets |
| 9 | Minimum packet volume |
| 10 | Maximum packet volume |
| 11 | Average packet volume |
| 12 | Standard deviation of the volume of packets |

Table 3 provides a summary of the implementation conditions of the proposed method. In the first scenario, Host 1 is the attacker and Hosts 2, 3, and 11 are DNS servers. Host 4 is the victim.

As aforementioned, on the control plane of an SDN, statistical information about the network links is collected by switches and transmitted to the controller. In the proposed

method, this information includes source IP address, destination IP address, source port address, destination port address, the number of packets in a flow, and the volume of packets in a flow. As mentioned above, one of the most important criteria for the detection of amplification DDoS attacks is difference in the size and number of packets in request and response flows. To allow for a correct detection, therefore, we have separated the request and response flows of normal and malicious traffic.

To create an amplification DDoS attack, Scapy is used for creating an amplification DNS attack.

After creating the dataset, we reduce the number of attributes down to 12 by examining the existing attributes and evaluating their importance through the ML algorithms. The final attributes are listed in Table 4.

As shown in Figure 8, the results were evaluated using an ROC curve. This graph was used to specify which ML algorithms had provided a correct classification for the dataset of the proposed method. The horizontal axis of the curve represents false-positive rate (FPR), and the vertical axis shows true-positive rate (TPR). The algorithm with the largest area under the curve has the best classification capability.

As can be seen in Figure 8, in the proposed method with the conditions of the first scenario, the SVM and RF algorithms had a better performance than the other algorithms, while the decision tree algorithm was found to have the weakest classification capability. We shall now compare our results with the two papers discussed in the review of the literature section. The first paper is [18]. The implementation method of this paper was offline, and it was implemented in an SDN. It was implemented solely by a support vector algorithm, and its results are referred to as Chen in the graphs. The second study is [9], which was implemented offline in traditional networks and whose results are specified as Gao in the graphs. Both papers are aimed at the detection of amplification DDoS attacks. The first criterion for comparing the three methods is accuracy. Figure 9 compares the accuracy level of the three methods. The horizontal axis shows the algorithms, and the vertical axis shows the accuracy level ranging from 0 to 1. Accuracy can be calculated via equation (3). True positive (TP) denotes the traffic which is an attack and has been detected correctly; true negative (TN) denotes traffic which is normal and has been detected correctly; false positive (FP) denotes normal traffic which has been wrongly specified as malicious; false negative (FN) denotes malicious traffic which has been wrongly specified as normal.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}. \tag{3}$$

As can be seen from the graph, the proposed method with the conditions of the first scenario has the highest value for the GBC and SVM algorithms. Overall, it is more accurate than Chen and Gao methods in all algorithms. The next graph compares recall rates. As in the accuracy graph, the horizontal axis represents the ML algorithms, and the vertical axis shows the recall rate. Recall rate can be calculated as follows:

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}. \tag{4}$$

As can be seen in Figure 10, the recall rate of the proposed approach with the conditions of the first scenario and the Chen method is the highest for the GBC, RF, DTree, and SVM algorithms, whereas the Yuxuan method has the lowest recall rate for the GBC algorithm. The next criterion for comparison is precision. It refers to the quantitative ratio of the items correctly classified by the ML algorithms to the total number of classified items (whether correctly or incorrectly) by the algorithm. It is calculated as follows:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}. \tag{5}$$

As in the previous graphs, the horizontal axis represents the ML algorithms and the vertical axis shows precision. Figure 11 compares the precision level of the three methods.

As can be seen in Figure 11, the highest precision in the proposed method with the conditions of the first scenario belongs to the GBC and SVM algorithms and the highest precision in the Chen method belongs to the GBC and RF algorithm. The lowest precision belongs to the DTree algorithm in the Yuxuan method. The next comparison

Table 5: FPR values.

| Algorithm | Proposed method | Chen | Gao |
|---|---|---|---|
| GBC | 0 | 0.0625 | 0 |
| RF | 0.02 | 0.05 | 0 |
| DTree | 0.06 | 0.08 | 0.12 |
| SVM | 0 | 0.02 | 0.05 |
| AdaBoost | 0.08 | 0.06 | 0.02 |

Table 6: TNR values (first scenario).

| Algorithm | Proposed method | Chen | Gao |
|---|---|---|---|
| GBC | 1 | 0.937 | 1 |
| RF | 0.977 | 0.944 | 1 |
| DTree | 0.937 | 0.914 | 0.872 |
| SVM | 1 | 0.979 | 0.942 |
| AdaBoost | 0.916 | 0.936 | 0.979 |

Table 7: FNR values (first scenario).

| Algorithm | Proposed method | Chen | Gao |
|---|---|---|---|
| GBC | 0 | 0 | 0.03 |
| RF | 0 | 0 | 0.16 |
| DTree | 0 | 0 | 0.06 |
| SVM | 0 | 0 | 0 |
| AdaBoost | 0 | 0.06 | 0.16 |

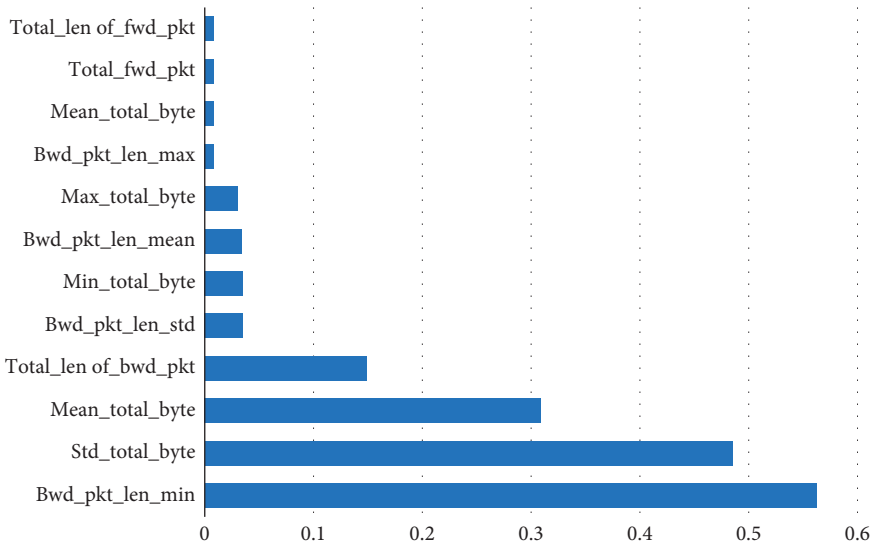


Figure 7: The importance of the selected features as obtained by the ML algorithms.

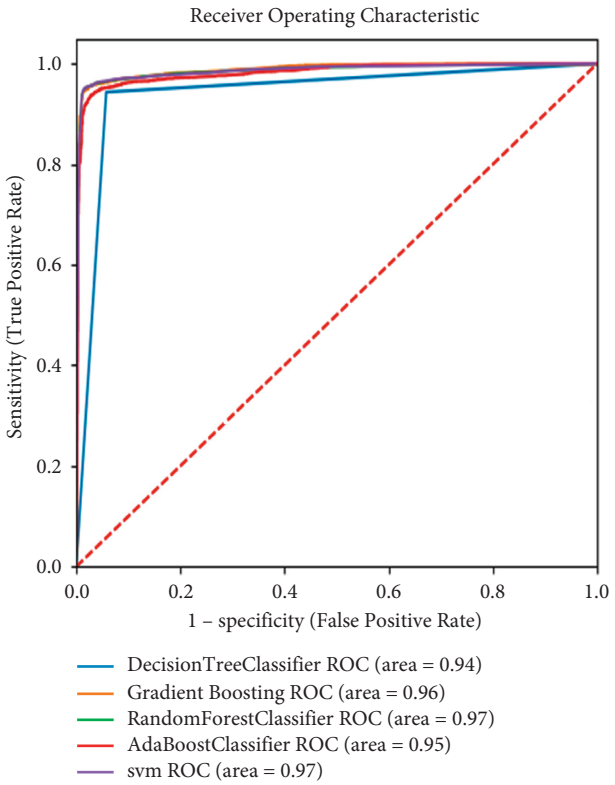


Figure 8: ROC curve of different algorithms.

criterion is F-score. On this graph, the horizontal axis lists the ML algorithms and the vertical axis shows the obtained F-scores. Figure 12 compares the F-scores of the three methods, which can be calculated via equation (6). This equation combines equations (4) and (5).

$$\text{F} - \text{score} = 2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}. \quad (6)$$

As shown in Figure 12, all ML algorithms in the proposed method with the conditions of the first scenario have gained better F-scores than the other two methods. False-positive rate (FPR) is obtained by dividing the number of normal items that were detected as malicious by the total number of normal items that were detected as malicious and normal items that were detected correctly as normal. Equation (7) calculates the value of FPR.

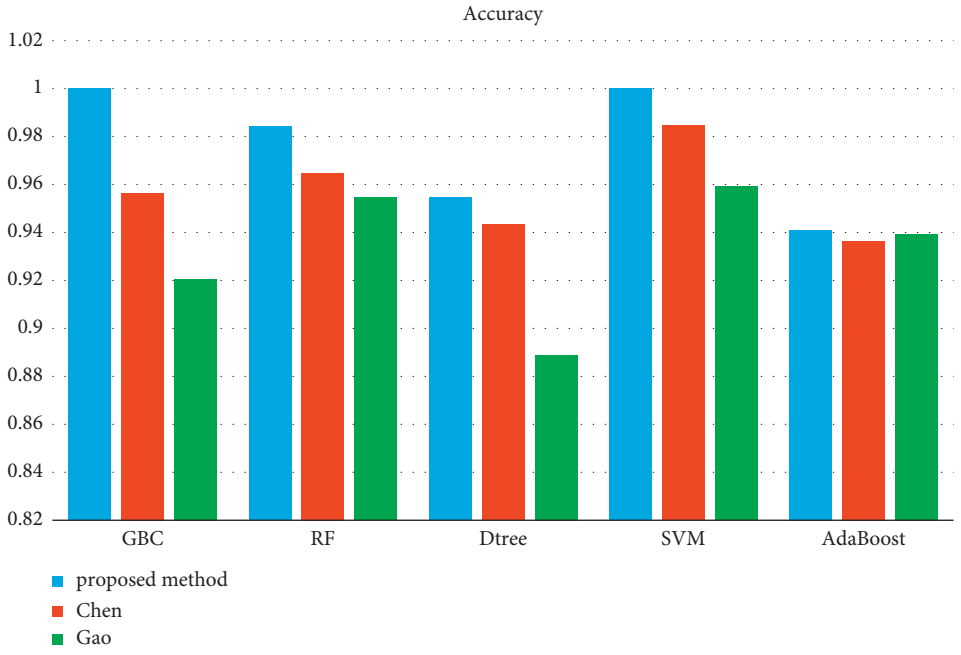


Figure 9: Accuracy for different ML algorithms (first scenario).

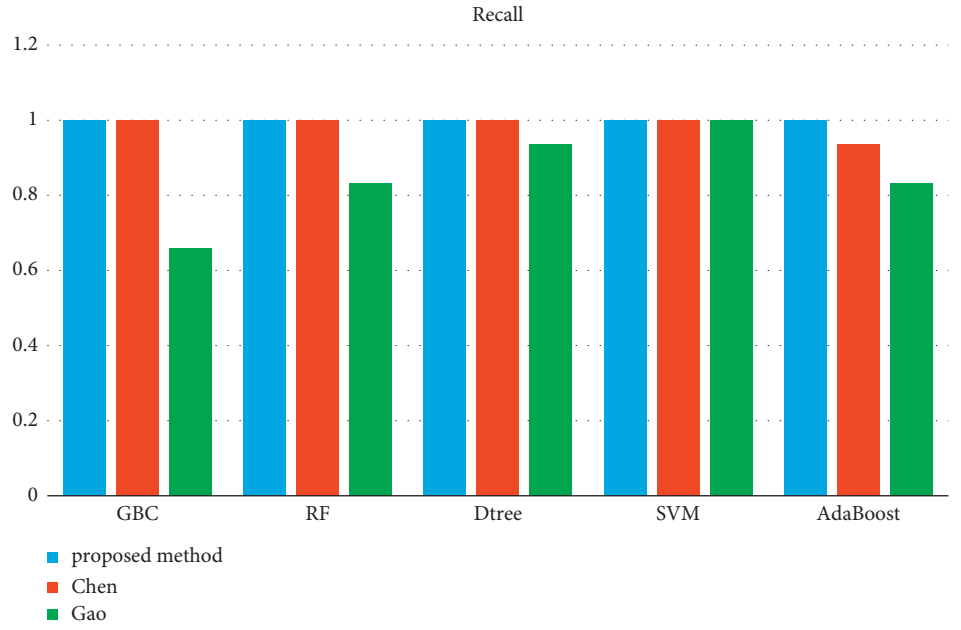


Figure 10: Recall for different ML algorithms (first scenario).

$$\text{FPR} = \frac{\text{FP}}{\text{FP} + \text{TN}}. \tag{7}$$

As shown in Table 5, the FPR of the proposed method with the conditions of the first scenario is less than that of the other methods. This value is smallest for the GBC and SVM algorithms in the proposed method. True-negative rate (TNR), which is also known as sensitivity, is obtained by dividing the number of normal items that were detected correctly by the total number of normal items that were

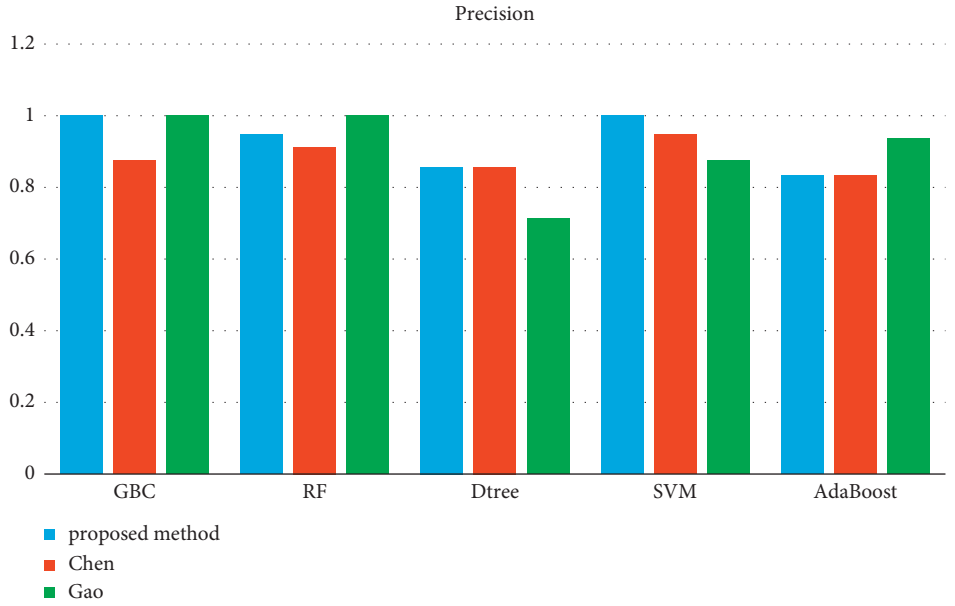


Figure 11: Precision for different ML algorithms (first scenario).

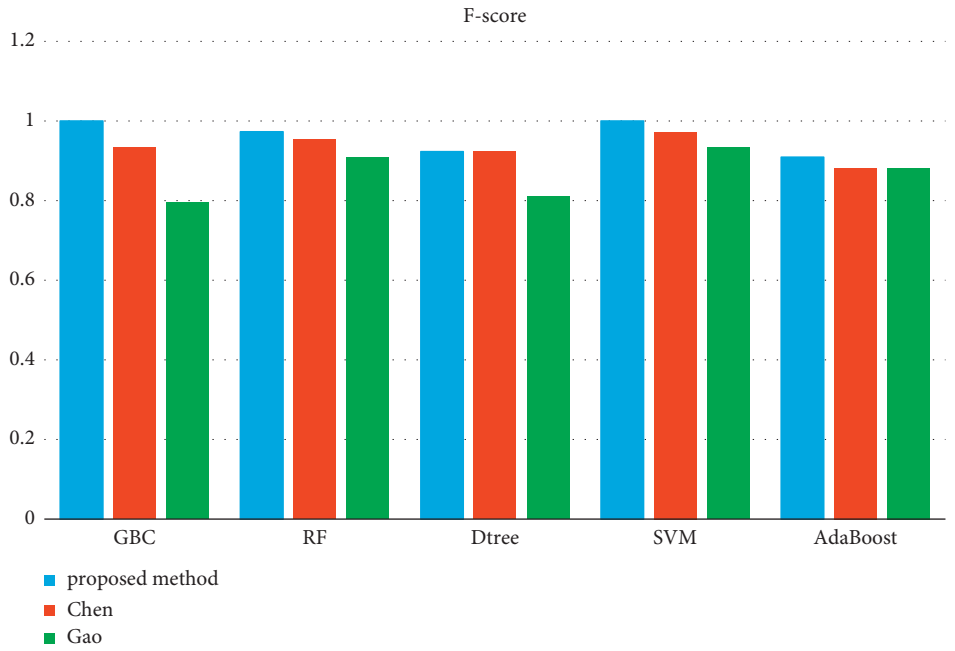


Figure 12: F-score for different ML algorithms (first scenario).

detected correctly and the normal items that were detected as malicious. The following equation is used to calculate TNR as follows:

$$\text{TNR} = \frac{\text{TN}}{\text{FP} + \text{TN}}. \tag{8}$$

As it can be seen in Table 6, the TNR of most of the ML algorithms in the proposed method with the conditions of the first scenario is greater than in the other methods. False-negative rate (FNR) refers to the ratio of the number of items that were wrongly detected as normal to the total number of items that were wrongly detected as normal and malicious items that were detected as normal. This value can be calculated by

$$\text{FNR} = \frac{\text{FN}}{\text{FN} + \text{TP}}. \tag{9}$$

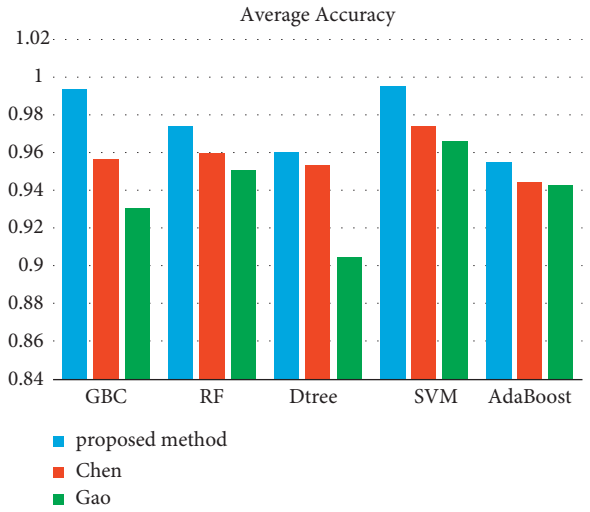


Figure 13: Average accuracy of three scenarios.

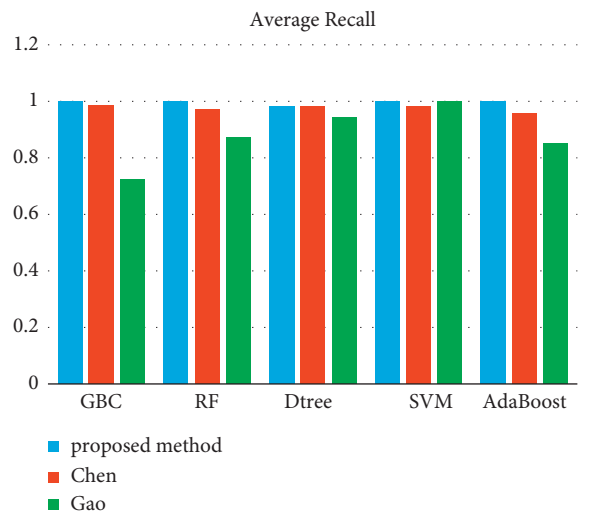


Figure 14: Average recall of three scenarios.

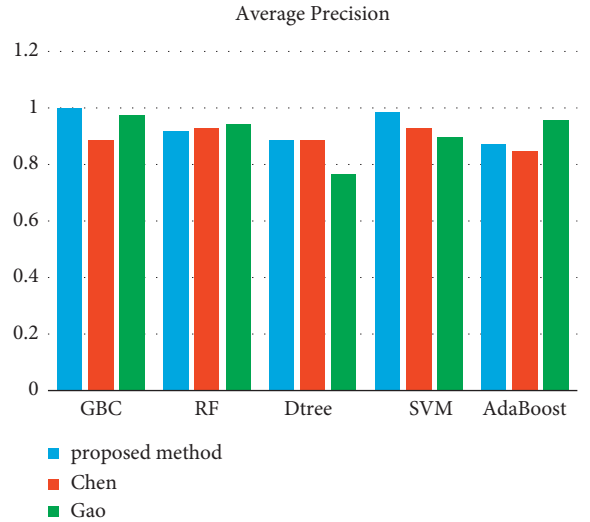


Figure 15: Average F-score of three scenarios.

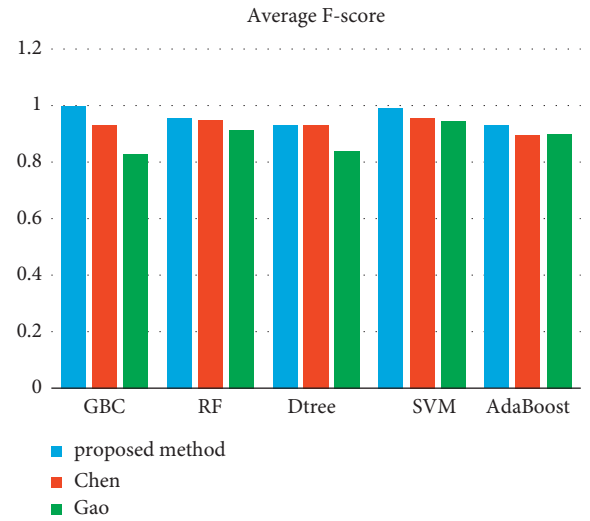


Figure 16: Average F-score of three scenarios.

Table 8: Average of FPR values.

| Algorithm | Proposed method | Chen | Gao |
|---|---|---|---|
| GBC | 0.008 | 0.055 | 0.007 |
| RF | 0.053 | 0.044 | 0.02 |
| DTree | 0.047 | 0.057 | 0.104 |
| SVM | 0.006 | 0.028 | 0.044 |
| AdaBoost | 0.063 | 0.057 | 0.014 |

Table 9: Average of TNR values.

| Algorithm | Proposed method | Chen | Gao |
|---|---|---|---|
| GBC | 0.991 | 0.944 | 0.992 |
| RF | 0.963 | 0.953 | 0.98 |
| DTree | 0.951 | 0.938 | 0.12 |
| SVM | 0.993 | 0.971 | 0.952 |
| AdaBoost | 0.950 | 0.940 | 0.984 |

Table 10: Average of FNR values.

| Algorithm | Proposed method | Chen | Gao |
|---|---|---|---|
| GBC | 0 | 0.015 | 0.17 |
| RF | 0 | 0.028 | 0.125 |
| DTree | 0.018 | 0.017 | 0.076 |
| SVM | 0 | 0.017 | 0.04 |
| AdaBoost | 0 | 0.04 | 0.014 |

As shown in Table 7, all ML algorithms used in the proposed method with the conditions of the first scenario have the lowest FNR.

We also tested the criteria obtained in the first scenario for the second and third scenarios. The results of the second and the third scenarios were close to the first scenario test. The average case results of all three scenarios are displayed in the form of graphs and tables.

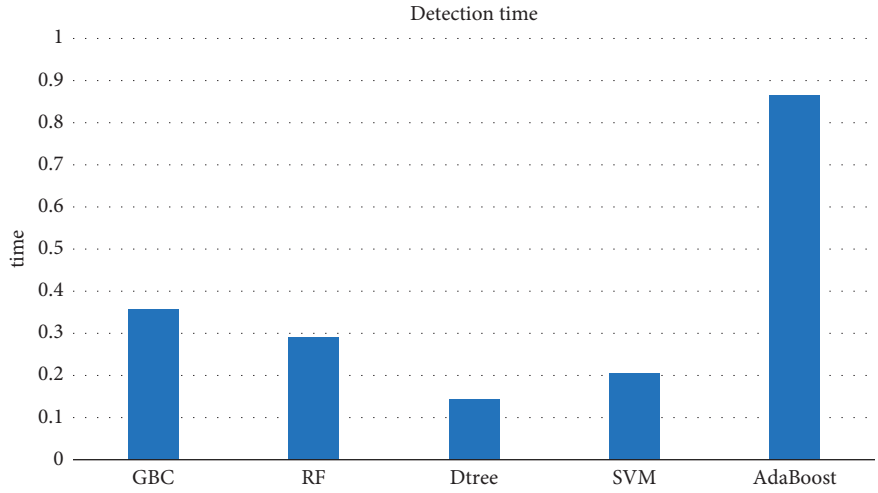


Figure 17: Speed of detection for different ML algorithms.

According to the graphs, the averages obtained in Figures 13–16 are close to the values of the first scenario. In the arrangements made, the values obtained for the first scenario, the second scenario, and the third scenario are approximately equal. According to the mean graphs, it can be concluded that the proposed method has worked better in all three scenarios comparing to the Chen and Gao methods. We also obtained the average of FPR, TPR, and TNR criteria in all three scenarios which can be seen in Tables 8–10.

Considering the values of Tables 8–10, it can be concluded that the proposed method outperforms the Chen and Gao methods.

The speed of processing refers to the classification time required by each ML algorithm to distinguish malicious attacks from normal traffic. On this graph, the horizontal axis shows the ML algorithms and the vertical axis shows the processing time required by each of the algorithms. In fact, the graph in Figure 13 shows the average processing time required for the detection of attacks.

As can be seen in Figure 17, the DTree algorithm has a higher processing speed than the other algorithms. AdaBoost was found to be the slowest algorithm in distinguishing malicious attacks from normal traffic.

## 5. Conclusion

In this paper, a new ML method has proposed to detect DRDOS attacks in SDN-based networks. The proposed method is implemented on the controller and leverages the capabilities of SDN architecture to gather useful information about DNS traffic flows. Then, it makes use of ML algorithms to detect the presence of DNS amplification attack. To train the ML algorithms, we have selected the most important features from CIC-IDS-2017 dataset. The performance of the proposed method compared to other state-of-the-art techniques has been conducted in a comprehensive manner. The simulation results confirm that using decision tree algorithm as an ML model in the proposed method achieves the shorter attack detection time, whereas SVM and GBC algorithms outperform other algorithms in terms of accuracy. Furthermore, the results showed that our proposed method is applicable and can be beneficiary in online detection of DNS-based amplification attacks in SDN. In the future work, we aim to use other new machine learning methods such as deep learning, and we can identify other types of DRDoS attacks.

## Data Availability

The experimental data consist of public datasets: CIC-IDS-2017 dataset [22] and Scapy data generator [23].

## Conflicts of Interest

The authors declare that there are no conflicts of interest.

## References

[1] Y. Zhang, L. Cui, W. Wang, and Y. Zhang, "A survey on software defined networking with multiple controllers," *Journal of Network and Computer Applications*, vol. 103, pp. 101–118, 2018.

[2] D. Kreutz, F. M. V. Ramos, P. E. Verissimo, C. E. Rothenberg, and S. Azodolmolky, "Software-defined networking: a comprehensive survey," *Proceedings of the IEEE*, vol. 103, no. 1, pp. 14–76, 2014.

[3] M. Jammal, T. Singh, A. Shami, R. Asal, and Y. Li, "Software defined networking: state of the art and research challenges," *Computer Networks*, vol. 72, pp. 74–98, 2014.

[4] F. Hu, *Network Innovation through OpenFlow and SDN: Principles and Design*, CRC Press, Florida, USA, 2014.

[5] J. S. H. Miranda, "Fault isolation in software defined networks," MSc thesis, Tecnica Lisboa University, Lisboa, Portugal, 2016.

[6] F. Hu, Q. Hao, and K. Bao, "A survey on software-defined network and openflow: from concept to implementation," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 4, pp. 2181–2206, 2014.

[7] S. T. Zargar, J. Joshi, and D. Tipper, "A survey of defense mechanisms against distributed denial of service (DDoS) flooding attacks," *IEEE communications surveys & tutorials*, vol. 15, no. 4, pp. 2046–2069, 2013.

[8] S. Dong, K. Abbas, and R. Jain, "A survey on distributed denial of service (DDoS) attacks in SDN and cloud computing environments," *IEEE Access*, vol. 7, pp. 80813–80828, 2019.

[9] Y. Gao, Y. Feng, J. Kawamoto, and K. Sakurai, "A machine learning based approach for detecting DRDoS attacks and its performance evaluation," in *Proceedings of the 2016 11th Asia Joint Conference on Information Security (AsiaJCIS)*, IEEE, Fukuoka, Japan, Auguest 2016.

[10] C. Rossow, "Amplification hell: revisiting network protocols for DDoS abuse," in *Proceedings of the Network and Distributed System Security Symposium*, NDSS, San Diego, California, USA, February 2014.

[11] F. J. Ryba, M. Orlinski, M. Wahlisch, C. Rossow, and T. C. Schmidt, "Amplification and DRDoS attack defense--a survey and new perspectives," 2015.

[12] N. Z. Bawany, J. A. Shamsi, and K. Salah, "DDoS attack detection and mitigation using SDN: methods, practices, and solutions," *Arabian Journal for Science and Engineering*, vol. 42, no. 2, pp. 425–441, 2017.

[13] R. Santos, D. S. Silva, W. D. E. Santo, and A. L. Ribeiro, "Machine learning algorithms to detect DDoS attacks in SDN," *Concurrency and Computation: Practice and Experience*, vol. 32, no. 16, Article ID e5402, 2020.

[14] P. Amaral, J. Dinis, P. Pinto, L. Bernardo, J. Tavares, and H. S. Mamede, "Machine learning in software defined networks: data collection and traffic classification," in *Proceedings of the 2016 IEEE 24th International Conference on Network Protocols (ICNP)*, November 2016.

[15] J. Xie, F. R. Yu, T. Huang et al., "A survey of machine learning techniques applied to software defined networking (SDN): research issues and challenges," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 1, pp. 393–430, 2018.

[16] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, MIT press, Cambridge, USA, 2018.

[17] O. Rahman, M. A. G. Quraishi, and C.-H. Lung, "DDoS attacks detection and mitigation in SDN using machine learning," in *Proceedings of the 2019 IEEE World Congress on Services (SERVICES)*, July 2019.

[18] C.-C. Chen, Y. R. Chen, W. C. Lu, S. C. Tsal, and M. C. Yang, "Detecting amplification attacks with software defined networking," in *Proceedings of the 2017 IEEE conference on dependable and secure computing*, Auguest 2017.

[19] F. S. D. Lima Filho, F. A. F. Silveira, A. D. M. Brito Junior, G. V. Solar, and L. F. Silveria, "Smart detection: an online approach for DoS/DDoS attack detection using machine learning," *Security and Communication Networks*, vol. 201915 pages, Article ID 1574749, 2019.

[20] S. Nanda, F. Zafari, C. Decusatis, E. Wedaa, and B. Yang, "Predicting network attack patterns in SDN using machine learning approach," in *Proceedings of the 2016 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN)*, November 2016.

[21] I. L. Meitei, K. J. Singh, and T. De, "Detection of DDoS DNS amplification attack using classification algorithm," in *Proceedings of the International Conference on Informatics and Analytics*, Pondicherry, India, Auguest 2016.

[22] ids-2017@www.unb.ca, 2021, https://www.unb.ca/cic/datasets/ids-2017.html.

[23] Scapy Python Library@www.scapy.net, 2021, https://scapy.net/.

[24] Ryu Software Defined Networking Framework@ryu.readthedocs.io, 2021, https://ryu.readthedocs.io/en/latest/getting_started.html.

[25] Mininet Network Emulator@mininet.org, 2021, http://mininet.org/overview/.