The Journal of Supercomputing (2021) 77:2383–2415
https://doi.org/10.1007/s11227-020-03323-w

The DDoS attacks detection through machine learning and statistical methods in SDN

Afsaneh Banitalebi Dehkordi[1] · **MohammadReza Soltanaghaei**[1] · **Farsad Zamani Boroujeni**[1]

Published online: 15 June 2020

Abstract

The distributed denial-of-service (DDoS) attack is a security challenge for the software-defined network (SDN). The different limitations of the existing DDoS detection methods include the dependency on the network topology, not being able to detect all DDoS attacks, applying outdated and invalid datasets and the need for powerful and costly hardware infrastructure. Applying static thresholds and their dependency on old data in previous periods reduces their flexibility for new attacks and increases the attack detection time. A new method detects DDoS attacks in SDN. This method consists of the three collector, entropy-based and classification sections. The experimental results obtained by applying the UNB-ISCX, CTU-13 and ISOT datasets indicate that this method outperforms its counterparts in terms of accuracy in detecting DDoS attacks in SDN.

Keywords Distributed denial-of-service attacks · Software-defined networks · High-volume DDoS attack · Low-volume DDoS attack · Network security

1 Introduction

The SDN is a new architecture consisting of the three data, control and application plane layers, where data and control layers are independent of each other, as shown in Fig. 1. The data plane consists of switches and routers involved in network traffic forwarding; the control plane constitutes the network intelligent section consisting

✉ MohammadReza Soltanaghaei
soltan@khuisf.ac.ir

Afsaneh Banitalebi Dehkordi
Banitalebi97@gmail.com

Farsad Zamani Boroujeni
f.zamani@khuisf.ac.ir

[1] Department of Computer Engineering, Isfahan (Khorasgan) Branch, Islamic Azad University, Isfahan, Iran

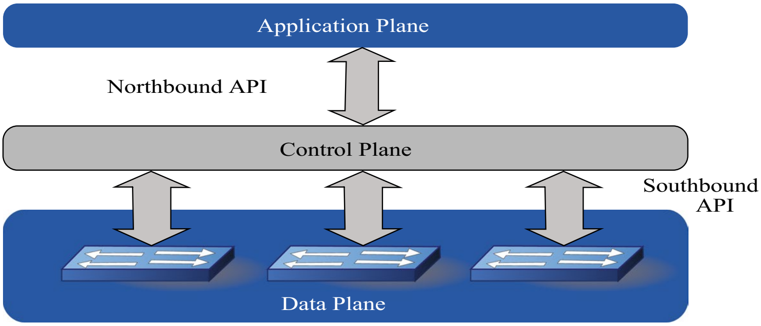


Fig. 1 SDN architecture

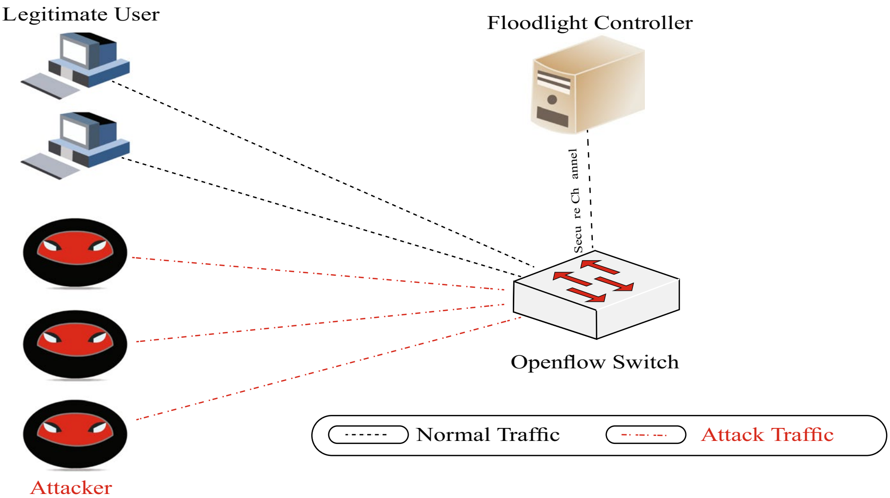


Fig. 2 DDoS attack Schema in SDN

of NOX, POX, Beacon, Floodlight and OpenDaylight controllers, and the application plane contains applications for SDN configuration [1].

The IT organizations may possibly encounter security procedures like DDoS attacks due to the lack of network coherency during re-configuration of the networks to SDN [2]. The DDoS is one of the most adverse attacks in the Internet realm, which weakens the network and the server by influencing the network bandwidth or connectivity in providing regular service [3], as shown in Fig. 2, where as observed the attackers put in too many requests to the open-flow switch from different hosts in a simultaneous manner, thus facing the network with difficulties.

The DDoS attacks target a wide spectrum of different resources and sites, beginning from servers' banks up to new sites by introducing big challenges for the managers and users of these systems. On Feb 28, 2018, the GitHub site, one of the most important code variety perceptions for programs, was attacked with a high mass traffic of 1.3 Tbps volume, which made it to become off-line for 5 min. This attack introduced many problems to this site [4]. In a time interval within February 5–March 1, 2019, about 17 DDoS attacks were made on University of Albany site, which disturbed the server therein for at least 5 min. Though the data related to the instructors and students were exempt, some of the servers become off-line [5]. These nonstop attacks necessitate devising procedures in detecting and preventing the DDoS attacks.

There exist approaches in this context which next to their advantages have the following drawbacks.

Difficulty in selecting the appropriate time periods for monitoring the traffic in periodic methods [6], the shortcoming and delays in detecting DDoS attacks may lead to losing resources such as bandwidth and CPU [7], deactivation of the controller and switch, unwanted increase in response time [8] and maintaining the network security at high cost of adding hardware therein.

A method including statistical and machine learning methods involved in SDN is proposed in this article to overcome the available drawbacks in DDoS attack detection.

In this method, the mechanism for selection of time periods is applied in monitoring attacks, something not considered in the available methods. Attempt is made here to select the best time period for achieving the maximum detection rate, which is not necessarily of the lowest or highest volume. Periodic monitoring and scheduled traffic screening increase the efficiency of the controller in terms of the workload. Another advantage of this idea is that no custom hardware is necessary to detect attacks. This method increases the accuracy of DDoS detection and provides independence from the network topology.

The assessed attacks here are of the HTTP-based application layer attack type [9], which are observed in their low-volume or high-volume states. The high-volume attacks send many requests to a server or computer and consume extra bandwidth and processors therein [10], while the low-volume attacks have lower entry traffic mass capable of being deceived by expert or impostor attackers [11]. In this method, both these states are assessed. This model consists of collector, entropy-based and classification sections.

The statistical information from switches and host is collected in the controller sections. The entropy volume and the static and dynamic thresholds are calculated through the entropy-based section.

The 15 features for the hosts of the same flow and recorded data samples for incoming packets are extracted through the classification section. The samples are fed into the classification section as the training inputs to devise models through different classification algorithms.

This method yields 99.85% accuracy with 0.1 FPR on UNB-ISCX and 99.12% on CTU-13 dataset. These results indicate this model's outperformance versus its counterparts. The main contribution of this article is to combine machine learning and statistical methods to improve the detection of DDoS attacks in SDN networks. In the available methods, the advantage of statistical methods and machine learning combination is not addressed in achieving higher detection performance.

This article is organized as follows: The literature is reviewed in Sect. 2; the method is proposed in Sect. 3; the datasets are presented in Sect. 4; the model is evaluated in Sect. 5; the model is implemented in Sect. 6; the results are expressed in Sect. 7; the analysis are run in Sect. 8; the experiments are compared in Sect. 9; and the article is concluded in Sect. 10.

2 Literature review

There exist many studies on DDoS attack detection. The findings of some of the available articles are briefed in this section.

Researchers in [12] applied the *K*-means clustering and Naive Bayes method for DDoS attack detection, consisting of: (1) clustering the similar data as to their behaviors in groups and labeling all data according to K cluster and (2) classifying the labeled data groups through Naive Bayes algorithm.

The computer vision technique is applied to detect DDoS attacks in [13], where unlike the statistical and machine learning methods, the traffic records are considered as images and detecting the attacks is viewed as a computer version issue. A multivariable coherence analytical method is introduced for accurate traffic record detection and its conversion into images. This method is named the Earth mover's distance (EMD) computed based on the measured distance between two probable distributions.

As to the known and unknown DDoS attacks, researchers in [14] applied the artificial neural network (ANN) and revealed that the method is subject to algorithm training through the given dataset. Their proposed method is compared with its counterparts such as the backpropagation (BP), Chi-square and support vector machines (SVM) and Snort. They obtained a detection accuracy of 98%.

The DDoS attack detection in cloud computing and SDN networks is assessed in [15], where different models with features are applied to the datasets involved in both the training and test. For them, to increase efficiency updating is a must. Among the three proposed DDoS attack detection models in SDN networks, the best is Mglobal with 89.30% accuracy.

The authors in [16] applied different features to detect whether an attack has occurred or not. Because there exist more than one major parameter in judging DDoS attacks, the significant issue is related to how these parameters are determined; that is, the destination Internet Protocol (IP) address is considered as one of the attack detection parameters which can be detected by entropy. The detection method is evaluated through this model and many parameters.

A fast attack detection method is proposed in [17] to decrease the controllers and switches workload, where the neural network algorithm is applied. A combination of entropy-based and classification algorithms is presented as well. This method can detect both the high-volume and low-volume DDoS attacks.

To implement their own model, researchers in [18] applied the two data mining algorithms of C5.0 and Ripper. Their model is tested on UNB-ISCX datasets and a detection rate of 99% plus is achieved.

Researchers in [19] applied a statistical approach to detect the attacks next to learning machine techniques. In the statistical approach, usually the predetermined distributions are applied to model the traffic network's normal and abnormal behaviors, in addition to the distance measures techniques, and in the machine learning stage, the *K*-Means, SVM, decision tree, Naive Bayes algorithm and AI algorithm are applied as a classifier.

A new solution for determining DDoS attack in IOT network infrastructures is proposed in [20], where for managing high traffic flows, the sFlow- and adaptive polling-based sampling techniques are applied in the data-plane layer. After sampling the distributed traffic in data plane, to increase real attack detection, the Snort-IDS and stacked autoencoders (SAE), an unsuperficial algorithm, are applied to obtain the high accuracy and low FPR to distinguish normal traffic from attack.

In a general assessment in [21], the deep learning modules of convolutional neural networks, deep neural networks, recurrent neural networks and deep Boltzmann machines models are of concern. The efficiency of the model of concern is determined by assessing every model in both the binary and multiclass categories by applying the CSE-CIC-IDS2018 and BoT–IoT datasets which contain real traffic. They revealed that implementation of their method is costly and complex because it requires special hardware such as Graphic Process Unit (GPU) and hundreds of software machines.

The researchers in [22] proposed a dynamic multilayer perceptron (MLP) combined with a feature selection technique to detect DDoS attacks, where a feedback mechanism is applied to promote and reconstruct the detector system when detection is not accurate. In their model, as the complexities of traffic network increase and change, some of the selected features will not be able to distinguish the traffic and normal attacks and determine the failure therein. The proposed method in their article in comparison with their counterparts can be of good functionality, while applying feedback mechanism here can enhance FPR and FNR.

3 The proposed method

In this study, a combination of entropy-based method and classification algorithm is applied for detecting high-volume and low-volume DDoS attacks. A two-class classification task for distinguishing normal flows from attacks is of concern here. The three applications introduced in Floodlight controller [23] for collecting flows and calculating entropy are shown in Fig. 3.

The method shown in Fig. 3 consists of the collector, entropy-based and classification sections, which operate together to detect the DDoS attacks that occur in the Floodlight controller. Each section is introduced in the following text.

3.1 Collector section

Both the statistics of the network flows and communications recorded by switches for a specific period of time are collected in this section. These statistics include

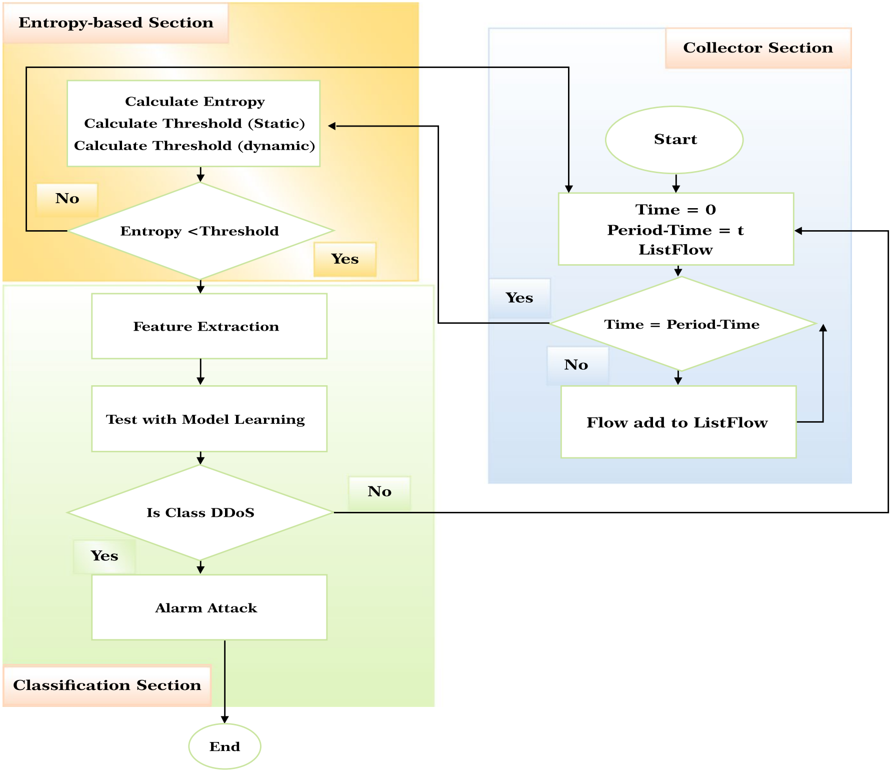


Fig. 3 Proposed method

the total count of the bytes sent, the count of packets sent and the flow time. Upon establishing a connection between two hosts, the first packet is sent to the controller, to be stored next to IP source, source port, destination IP, destination port, packet bytes and packet arrival time [24]. This phenomenon holds true for all packet-in messages. After making all the flows available, the statistics between the two hosts are obtained and given to the controller.

3.2 Entropy-based section

Here, entropy is applied to detect most of the attacks. Providing a fast and convenient manner in filtering suspicious flows is the main advantage of entropy-based filtering. This section is easily developed and implemented in SDN network environments, where low CPU load and easy implementation by the controller suffice.

The DDoS attacks impose additional overhead and disrupt Web activities; thus, the target system is measured by calculating the entropy of each IP in SDN networks. To calculate the entropy, it is assumed that there exists a time window, W, with n distinct elements and $X_{(i,t)}$ is the observation i in the set at time t. The size of W in Eq. (1) is named as the size of time window [25].

$$W = \{X_{(1,t)}, X_{(2,t)}, \dots, X_{(n,t)}\} \tag{1}$$

where W is the time window, and $X_{(i,t)}$ is the count of flows i in time t at n different possible states.

The probability of $X_{(i,t)}$ occurring in W is calculated through Eq. (2):

$$p(X_{(i,t)}) = \frac{X_{(i,t)}}{n} \tag{2}$$

where $p(X_{(i,t)})$ is the occurrence probability of each $X_{(i,t)}$ in W.

To calculate the entropy $H_{(i,t)}$, the probability of each element in the set should be multiplied by its logarithm and summed through Eq. (3).

$$H_{(i,t)} = -\sum_{i=1}^{n} P(X_{(i,t)}) \log P(X_{(i,t)}) \tag{3}$$

where $P(X_{(i,t)})$ is the occurrence probability of each IP.

If the calculated entropy < threshold (Thr), as expressed in Eq. (4), then the occurrence of an attack is reported.

$$H_{(i,t)} < Thr \tag{4}$$

where Thr is a threshold in this network.

The optimal entropy for each period is determined by testing different time periods. Changing the time periods is very easy in the SDN controller, and this flexibility is one of the advantageous features in SDN networks. Both the time period duration and threshold size are effective in attack detection. The static and dynamic thresholds are introduced in [26], and the detection of high-volume DDoS attacks with DARPA2000 is assessed in [27]. The DARPA2000 datasets are detected by experts based on the DDoS attacking software, indicating that these attacks are simple in structure and type in spite of the complexity of the real data. In this study, these two thresholds are evaluated for both the high- and low-volume attacks by running tests on datasets collected from actual SDN networks and a method is proposed and compared for threshold calculation so as to select the best threshold volume for each type of attack.

3.2.1 Static threshold

This threshold has a static volume, based on the packets specified to the DDoS attacks. Normal traffic and attack traffic are transmitted separately to the network at different time periods. The mean volume of the entropy for different time periods is calculated once for the attack mode and once for the normal mode. Consequently, the static threshold is obtained through Eq. (5).

$$Thr = T_1 = \frac{\overline{H}_{attack} + \overline{H}_{normal}}{2} \tag{5}$$

where $\overline{H}_{attack}$ is the entropy average in normal flows and $\overline{H}_{normal}$ is the entropy average in the attack flow.

3.2.2 Dynamic threshold

A computational method based on time sequence is applied to calculate the dynamic threshold, because it is fast in detecting DDoS attacks in small time windows, as in Eq. (6):

$$Thr = T_2 = \bar{H}_{(i,t-1)} + C_d.\sigma_{H_{(i,t-1)}} \tag{6}$$

where $\bar{H}_{(i,t-1)}$ is the calculated mean volumes of the entropies, as in Eq. (7), $\sigma_{H_{(i,t-1)}}$ is the standard deviation (SD), at time $t-1$, as in Eq. (8), and C_d is the constant volume of a coefficient determined based on experiments, which does not depend on the time period and the volume of previous entropy.

$$\overline{H}_{(i,t-1)} = \frac{1}{t}\sum_{i=1}^{t-1} H_{(i,t-1)} \tag{7}$$

$$\sigma_{H_{(i,t-1)}} = \frac{1}{t}\sum_{i=1}^{t-1} (H_{(i,t-1)} - \bar{H}_{(i,t-1)})^2 \tag{8}$$

where $H_{(i,t-1)}$ calculates the entropy levels for different time periods and $\bar{H}_{(i,t-1)}$ is the entropy average. At this stage, the entropy volume and dynamic threshold are calculated for each time period by applying a C_d value specifically calculated for the dataset. If the entropy value < the threshold, the attack is detected and a volume is added to the alarm rate parameter that calculates the volume of attack alarms. C_d is an experimental parameter, and its volume is influenced by the accuracy of attack detection. Because selecting the best value for C_d is subjective, depending on different parameters, to calculate the best C_d for each time period, it is better to consider an interaction between the different parameters. One of these parameter has to do with the ability of detecting all attacks, which should not make the count of time periods different, require less computational burden and generate low false alarm rates.

To select the best C_d, first, in each time period, the TPR with volume of 100 is of concern, next among the selected situations where the FPR is the lowest is of concern, consequently, the obtained C_d volume is considered as the best C_d at the best time period.

By determining the best time period and best C_d volume, that portion of the flow subject to potential attack is detected, selected and forwarded to the classification section to increase the attack detection accuracy. Because this step eliminates a portion of the normal flow that is correctly detected, the count of the normal flow and attack flow is balanced before being delivered to the classification section.

Fig. 4 Flow between two hosts in SDN

Table 1 Features extracted

Features	Explanations
Host A	
SenderSrc	The count of one-way connections of the transmitter host to the total connections of the desired node ratio
ReceiverSrc	The count of connections of the recipient host to the total connections of the desired node ratio
EntropySentSrc	Calculating the entropy of the flows, the appropriate host of the transmitter
EntropyReceiveSrc	Calculating the entropy of the flows, the appropriate host of the recipient
CountSentSrc	The count of the flows, appropriate host of the transmitter
CountReceiveSrc	The count of the flows, appropriate host of the recipient
Host B	
SenderDst	The count of one-way transmitter host to the total connections of the appropriate node ratio
ReceiverDst	The count of connections recipient host to the total connections of the node ratio
EntropySentDst	Calculating the entropy of the flows, the appropriate transmitter host
EntropyReceiveDst	Calculating the entropy of the flows, the appropriate recipient host
CountSentDst	The count of flows, appropriate transmitter host
CountReceiveDst	The count of flows, appropriate recipient host
Both A and B	
CountPacket	The count of packets in the relevant flow
SumByte	The total count of bytes with their specified flows
Packet-in	This feature is the first packet which transmits a flow for any host that begins the flow whether A or B and is raised in SDN networks

3.3 Classification section

Here, a portion of the dataset at entropy-based section is identified as attack and considered as the entry. As observed in Fig. 4, every flow is considered as one edge forming both the ends of the host's graph node

For collecting the flows and extracting the features of concern, each IP is first considered as a node, and all the connections between those two and other nodes are applied to obtain the features. In feature collection, the neighbors of a node are of concern, as given in Table 1.

A set of 15 features is extracted for training the classifiers including 12 features for both the host of a flow (six features for each host) and three shared features among the hosts. The features independent of speed and type of attack during machine learning are extracted through this proposed method, which is able to detect both high-volume and low-volume DDoS attacks (Table 1). After extracting the features, the training samples are given as inputs to the BayesNet, J48, RandomTree, logistic regression, REPTree classifiers classification algorithms [28] to construct the classification models.

To train and test the data, the K-fold method $K = 10$ is of concern [29]. In this method, for data separation, they must be distributed in tenfold in a random manner. Each period in this implementation has some flows where all are put in uniform folds. This operation is iterated for ten times, and the classification algorithms are obtained for modeling.

By comparing the obtained results, the best classification algorithm that improves the accuracy of attack detection is selected for each case.

4 The datasets

To evaluate the performance of this method, the well-known datasets, UNB- ISCX[1] [30] and CTU-13[2] [31], are selected and applied. Next to these datasets, ISOT[3] [32] is applied only for the normal traffic. The first dataset, UNB-ISCX, is prepared by Canadian Institute of Cyber Security,[4] consisting of different sections. In this article, the two sections of the datasets, namely ISCX-SlowDDos-2016 and ISCX-IDS-2012, are applied. The ISCX-SlowDDos-2016 dataset contains both the high- and low-volume DDoS attacks. The ISCX-IDS-2012 dataset contains different types of attacks like the HTTP DDoS. The HTTP GET DDoS attack is generated by an IRC Botnet and a brute force SSH attack. Each scenario contains a pcap file of both the attack and normal flows. The CTU-13 dataset is collected from the Czech University,[5] with the objective of generating a real traffic for the botnet combined with normal and background traffic. This dataset is extracted from 13 different samples of different botnet scenarios. In this study, scenarios 10 and 11 are applied to detect the DDoS attacks. The traffic applied in the CTU-10 scenario is of the UDP DDoS type and in the CTU-11 scenario of the ICMP DDoS type. The ISOT dataset is generated through the IT Research Center at the University of Victoria.[6] In this study, the normal section of this dataset is applied in combination with other dataset (Table 2).

1 http://www.unb.ca/cic/datasets/ids-2017.html.
2 https://www.stratosphereips.org/datasets-ctu13.
3 https://www.uvic.ca/engineering/ece/isot/datasets/.
4 https://www.unb.ca/cic/.
5 https://www.esncz.org.
6 https://www.uvic.ca/.

Table 2 Dataset statistics

Period of time (s)	Avg duration (s)	Avg packet number	Avg flow number	Avg packet-in flow
ISCX-SlowDDos-2016				
15	1.19	1106.58	101.24	13.40
20	1.17	1475.09	134.93	13.44
25	1.17	1143.41	168.59	13.37
30	1.16	2211.62	202.25	13.11
45	1.15	3315.38	303.06	13.38
ISCX-IDS-2012				
15	1.9666	87.1584	3.0967	25.8474
20	2.0097	92.974836	3.3031	23.62475
25	2.0634	96.8227	3.4408	22.9491
30	2.0952	99.6229	3.5392	23.0845
45	2.13748	1.4.6096	3.71645	21.07966
CTU-10				
15	0.700	9930.22	565.56	30.12
20	0.698	13,211.80	752.08	29.97
25	0.696	16,505.15	939.16	29.75
30	0.696	19,782.14	1124.83	29.70
45	0.695	29,564.09	1680.69	29.49
CTU-11				
15	7.57	677,457.77	9337.82	2.65
20	6.52	70,098.21	9661.93	2.54
25	4.99	72,601.71	100,005.93	2.39
30	4.31	70,104.66	9665.17	2.31
45	4.74	96,807.43	13,343.20	2.40

5 Evaluation

The performance of this proposed method is evaluated through the accuracy (ACC), precision, recall, F-measure, true positive rate (TPR) and false positive rate (FPR) metrics, the related equations of which are presented in Table 3. Let P be the count of the actual positive (attack) examples and N be the count of the actual negative (normal). The TPR measure is the ratio of attacks correctly recognized as attack, and the FPR is the normal cases, wrongly classified as attack. The alarm rate is the examples classified as attack to the total count of classified samples ratio. By applying WEKA Software, the mean absolute error (MAE), root-mean-square error (RMSE), root-mean-square(RMS) and root absolute error (RAE) [33] are obtained as shown in Table 3.

Table 3 Equation parameters

Parameter	Equation
Accuracy	$\frac{TP+TN}{TP+FN+TN+FP}$
Precision	$\frac{TP}{TP+FP}$
Recall	$\frac{TP}{TP+FN}$
F-measure	$\frac{2\times Precision\times Recall}{Precision+Recall}$
TPR	$\frac{TP}{TP+FN}$
FPR	$\frac{FP}{FP+TN}$
Alarm rate	$\frac{(TP+FP)}{(TP+TN+FP+FN)}$
MAE	$\frac{\sum_{i=1}^{n} \lvert y_i-x_i \rvert}{n}$
RMSE	$\sqrt{\frac{\sum_{i=1}^{n} (x_i-y_i)^2}{n}}$
RAE	$\frac{\sum_{i=1}^{n} \lvert x_i-\bar{x}_i \rvert}{\sum_{i=1}^{n} \lvert y_i-\bar{y} \rvert}$

6 Implementation

For this implementation, the necessary software and tools are introduced as follows:

6.1 The implementation environment and tools

The experiments are run on an ASUS laptop with an AMD (Bristol Ridge), FX-9830P CPU 2.8 GHz processor accompanied with a 12GB of RAM. The operating system is the Linux Ubuntu 14.04 LTS run on Window 8.1 host machine. The Floodlight is chosen as the network controller and the Mininet 2.2.1 [34] for network simulation.

The Eclipse Neon.3 [35] is applied here for programming the modules in the Floodlight controller.

6.2 SDN network configuration

To implement this method in SDN, first a module must be implemented in the Floodlight controller which would detect the DDoS attacks. For this purpose to commence the SDN networks, the network configuration is a must prior to the prerequisites related to the controller and network installation and adjustment. For this purpose, a version of Java must be installed. In this study, the Java version 8 is installed followed by installing the Eclipse Neon 3 to configure the Floodlight controller operators according to the following steps [36]:

1. First install jdk 8 (using “java8offline.txt”)
2. $ cd

3. Download Eclipse Neon.3 installer:
4. $ wget http://ftp.ussg.iu.edu/eclipse/oomph/epp/neon/R/eclipse-inst-linux64.tar.gz
5. $ sudo apt-get remove eclipse eclipse- jdt eclipse-platform eclipse-rcp eclipse-pde eclipse- platform-data $ rm -r /.eclipse/
6. $ sudo tar -zxvf /eclipse-*.tar.gz&& cd eclipse-*
7. $. /eclipse-inst
8. //wait a minute, and then choose Eclipse IDE for Java Developers
9. //choose Install
10. //choose accept (for everything)
11. //select launch

The Eclipse Neon.3 is configured as follows:

1. $ ant eclipse
2. Open eclipse and create a new workspace:
   File – > Import – > General – > Existing Projects into Workspace.
   Then, click "Next."
3. From Select root directory, click Browse. Select the parent directory where you placed floodlight earlier.
4. Check the box for *Floodlight*. No other projects should be present and none should be selected.
5. Click Finish.

At this stage, after installing and adjusting the Eclipse Neon.3, the same should be done in the SDN, with respect to the controller Floodlight therein as follows:

1. sudo apt-get install build-essential git ant maven python-dev
2. git clone git://github.com/floodlight/floodlight.git
3. cd floodlight
4. git submodule init
5. git submodule update
6. ant
7. sudo mkdir /var/lib/floodlight
8. sudo chmod 777 /var/lib/floodlight

Now that the Floodlight controller is installed, its implementation is subject to the following procedures:

1. sudo apt-get remove openvswitch-testcontroller
2. java -jar target/floodlight.jar
3. in order to see web GUI open link below in browser:

http://127.0.0.1:8080/ui/pages/index.html

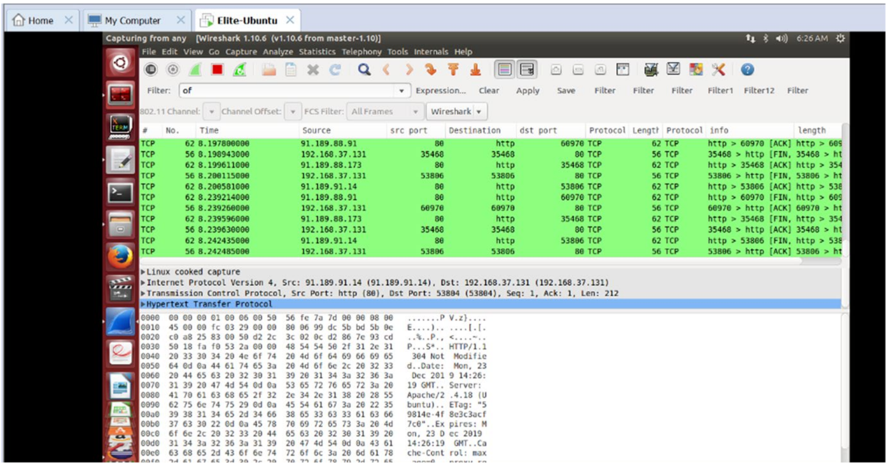


Fig. 5 Flow assessment in Wireshark

To install this Floodlight controller in SDN, the following steps are observed to design this proposed module in DDoS detection:

1. Expand the Floodlight item in the Package Explorer and find the *src/main/java* folder.
2. Right click on the *src/main/javafolder* and choose New/Class.
3. Enter net.floodlightcontroller.ddosdetection in the "Package" box.
4. Enter DdosDetection in the Name box.
5. Next to the Interfaces box, choose Add ...
6. Add the IOFMessageListener and the IFloodlightModule, click OK.
7. Click Finish in the dialog.

Now, this proposed algorithm is added to the Floodlight controller in the form of DdosDetection module, but it is not able to be run and must be registered in the controller first, if the following steps are met:

1. src/main/resources/META-INF/services/net.floodlightcontroller.core.module.IFloodlightModulenet.floodlightcontroller.ddosdetection.DdosDetection
2. src/main/resources/floodlightdefault.properties net.floodlightcontroller.ddosdetection.DdosDetection

At this point, through this proposed module, to simulate the attack, the dataset of concern is injected into the network through the attacker and the Floodlight controller is able to detect the DDoS attacks. This process is run in Mininet emulator through Tcpreplay tool [37], which with its selected speed redistributes the pcap *files* in the network. To detect the DDoS attacks, the data of concern required for

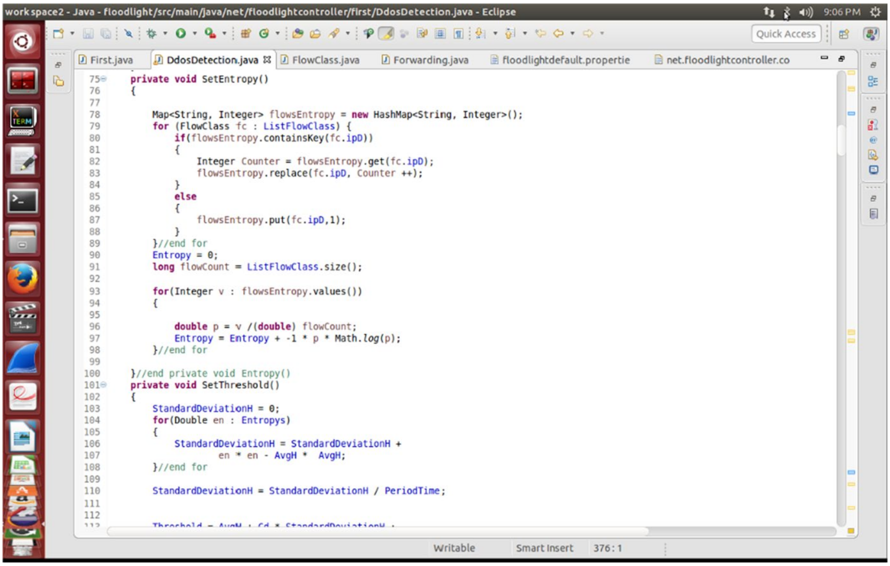


Fig. 6 Implementation of affect detector module in Floodlight controller in Eclipse Neon.3

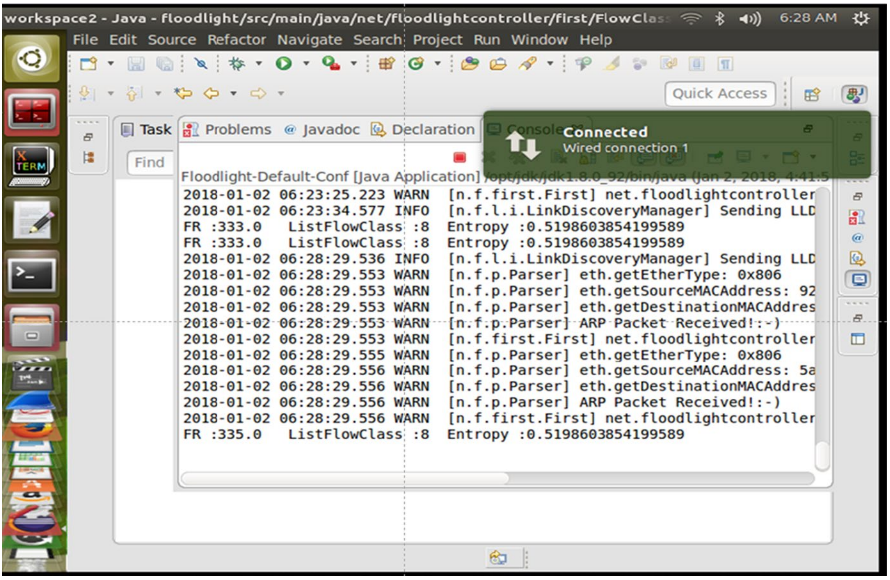
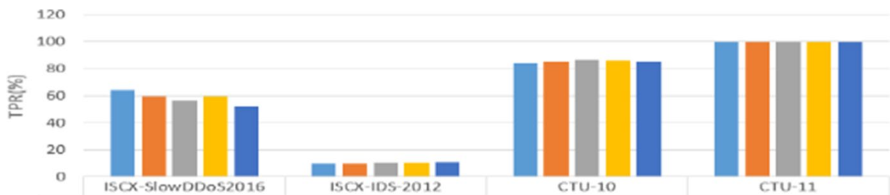


Fig. 7 Implementation of this Module and the Entropy calculation

the injectable dataset must be collected from the switch, host and the available communications. The data related to the packages and flows transmitted in the network in wireshark are shown in Fig. 5.



	ISCX-SlowDDoS2016	ISCX-IDS-2012	CTU-10	CTU-11
15	63/97	9/94	84/21	100
20	60/09	9/99	84/7	100
25	56/81	10/55	86/95	100
30	60	10/53	86/2	100
45	52/42	11/05	84/61	100

Fig. 8 Evaluation results of static threshold in high-volume DDoS attack detection

After flow data extraction and its transmission to the Floodlight controller, by applying this module, the attack detection assessment begins.

A portion of this module related to the entropy calculations is implemented through the Floodlight controller in Eclipse Neon.3 which is shown in Fig. 6.

The implementation code reveals a portion of the control module for this purpose. The module implementation in the controller and obtaining the entropy volume are shown in Fig. 7.

The modules implemented in the controller which begin to calculate the entropy by summing the flows and providing the list flow are shown in this figure.

After the attack detection is made in the entropy-based section, some of the results identified as attack are sent to the other sections of the learning machine, where by running classification algorithm in WEKA Software, the *K*-fold method detects the attacks.

7 Results of the experiments

The objective here is to identify the low-volume DDoS and high-volume DDoS attacks in their separate sense. In this study, the periods are within 10–240 s range, among which the ones within 15–45 s range are considered the best for this purpose. This is justified by the fact that more attacks are detected in a shorter time within this period.

If a long time period is chosen, the response time would increase, and the attack detection would be delayed as well, that is, the detection may occur after the attack cause destruction, thus making controller and switches handle large volumes of attacks flows and causing harm thereof. By choosing short time periods, the attack detection process begins early and makes the controller to overuse the CPU and network bandwidth resources, which affects the controller performance, and consequently, the attacks spread over a long time span, that is, a negative effect on attack detection. With respect to these two points, different time periods within 15–45 s range constitute the best choice.

Table 4 Evaluation results of dynamic threshold for high-volume DDoS attack detection in the ISCX-SlowDDos-2016 database

Period time (s)	C_d	Alarm rate (%)	TPR (%)	FPR (%)	ACC (%)	Precision (%)	*F*-measure (%)
15	− 1	20.54	88.60	16.88	83.39	22.00	35.25
	0	54.89	99.26	52.21	50.13	9.22	16.88
	1	87.16	100	86.47	17.93	5.85	11.06
20	− 0.2	47.22	100	44.37	57.89	10.82	19.53
	0	55.03	100	52.61	50.07	9.28	17.00
	1	86.35	100	85.62	18.75	5.92	11.17
25	− 0.2	45.88	100	42.82	59.45	11.64	20.86
	0	54.84	100	52.29	50.50	9.74	17.75
	1	87.24	100	86.52	18.09	6.12	11.54
30	− 0.2	46.45	100	43.49	58.78	11.28	20.27
	0	54.80	100	52.30	50.43	9.56	17.45
	1	87.27	100	86.57	17.96	6.00	11.33
45	− 0.3	40.16	100	36.67	65.34	13.71	24.12
	0	53.85	100	51.16	51.65	10.22	18.55
	1	87.27	100	86.53	18.23	6.31	11.87

Table 5 Results of dynamic threshold in high-volume DDoS attack detection in the ISCX-IDS-2012 dataset

Period of time (s)	C_d	Alarm rate (%)	PR (%)	FPR (%)	ACC (%)	Precision (%)	*F*-measure (%)
15	0	11.57	59.87	7.84	89.84	37.08	45.80
	1	18.51	78.89	13.84	85.63	30.56	44.06
	1.9	20.06	100	13.88	87.11	35.74	52.66
20	0	11.47	60.49	7.75	90.00	37.17	46.05
	1	18.50	78.65	13.94	85.54	29.97	43.41
	1.8	20.03	100	13.97	87.01	35.20	52.07
25	0	11.49	60.85	7.74	90.03	37.36	46.3
	1	18.57	79.84	13.92	85.64	30.34	43.97
	2	20.03	100	13.95	87.03	35.24	52.12
30	0	11.5	61.53	7.73	90.11	37.47	46.58
	1	18.57	79.93	13.95	85.62	30.16	43.79
	1.8	20.01	100	13.98	86.99	35.02	51.88
45	0	11.45	63.03	5.88	92.42	38.12	47.51
	1	18.64	80.86	10.83	88.72	30.04	43.81
	2	19.99	100	10.84	89.74	34.65	51.46

Table 6 Results of dynamic threshold in high-volume DDoS attack detection in the CTU-10 dataset

Period of time (s)	C_d	Alarm rate (%)	TPR (%)	FPR (%)	ACC (%)	Precision (%)	*F*-measure (%)
15	− 1	24.44	88.59	17.27	83.31	36.46	51.66
	0	38.21	92.98	32.09	70.43	24.48	38.75
	2.9	83.14	97.36	81.55	26.39	11.78	21.02
20	− 1	22.30	9.41	14.86	85.56	40.00	55.27
	0	36.15	96.47	29.46	73.12	26.62	41.73
	3.9	92.37	98.82	91.65	17.37	10.67	19.26
25	− 1	21.40	89.85	13.70	86.65	42.46	57.67
	0	35.48	94.20	28.87	73.46	26.85	41.80
	3.8	93.25	98.55	92.65	16.56	10.69	19.29
30	− 1	21.26	93.10	13.11	87.52	44.62	60.33
	0	34.62	94.82	27.78	74.51	27.91	43.13
	4	95.07	98.27	94.71	14.76	10.53	19.03
45	− 1	18.37	92.30	9.94	90.28	51.42	66.05
	0	34.12	94.87	27.19	75.06	28.46	43.78
	1.4	71.39	97.43	68.42	38.32	13.97	24.43

Table 7 Results of dynamic threshold in high-volume DDoS attack detection in the CTU-11 dataset

Period of time (s)	C_d	Alarm rate (%)	TPR (%)	FPR (%)	ACC (%)	Precision (%)	*F*-measure (%)
15	− 1.5	42.18	100	32.72	71.87	33.33	50.00
	0	54.68	100	47.27	59.37	25.71	40.90
	1	73.43	100	69.09	40.62	19.14	32.14
20	− 1.3	58.33	100	51.21	56.25	25.00	40.00
	0	62.50	100	56.09	52.08	23.33	37.88
	1	68.75	100	63.41	45.83	21.21	35.00
25	− 1.8	50.00	100	42.42	63.15	26.31	41.66
	0	52.63	100	45.45	60.52	25.00	40.00
	1	60.52	100	54.54	52.63	21.73	35.71
30	− 1.5	53.12	100	44.44	62.50	29.41	45.45
	0	59.37	100	51.85	56.25	26.31	41.66
	1	65.62	100	59.25	50.00	23.80	38.46
45	− 1.4	57.14	100	47.05	61.90	33.33	50.00
	0	66.66	100	58.82	52.38	28.57	44.44
	1	66.66	100	58.82	52.38	28.57	44.44

7.1 High-volume DDoS attack detection results

In this section, the results of the experiments on various datasets are described with the purpose of high-volume DDoS attack detection.

Table 8 Value of the best C_d and the best time period for attack detection in different datasets

Dataset name	Attack type	Best time period	Optimal C_d value
ISCX-SlowDDos-2016	High volume	45	− 0.3
ISCX-IDS-2012	High volume	45	2
CTU-10	High volume	45	1.4
CTU-11	High volume	15	− 1.5

7.1.1 The results of entropy-based section

The results of applying entropy-based method with static threshold on UNB-ISCX and CTU-13 datasets are bar-charted in Fig. 8, where, as observed, this parameter fails to identify attacks in the periods that contain both normal and attack flows.

The findings in Fig. 8 indicate that the static threshold method lacks proper functionality in detecting high-volume DDoS attacks in all datasets except for CTU-11. Now, the dynamic results are assessed:

The results of the entropy-based method for high-volume DDoS attack detection through the dynamic threshold are presented in Tables 4, 5, 6 and 7.

All assessments in Tables 4, 5, 6 and 7 at all time periods take place within 15–45 s range. The volume of C_d and the experimental volume in dynamic threshold equations are assessed within −4 to 4 range. Not all these volumes are expressed in the tables, but the ones more effective in attack predictions.

In these tables, for different time periods in the first step, TPR becomes 100, which is of concern, and when FPR volume is low at all states, it is selected as the optimized C_d volume of the period (Tables 4, 5, 6 and 7). The dataset in this section is identified as a portion of the attack and is directed to the classification section. The results of different dataset evaluations are tabulated in Table 8.

The results in Table 8 indicate that the appropriate range for selecting C_d values for different datasets is between −2 and 2.

7.1.2 The results of classification algorithms

By applying the BayesNet, J48, logistic regression, RandomTree and REPTree classification algorithms, a module is developed to determine the normal and attack flows.

To train and test the data, the *K*-fold method is applied where $K = 10$ is of concern. The volume of the parameters of concern in learning and testing for different classification algorithms is shown in Table 9a–e.

The data regarding the ten steps of *K*-fold method for training and test and the average therein for the ISCX-SlowDDos2016 dataset are given in Table 10a–e.

The data regarding ten steps of *K*-fold method for training and test and the average therein for the ISCX-SlowDDoS2016, ISCX-IDS-2012, CTU-10, ctu-11 datasets are presented in Table 11.

Table 9 Volume of the parameters of concern in learning and testing for different classification algorithms applied in *K*-fold method

(a) J48 algorithm	
J48 classifier parameters	
Batch size	100
Binary split	False
Collapse tree	True
Confidence factor	0.25
doNotMakeSplitPointActualValue	False
minNumObj	2
numDecisionPlaces	2
numFolds	3
reduceErrorPruning	False
saveinstanceData	False
Seed	1

(b) REPTree algorithm	
REPTree classifier parameters	
Batch size	100
displayModelInOldFormat	False
doNotCjeckCapabilities	False
Initial count	0.0
maxDepth	− 1
minNum	2.0
minVarianceProp	0.001
noPruning	False
numDecimalPlaces	2
numFolds	3
Seed	1

(c) Naive Bayes algorithm	
Naive Bayes classifier	
Batch size	100
Debug	False
displayModelInOldFormat	False
doNotCjeckCapabilities	False
NumDecimalPlaces	4
useKernelEstimator	False
UseSupervisedDiscretization	False

(d) Logistic regression algorithm	
Logistic regression classifier	
Batch size	100
Debug	False

Table 9 (continued)

(d) Logistic regression algorithm	
Logistic regression classifier	
displayModelInOldFormat	False
doNotCjeckCapabilities	False
Maxits	− 1
NumDecimalPlaces	4
Ridge	1.0E−8
useConjugateGradientDecent	False

(e) BayesNet algorithm	
Bayes net classifier parameters	
Batch size	100
Debug	False
DoNotCheckCapabilities	False
Estimator	Simple Estimator-A 0.5
numDecimalPlaces	2
Search algorithm	K2-p1-S BAYES
useADTree	False

The content of Table 11 indicates that in the classification section, the best algorithm for detecting high-volume DDoS attacks in ISCX-SlowDDos-2016 dataset is the REPTree algorithm at 99.88% accuracy and 0.04% FPR volume. The evaluation results of the ISCX-IDS-2012 dataset revealed that the REPTree algorithm with 99.85% accuracy and 0.1% FPR volume is the best algorithm to detect high-volume DDoS attacks. As to the CTU-10 dataset, the J48 algorithm with an accuracy of 99.12% and 0.35% FPR volume is the best algorithm to detect high-volume DDoS attacks. As to CTU-11 dataset, the classification section results suggest that the logistic algorithm is of higher accuracy in high-volume DDoS attack detection with higher FPR volume. Between the two RandomTree and REPTree algorithms, the accuracy volume in the first is high, while to FPR volume the second is low.

The comparative diagram of the best results in high-volume attack detection for different datasets is shown in Fig. 9.

7.2 Low-volume DDoS attack detection results

The results of these proposed methods for different datasets are provided for low-volume DDoS attack detection. Both the static and dynamic threshold methods are applied to examine entropy-based section results.

Table 10 Results of *K*-fold method as to the high-volume attack detection are obtained by applying ISCX-SlowDDos2016

Step	MAE	RMSE	RAE	Correctly classified instance	Incorrectly classified instance	TPR	FPR
(a) Logistic algorithm							
1	0.0085	0.0533	16.23	523,577	1340	99.74	0.255
2	0.0102	0.0767	14.016	556,416	3824	99.31	0.682
3	0.0089	0.0504	13.596	539,411	1560	99.71	0.288
4	0.008	0.0517	12.606	522,638	1356	99.74	0.258
5	0.0095	0.0636	12.377	574,583	3459	99.40	0.598
6	0.0123	0.0752	19.598	512,399	3559	99.31	0.689
7	0.0089	0.0514	13.83	532,823	781	99.85	0.146
8	0.0083	0.0559	14.538	549,042	1893	99.65	0.343
9	0.0086	0.0617	11.564	534,508	1919	99.64	0.357
10	0.0067	0.0465	10.332	510,709	902	99.82	0.176
11	Average-step		TPR	FPR	ACC	Precision	*F*-Measure
			99.87	0.39	99.62	88.82	94.02
(b) J48 algorithm							
1	0.0007	0.0225	1.3435	524,631	286	99.94	0.054
2	0.0019	0.0413	2.6106	559,257	983	99.82	0.175
3	0.0008	0.0252	1.209	540,611	360	99.93	0.066
4	0.0006	0.0205	0.9502	523,753	241	99.95	0.046
5	0.0006	0.0196	0.7507	577,796	246	99.95	0.042
6	0.0049	0.0689	7.8996	51,371	2487	99.51	0.482
7	0.0023	0.046	3.4912	532,461	1143	99.78	0.214
8	0.0012	0.0315	2.0813	550,365	570	99.89	0.103
9	0.0011	0.0309	1.5207	535,897	530	99.90	0.098
10	0.0005	0.0189	0.7737	511,415	196	99.96	0.038
11	Average step		TPR	FPR	ACC	Precision	*F*-measure
			98.59	0.09	99.87	97.53	98.06
(c) BayesNet algorithm							
1	0.0078	0.0868	15.07	520,673	4244	99.19	0.808
2	0.0059	0.0753	8.09	556,963	3277	99.41	0.584
3	0.0077	0.0852	11.65	536,993	3978	99.26	0.735
4	0.0192	0.1374	30.21	513,900	10094	98.07	1.926
5	0.0031	0.0564	4.30	576,175	1867	99.67	0.323
6	0.0062	0.0769	9.94	512,772	3186	99.38	0.617
7	0.0039	0.0568	5.990	531,737	1867	99.65	0.349
8	0.0088	0.091	15.42	546,067	4868	99.11	0.883
9	0.0027	0.0513	3.68	534,954	1473	99.72	0.274
10	0.0023	0.0472	3.5549	510,412	1199	99.76	0.234
11	Average step		TPR	FPR	ACC	Precision	*F*-measure
			96.23	0.58	99.33	83.48	89.40
(d) RandomTree algorithm							
1	0.0056	0.0735	10.667	522,059	2858	99.45	0.544

Table 10 (continued)

Step	MAE	RMSE	RAE	Correctly classified instance	Incorrectly classified instance	TPR	FPR
2	0.0004	0.0174	0.6083	560,029	211	99.96	0.037
3	0.0007	0.024	1.0662	540,629	342	99.93	0.063
4	0.0004	0.017	0.6526	523,818	176	99.96	0.033
5	0.0015	0.0365	1.9033	577,250	792	99.86	0.137
6	0.0006	0.0228	1.0333	515,653	305	99.94	0.059
7	0.0005	0.0203	0.7817	533,373	231	99.95	0.043
8	0.0012	0.0334	2.192	550,229	636	99.88	0.115
9	0.0011	0.0313	1.4758	535,877	550	99.89	0.102
10	0.0005	0.0206	0.8003	511,380	231	99.95	0.045
11	Average step		TPR	FPR	ACC	Precision	*F*-measure
			98.01	0.05	99.88	98.53	98.27
(e) REPTree algorithm							
1	0.0054	0.0722	10.411	522,155	2762	99.47	0.526
2	0.0011	0.0306	1.5591	559,676	564	99.89	0.100
3	0.0005	0.0171	0.6987	540,790	181	99.96	0.033
4	0.001	0.0289	1.5754	523,528	466	99.91	0.088
5	0.0022	0.0446	2.8207	576,863	1179	99.79	0.204
6	0.0008	0.0243	1.2624	515,609	349	99.93	0.067
7	0.0003	0.0122	0.4253	533,507	97	99.98	0.018
8	0.0004	0.0139	0.6696	550,801	134	99.97	0.024
9	0.0006	0.0198	0.7546	536,184	243	99.95	0.045
10	0.0006	0.0221	0.9844	511,346	265	99.94	0.051
11	Average step		TPR	FPR	ACC	Precision	*F*-measure
			97.99	0.04	99.88	98.60	98.29

7.2.1 The results of the entropy-based section

These results together with the static threshold for low-volume DDoS attack detection are presented in Table 12.

As observed in Table 12, both the FPR and the TPR volumes are low, while due to the abundance of attacks in one or more specific time periods, the results are relatively better for short time periods.. The results of the entropy-based section for detecting low-volume DDoS attacks applying dynamic threshold are presented in Table 13.

The results in Table 13 indicate that the best time period for detecting low-volume attacks in the ISCX-SlowDDos-2016 dataset is 30 s at $C_d = 1$, as the best volume.

Table 11 Classification technique results for different datasets in different algorithms

Algorithms	TPR (%)	FPR (%)	ACC (%)	Precision (%)	*F*-measure (%)
ISCX-SlowDDos-2016					
BayesNet	96.23	0.58	99.33	83.48	89.40
J48	98.59	0.09	99.87	97.53	98.06
Logistic regression	99.87	0.39	99.62	88.82	94.02
RandomTree	98.01	0.05	99.88	98.53	98.27
REPTree	97.99	0.04	99.88	98.60	98.29
ISCX-IDS-2012					
J48	99.64	0.10	99.83	99.66	99.65
BayesNet	96.70	0.54	98.80	98.22	97.46
Logistic regression	96.69	4.00	96.14	86.47	91.29
Naive Bayes	90.76	2.58	95.84	91.56	91.16
RandomTree	99.65	0.12	99.82	99.60	99.62
REPTree	99.68	0.10	99.85	99.66	99.67
CTU-10					
J48	98.60	0.35	99.12	99.64	99.11
BayesNet	96.24	0.03	98.10	99.96	98.06
Logistic regression	97.41	0.62	98.39	99.38	97.89
Naive Bayes	96.14	0.11	98.01	99.88	97.97
RandomTree	98.07	0.48	98.79	99.51	99.51
REPTree	98.10	0.34	98.87	99.64	98.86
CTU-11					
J48	96.47	0	96.60	100	98.20
BayesNet	38.02	0	40.38	100	55.09
Logistic regression	99.99	9.20	99.64	99.63	99.80
Naive Bayes	99.08	14.49	98.57	99.42	99.24
RandomTree	98.88	0.61	98.88	99.97	99.42
REPTree	98.53	0.76	98.55	99.96	99.23

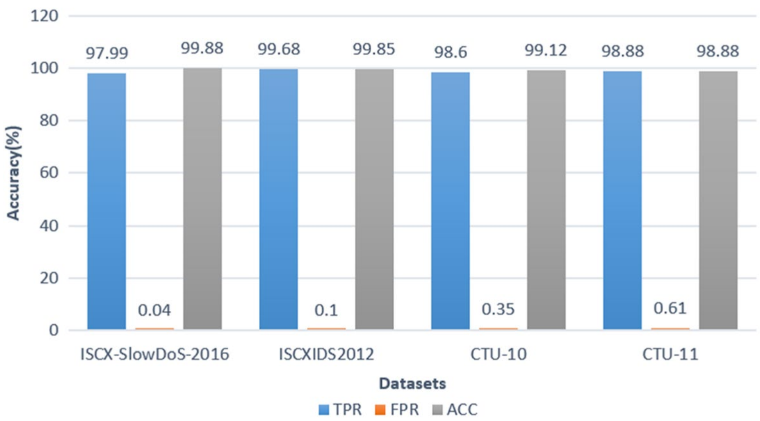


Fig. 9 The best results on attack detection in each dataset

Table 12 Evaluation results of static threshold in low-volume DDoS attack detection in the ISCX-SlowDDos-2016 dataset

Time period (s)	Entropy in attack	Entropy in normal	TPR (%)	FPR (%)	ACC (%)	Precision (%)	*F*-measure (%)
10	0.91	2.81	65.87	4.04	94.32	48.25	55.70
15	0.91	2.84	63.97	1.72	96.52	66.66	65.29
20	0.91	2.87	60.09	0.88	97.12	78.61	68.11
25	0.91	2.86	56.81	0.41	97.29	88.49	69.20
30	0.91	2.86	60	0.26	97.65	92.55	72.80
45	0.91	2.84	52.42	0.16	97.21	94.73	67.50

Table 13 Evaluation results of dynamic threshold in low-volume DDoS attack detection for the ISCX-SlowDDos-2016 dataset

Time period	C_d	Alarm rate (%)	TPR (%)	FPR (%)	ACC (%)	Precision (%)	*F*-measure (%)
15	-1	26.07	71.56	24.19	75.63	10.89	18.91
	0	55.71	91.46	54.23	47.58	6.51	12.16
	1.4	89.87	100	89.45	14.09	4.41	8.46
20	-1	28.67	81.01	26.55	73.73	10.98	19.34
	0	56.38	93.03	54.90	46.96	6.41	12.00
	1.1	84.07	100	83.43	19.81	4.62	8.84
25	-1	27.94	86.15	25.54	74.91	12.18	21.35
	0	56.12	93.84	54.57	47.33	6.60	12.34
	1.5	90.08	100	89.68	13.86	4.38	8.40
30	-1	28.86	87.50	26.56	73.96	11.40	20.17
	0	56.92	97.11	55.35	46.61	6.41	12.03
	1	82.56	100	35.52	68.27	25.24	40.31
35	-1	27.98	87.87	25.79	74.69	11.08	19.69
	0	56.92	98.48	55.40	46.49	6.10	11.50
	1.1	84.00	100	83.41	19.52	4.20	8.06

7.2.2 The results of classification algorithm

Similar to the approach in detecting high-volume attacks, the RandomTree, Logistic Regression, J48, BayesNet and REPTree classification algorithms and the *K*-fold method at $K = 10$, are involved in low-volume attack detection. The details of this process are presented in Table 14a–e.

In the mentioned tables, the parameters are calculated for different classifying algorithms and ISCX-SlowDDoS2016 dataset, results of which are given in Table 15.

As observed in Table 15, most classifying algorithms except the Naive Bayes have high accuracy in detection and low alarm rate, while the efficiency of this

Table 14 The *K*-fold method results for low-volume attack detection by applying ISCX-SlowDDoS2016 dataset

(a) REPTree algorithm

Step	Mean absolute error	Root-mean-squared error	Relative absolute error	Correctly classified instance	InCorrectly classified instance	TPR	FPR
1	0.0002	0.0098	0.8036	524,971	58	99.98	0.011
2	0.0007	0.025	3.0129	504,569	326	99.93	0.064
3	0.0002	0.011	0.8608	523,214	73	99.98	0.014
4	0.0004	0.018	1.7751	544,230	188	99.96	0.034
5	0.0002	0.0094	0.5599	517,863	53	99.98	0.010
6	0.0002	0.01	0.6937	537,723	57	99.98	0.010
7	0.0004	0.0164	1.4678	549,482	150	99.97	0.027
8	0.0025	0.0488	8.3219	519,459	1245	99.76	0.239
9	0.0003	0.0128	0.9439	500,961	89	99.98	0.017
10	0.0003	0.015	1.3528	558,772	129	99.97	0.023
11	Average step		TPR	FPR	ACC	Precision	*F*-measure
			99.44	0.04	99.96	97.15	98.28
(b) J48 algorithm							
1	0.0002	0.0125	1.0562	524,941	88	99.98	0.01
2	0.0008	0.0267	3.3768	504,527	368	99.92	0.072
3	0.0002	0.0129	0.9811	523,193	94	99.98	0.018
4	0.0006	0.0218	2.28	544,154	264	99.95	0.048
5	0.0002	0.0102	0.6228	517,856	60	99.98	0.011
6	0.0014	0.0364	5.3664	537,058	722	99.86	0.134
7	0.0007	0.0236	2.6054	549,322	310	99.94	0.056
8	0.0024	0.0484	8.1773	519,478	1226	99.76	0.235
9	0.0008	0.0269	3.0671	500,682	368	99.92	0.07
10	0.0004	0.018	1.744	558,717	184	99.96	0.032
11	Average step		TPR	FPR	ACC	Precision	*F*-measure
			99.15	0.060	99.93	95.52	97.30
(c) Random tree algorithm							
1	0.0007	0.0259	3.137	524,672	357	99.93	0.068
2	0.0012	0.0335	4.982	504,321	574	99.88	0.113
3	0.0002	0.0123	0.8536	523,198	89	99.98	0.017
4	0.0003	0.0137	1.0489	544,313	105	99.98	0.019
5	0.0006	0.0237	2.0096	517,621	295	99.94	0.057
6	0.0002	0.0116	0.7035	537,701	79	99.98	0.014
7	00004	0.0187	1.6852	549,435	197	99.96	0.035
8	0.0014	0.0372	4.8223	519,976	728	99.86	0.139
9	0.0005	0.022	2.0206	500,800	250	99.95	0.049
10	0.0003	0.0162	1.372	558,756	145	99.97	0.025
11	Average step		TPR	FPR	ACC	Precision	*F*-measure
			98.98	0.04	99.95	96.95	97.95

Table 14 (continued)

(a) REPTree algorithm

Step	Mean absolute error	Root-mean-squared error	Relative absolute error	Correctly classified instance	InCorrectly classified instance	TPR	FPR
(d) Logistic algorithm							
1	0.002	0.0223	8.7296	524,787	242	99.95	0.046
2	0.0053	0.0576	22.339	503,175	1720	99.65	0.340
3	0.0032	0.0316	12.574	522,681	606	99.88	0.115
4	0.0022	0.025	8.8104	544,095	323	99.94	0.0593
5	0.0047	0.0496	15.707	516,509	1407	99.72	0.2717
6	0.0028	0.0247	10.803	537,478	302	99.94	0.056
7	0.0037	0.0421	14.495	548,626	1006	99.81	0.183
8	0.0052	0.0535	17.507	518,942	1762	99.66	0.338
9	0.0044	0.0462	16.448	499,856	1194	99.76	0.238
10	0.006	0.0639	24.23	556,231	2670	99.52	0.477
11	Average step		TPR	FPR	ACC	Precision	*F*-measure
			96.32	0.17	99.79	87.19	91.53
(e) Bayes net algorithm							
1	0.0004	0.0177	1.6909	524,849	180	99.96	0.034
2	0.0032	0.0546	13.677	503,231	1664	99.67	0.329
3	0.0035	0.0566	13.911	521,536	1751	99.66	0.334
4	0.001	0.0289	4.069	543,890	528	99.90	0.097
5	0.0047	0.067	15.602	515,533	2383	99.53	0.460
6	0.0033	0.0555	12.733	536,048	1732	99.67	0.322
7	0.0029	0.0516	11.147	547,980	1652	99.69	0.300
8	0.002	0.0422	6.5925	519,611	1093	99.79	0.209
9	0.0035	0.0558	13.116	499,384	1666	99.66	0.332
10	0.0047	0.0684	19.121	556,266	2635	99.52	0.4715
11	Average step		TPR	FPR	ACC	Precision	*F*-measure
			87.46	0.12	99.71	91.09	89.24

Table 15 Results of the ISCX-SlowDDos-2016 dataset for low-volume DDoS attack detection

Algorithm	TPR (%)	FPR (%)	ACC (%)	Precision (%)	*F*-measure (%)
J48	99.15	0.06	99.93	95.52	97.30
BayesNet	87.46	0.12	99.71	91.09	89.24
Logistic regression	96.32	0.17	99.79	87.19	91.53
Naive Bayes	36.44	0.07	97.75	95.08	52.69
RandomTree	98.98	0.04	99.95	96.95	97.95
REPTree	99.44	0.04	99.96	97.15	98.28

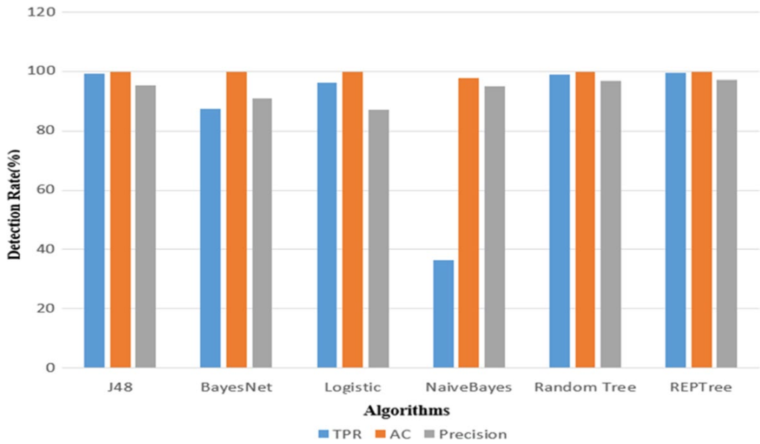


Fig. 10 Comparison of the ISCX-SlowDDos-2016 dataset results for low-volume DDoS attack detection

proposed model for low-volume attack detection is outstanding. The low-volume DDoS attack detection results for different classifying algorithms are bar-charted in Fig. 10 for comparison.

High accuracy and low FPR in all classifying algorithms shown in Fig. 10 reveal the high efficiency and quality of the features extracted through this proposed method in low-volume attack detection.

The results indicate that REPTree algorithm has the high accuracy of 99.96% and a low FPR value of 0.04% in detecting low-volume DDoS attacks.

8 Analysis of computational complexity and time cost of this proposed method

The method proposed here is a combination of entropy-based and classification method. Its computational complexity is derived from the combination of complexity of these two methods. The entropy is calculated through the entropy-based step, and the results are compared with the threshold where calculations are of O (n) computational complexity and n is the flow count. Assuming that the count of time period is d, the computational complexity here is calculated in Eq. (9):

$$\text{Computational-Complexity}_{\text{Entropy}} = O(d \times n) \tag{9}$$

where n is the flow count and d is the time period count. In the classification section, the complexity of calculating the flow features is of $O(n)$ order, while different classification algorithms have different computational complexities (e.g., decision tree algorithms have computational complexities in $O(\log n)$ order which leads to a total of $O(n) + O(\log n)$). Assuming that the count of time period is d, the computational complexity for classification step would be expressed in Eq. (10):

Table 16 Results of DDoS attack detection with different methods for the UNB-ISCX dataset

Authors	Technique	Ref.	ACC (%)	FPR
Warusia Yassin et al.	*K*-means + NBC	[12]	99	2.2%
Zhiyuan Tan et al.	Computer vision technique	[13]	90.12	7.92%
Alan Saied et al.	Neural network	[14]	98	Not mentioned
Bing Wang et al.	Cloud computing	[15]	89.30	Not mentioned
Naser Fallahi et al.	Ripper + C5.0	[18]	99	2%
Carlos Catania et al.	Machine learning	[38]	81.80	8.2%
Proposed method	Statistical method + machine learning		99.85	0.1%

Table 17 Results of DDoS attack detection with different methods for the CTU-13 dataset

Authors	Technique	Ref.	ACC (%)	TPR	Precision	*F*-measure
P. Kalaivani	REPTree + SVM	[39]	98.40	99.10%	98.40%	98.70%
Ankit Bansal et al.	RNN neural network	[40]	98.39	84.47%	86.45%	85.45%
Ruidong Chen et al.	RandomForest model	[41]	93.61	Not mentioned	Not mentioned	Not mentioned
Proposed method	Statistical method + machine learning		99.12	98.60%	99.64%	99.11%

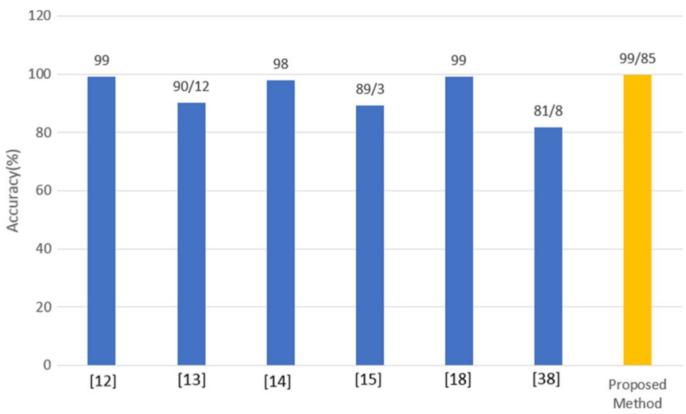


Fig. 11 Comparing the accuracy of this proposed method to other studies for the UNB-ISCX dataset

$$\text{Computational-Complexity}_{\text{Machine-Learning}} = O(d \times (n + \log\, n)) \tag{10}$$

Each one of these time periods has a constant coefficient, assumed as one in calculating the computational complexity. Now, by assuming that alarms are triggered

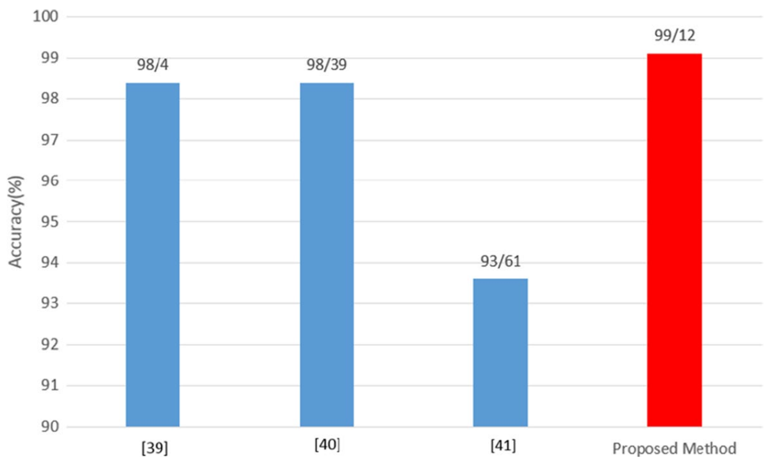


Fig. 12 Comparing the accuracy of this proposed method to other studies for the CTU-13 dataset

in $\frac{1}{k}$ periods, the computational complexity of an attack occurrence is calculated through Eq. (11):

$$\text{Computational-Complexity}_{\text{this-method}} = O\left(d \times n + \left(\frac{d}{k}\right) \times (n + \log n)\right) \quad (11)$$

where n is the flow count, d is the period count and $1/K$ is the count of periods considered as attacks.

9 Comparative performance experiments

In DDoS attack detections, the model is ranked as highly efficient which is of high accuracy and low FPR. The accuracy and FPR of the studies where the UNB-ISCX dataset is applied are shown in Table 16.

As observed in this table, this proposed method outperforms its counterpart at 99.85% accuracy and 0.1% FPR, where the error awareness level is lower than all mentioned methods. These results indicate the high efficiency of this proposed method. The three techniques applied in CTU-13 dataset for attack detection through this proposed method are compared to its counterparts in Table 17.

As observed in this table, an accuracy of 99.12%, the highest on CTU-13 dataset is obtained through this proposed method.

The content of Table 16 is reflected in bar-charts in Figs. 11 and 12

As observed in Figs. 11 and 12, the higher efficiency of this model for both the datasets in relation to the available methods is evident.

10 Conclusion and future works

SDNs are the latest in improving computer networks, due to their being flexible and reducing operational costs and providing security against DDoS attacks. To improve the security herein, a new method for detecting high-volume and low-volume DDoS attacks by applying a combination of statistical and machine learning techniques is proposed here. This method consists of the collector, entropy-based and classification sections.

This proposed method is evaluated and analyzed, and the findings indicate that the entropy-based sections with static threshold do not yield appropriate results according to experiments run on different datasets. The better results are obtained for the dynamic threshold at the cost of high FPR. To remove this drawback, different classification algorithms are run and more accurate results are obtained.

The significance of this method, as to accuracy, is its outperformance versus its counterparts. Results indicated that the accuracy of this proposed method is higher than other similar methods. Because this proposed model is to find solutions after attack event, the manner of DDoS attack prevention in SDN networks should be assessed. Though, in this article, the DDoS attacks are detected only by one controller in SDN, in the studies to come this method can be improved in networks by involving more than one controller.

References

1. Anithaashri T, Ravichandran G, Baskaran R (2019) Security enhancement for software defined network using game theoretical approach. Comput Netw 157:112–121
2. Todorova MS, Todorova ST (2016) DDoS attack detection in SDN-based VANET architectures. Master Appl Sci, 175
3. Behal S, Kumar K, Sachdeva M (2018) D-face: an anomaly based distributed approach for early detection of DDoS attacks and flash events. J Netw Comput Appl 111:49–63
4. Newman LH (2018) Github survived the biggest DDoS attack ever recorded. Wired, 1
5. Kupreev O, Badovskaya E, Gutnikov A (2019) DDoS attacks in Q1 2019
6. Hoque N, Kashyap H, Bhattacharyya DK (2017) Real-time ddos attack detection using FPGA. Comput Commun 110:48–58
7. Dayal N, Maity P, Srivastava S, Khondoker R (2016) Research trends in security and DDoS in SDN. Secur Commun Netw 9(18):6386–6411
8. Salloum SA, Alshurideh M, Elnagar A, Shaalan K (2020) Machine learning and deep learning techniques for cybersecurity: a review. In: Joint European-US workshop on applications of invariance in computer vision. Springer, pp 50–57
9. Prasad KM, Siva VS, Nagamuneiah J, Nelaballi S (2020) An ensemble framework for flow-based application layer DDoS attack detection using data mining techniques. In: ICT analysis and applications. Springer, pp 9–19
10. Chen W, Xiao S, Liu L, Jiang X, Tang Z (2020) A DDoS attacks traceback scheme for SDN-based smart city. Comput Electr Eng 81:106503
11. Agrawal N, Tapaswi S (2018) Low rate cloud DDoS attack defense method based on power spectral density analysis. Inf Process Lett 138:44–50
12. Yassin W, Udzir NI, Muda Z, Sulaiman MN et al (2013) Anomaly-based intrusion detection through k-means clustering and Naives Bayes classification. In: Proceedings of the 4th International Conference on Computer Informatics ICOCI
13. Tan Z, Jamdagni A, He X, Nanda P, Liu RP, Hu J (2014) Detection of denial-of-service attacks based on computer vision techniques. IEEE Trans Comput 64(9):2519–2533

14. Saied A, Overill RE, Radzik T (2016) Detection of known and unknown DDoS attacks using artificial neural networks. Neurocomputing 172:385–393
15. Wang B, Zheng Y, Lou W, Hou YT (2015) DDoS attack protection in the era of cloud computing and software-defined networking. Comput Netw 81:308–319
16. Yan Q, Gong Q, Deng F-A (2016) Detection of DDoS attacks against wireless SDN controllers based on the fuzzy synthetic evaluation decision-making model. Adhoc Sens Wirel Netw 33
17. Cui Y, Yan L, Li S, Xing H, Pan W, Zhu J, Zheng X (2016) SD-Anti-DDoS: fast and efficient DDoS defense in software-defined networks. J Netw Comput Appl 68:65–79
18. Fallahi N, Sami A, Tajbakhsh M (2016) Automated flow-based rule generation for network intrusion detection systems. In: 24th Iranian Conference on Electrical Engineering (ICEE). IEEE, pp 1948–1953
19. Liang X, Znati T (2019) On the performance of intelligent techniques for intensive and stealthy DDoS detection. Comput Netw 164:106906
20. Ujjan RMA, Pervez Z, Dahal K, Bashir AK, Mumtaz R, González J (2019) Towards sFlow and adaptive polling sampling for deep learning based DDoS detection in SDN. Future Gener Comput Syst
21. Ferrag MA, Maglaras L, Moschoyiannis S, Janicke H (2020) Deep learning for cyber security intrusion detection: approaches, datasets, and comparative study. J Inf Secur Appl 50:102419
22. Wang M, Lu Y, Qin J (2020) A dynamic MLP-based DDoS attack detection method using feature selection and feedback. Comput Secur 88:101645
23. Asadollahi S, Goswami B (2017) Experimenting with scalability of floodlight controller in software defined networks. In: International Conference on Electrical, Electronics, Communication, Computer, and Optimization Techniques (ICEECCOT). IEEE, pp 288–292
24. Shaghaghi A, Kaafar MA, Buyya R, Jha S (2020) Software-defined network (SDN) data plane security: issues, solutions, and future directions. In: Handbook of Computer Networks and Cyber Security. Springer, pp 341–387
25. Dai Y, He J, Wu Y, Chen S, Shang P (2019) Generalized entropy plane based on permutation entropy and distribution entropy analysis for complex time series. Physica A 520:217–231
26. Oshima S, Nakashima T, Sueyoshi T (2010) DDoS detection technique using statistical analysis to generate quick response time. In: International Conference on Broadband, Wireless Computing, Communication and Applications. IEEE, pp 672–677
27. Azeez N, Babatope A (2016) AANtiD: an alternative approach to network intrusion detection. J Comput Sci Appl 23(1):129–143
28. Thomas T, Vijayaraghavan AP, Emmanuel S (2020) Introduction to machine learning. In: Machine learning approaches in cyber security analytics. Springer, pp 17–36
29. Xiong Z, Cui Y, Liu Z, Zhao Y, Hu M, Hu J (2020) Evaluating explorative prediction power of machine learning algorithms for materials discovery using k-fold forward cross-validation. Comput Mater Sci 171:109203
30. Jazi HH, Gonzalez H, Stakhanova N, Ghorbani AA (2017) Detecting HTTP-based application layer DoS attacks on web servers in the presence of sampling. Comput Netw 121:25–36
31. Yavanoglu O, Aydos M (2017) A review on cyber security datasets for machine learning algorithms. In: IEEE International Conference on Big Data (Big Data). IEEE, pp 2186–2193
32. Bhamare D, Salman T, Samaka M, Erbad A, Jain R (2016) Feasibility of supervised machine learning for cloud security. In: International Conference on Information Science and Security (ICISS). IEEE, pp 1–5
33. Zollanvari A, Dougherty ER (2014) Moments and root-mean-square error of the Bayesian MMSE estimator of classification error in the gaussian model. Pattern Recogn 47(6):2178–2192
34. Al-Ayyoub M, Jararweh Y, Benkhelifa E, Vouk M, Rindos A et al (2017) A novel framework for software defined based secure storage systems. Simul Model Pract Theory 77:407–423
35. Abbott D (2011) Linux for embedded and real-time applications. Elsevier, Amsterdam
36. Izard R (2020) Floodlight controller. https://floodlight.atlassian.net/wiki/spaces/floodlightcontroller
37. Fernandes G, Rodrigues JJ, Carvalho LF, Al-Muhtadi JF, Proença ML (2019) A comprehensive survey on network anomaly detection. Telecommun Syst 70(3):447–489
38. Catania C, Garino CG (2013) Towards reducing human effort in network intrusion detection. In: 2013 IEEE 7th International Conference on Intelligent Data Acquisition and Advanced Computing Systems (IDAACS), vol 2. IEEE, pp 655–660
39. Kalaivani P, Vijaya M (2016) Mining based detection of botnet traffic in network flow. Int J Comput Sci Inf Technol Secur 6:535–540

40. Bansal A, Mahapatra S (2017) A comparative analysis of machine learning techniques for botnet detection. In: Proceedings of the 10th International Conference on Security of Information and Networks, pp 91–98
41. Chen R, Niu W, Zhang X, Zhuo Z, Lv F (2017) An effective conversation-based botnet detection method. Math Probl Eng 2017

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.