Journal of Computer Science

Original Research Paper

DDoS Attack Detection Using Enhanced Neural Network Algorithm in Software-Defined Networks

[1]Hema Surendrakumar Dhadhal and [2]Paresh P Kotak

[1]Department of Information Technology, Lukhdhirji Engineering College, Sama Kanthe, Morbi, India
[2]Principal Dr S & SS Ghandhy College of Engineering and Technology, Surat, India

Article history
Received: 13-05-2022
Revised: 27-08-2022
Accepted: 26-09-2022

Corresponding Author:
Hema Surendrakumar Dhadhal
Department of Information Technology, Lukhdhirji Engineering College, Sama Kanthe, Morbi, India
Email: hemadhadhal@gmail.com

Abstract: With the uncommon advancement of technology and networking, the world today is totally relying on it. The most obvious drawback of this technical progress resulted in several possible dangers. A Distributed Denial of Service (DDoS) attack is one in which a large number of compromised systems work together to prevent service from being provided to the targeted system. Consequently, to protect network servers, attack detection systems must be very effective. The proposed architecture has Software Defined Networks (SDN) which comprise controllers and SFlow agents. According to the article, the anomaly detection of statistical traffic, which is performed on both normal and pathological anomaly traces in the packet header, as well as traffic volume detection, is based on the suggested work provided in this study. The k-means clustering technique is used by the statistical anomaly detection system and the attack alert aggregation system. Low-level attack detection systems are used to generate cluster dissimilar warnings, which are then investigated further. Clusters generate meta-alerts based on the information they have gathered. After that, the report of meta-alerts is sent to the security specialists. The online alert aggregation technique, which is also known as the probabilistic model, is used to identify new assaults on a system. The k-means clustering method is used to improve the quality of the traffic data streams. The Enhanced Neural Network Algorithm (ENNA) is being used to advance the intelligent attack detection system, which is currently under development. It is utilized in open-day controllers to identify attacks with a 98.7% accuracy by using Mininet and the Python simulation tool (ODL). In future work, it is possible to evaluate how the suggested detection method would be used in the event of subnet attacks and its mitigation.

Keywords: Distributed Denial of Service, Open Day Controller (ODL), Modified K-Means Clustering, Enhanced Neural Network Algorithm, Traffic Volume Detection

Introduction

Because of technical advancements, individuals have come to depend on the internet for almost every activity in today's digital world. However, the same technology is being abused by attackers to obtain illegal access to the network and the devices linked to the internet, to acquire information, or to bring the network down. According to data Xu *et al.* (2019), the pace of network attack growth has been rising over the past 10 years. The following common risk factors may be used by attackers to breach our network's security. Aydin (2019) study of the past five-year data, indicates that, out of the top five attacks, Denial of Service (DoS) and Man in The Middle (MiTM) attacks are the most prevalent in conventional networks. DoS attacks often cause network capacity to be reduced or user services to be interrupted (Rghioui *et al*., 2014). A flood attack is one of its categories and it is more susceptible to causing a device or network bandwidth to become unavailable. While the network gets an excessive amount of traffic to the server, flooding attacks have long been a common component in an attacker's toolkit for overwhelming service and ultimately halting the system (Zargar *et al*., 2013). To create a connection between the devices and the server, the standard TCP connection uses three-way handshakes to establish a link between them. To send a TCP SYN flood, the targeted server's connection sequence must be known. It never completes the handshake, instead leaving the connected port active and unavailable for any further requests to be made of it. Our contributions: While many researchers are concerned about SDN security (Iqbal *et al*., 2020) in general, there has been little development in the industry, except for two major

open-source controllers, ODL and Open Network Operating Systems (ONOS). To make the SDN environment safer, this article attempts to intelligently identify and prevent Distributed Denial of Service (DDoS) floods and IP spoofing attacks. The goal is to make the SDN environment more secure. To minimize the controller overhead, we suggested an integrated SFlow monitoring controller with the OpenFlow controller, which was successfully implemented. When a proactive routing method is used in a controller instead of the usual reactive routing technique, it reduces the amount of communication that occurs. Finally, learning-based models are used to identify and mitigate threats in an intelligent manner. So, the major outcome of the proposed work is:

i) By using the hybrid k-means clustering with an enhanced neural network algorithm better performance can be achieved in the detection of a DDOS attack with decreasing processing loads and times in a promising manner
ii) The proposed model is compared with the existing models and the accuracy is achieved at 98.6% for DDoS detection
iii) The false positive rate is decreased to up to 6% only, also the precision, and recall rate is improved as compared to the existing models
iv) Also, the complexity and CPU utilization are reduced to a great extent as described and compared in the experimental results section

The SDN controller is the network's centralized entity that manages flow rules and traffic management. As a result, choosing the appropriate controller is critical for improved network performance. The learning-based methods improve the detection model's performance by learning various states and classifying normal and abnormal traffic flows in the network (Kasim, 2020). Machine learning and deep learning-based methods were suggested in several research publications to protect SDN against DDoS attacks. In this part, a few ideas are addressed and summarized in Table 1. To identify DDoS assaults in SDN, (Li and Lee, 2005) developed a deep learning-based approach. This model uses Convolution Neural Network (CNN), Recurrent Neural Network (RNN), and Long Short-Term Memory (LSTM) and it has input, recursive, hidden, and output layers. Based on the detection, the controller creates the drop policy and sends it to the switches. The raw network traffic data is processed by the feature selection module before being sent to the deep learning detection module. Bidirectional RNN is used in the deep learning module to detect anomalous network events. Both the hidden and output layers utilize the non-linear tanh activation function. This model has a 96% accuracy level; however, the training time is excessively long due to hardware and software dependencies. Mehr and Ramamurthy (2019) presented a machine learning-based DDoS attack detection SVM model. Network traffic packet-in messages is gathered for the training procedure. Before using the Support Vector Machine (SVM) machine learning model (Jalili *et al*., 2005), pick suitable features from the acquired data to train the data. The entropy value for the selected characteristics was computed before the learning process. It improves the model's precision. The SVM model is compared to other machine learning algorithms to determine its efficiency. In the Ryu SDN controller, the study indicates that SVM is a superior approach for DDoS detection. In their proposal, (Ujjan *et al.*, 2020) used a stacked autoencoder deep learning-based DDoS assault detection model. The data for the detection model was gathered in two ways: Packet-based data via SFlow sampling and time-based data from the adaptive polling technique. It lowers network complexity and controls plane overhead. Snort IDS and SAE deep learning models were employed in a detection module to enhance the model's accuracy. The sampling technique based on SFlow has a lower false rate than the sampling method based on polling. Although the SAE-based deep learning model has a shorter detection time and a simpler control plane, its accuracy is only 95%.

Table 1: Various mitigation methods against DDoS attacks for SDN

References. no	DDoS attack mechanism	Type of tool and SDN controller used		Experimental limitations
Ujjan *et al.* (2021)	Intrusion is detected based on entropy calculated on different	Open flow switch table entries attack	Mininet and Ryu	Fixed NIDS
Elsayed *et al*. (2020)	RNN and autoencoder method combination for DDoS detection	UDP, SNMP, NetBIOS, SYN,	Mininet, and POX	Only limited to the detection mechanism
de Assis *et al*. (2020)	CNN for DDoS detection	DDoS attacks	Mininet and flood light also, CicDDoS 2019	Data tested on a limited number of hosts so by increasing the hosts the result may vary
Macedo *et al*. (2016)	PATMOS	DDoS attacks	Mininet and POX	The problem with using clustering
Hu *et al*. (2017)	Traffic flow migration	DDoS attacks	Mininet and Ryu	Consistency issue may occur in case of multiple controllers environment
Kalliola *et al*. (2015)	Network traffic rule based DDoS mitigation	TCP-SYN traffic flooding attack	Mininet, ODL	May increase the consumption of bandwidth and thus increasing the overhead as well
Shin *et al*. (2013)	Avant-guard	TCP-SYN	Mininet, POX on Traffic flooding	For the protocols which are not based on TCP-SYN traffic

Ma *et al*. (2020), a deep learning-based CNN model for detecting DDoS was suggested. The CNN algorithm comes with a standard feature picker that identifies features in the input data and maps them to a feature map. The output feature map was imported into the activation map, which was then decreased in size for the following phase. The final ANN detection phase uses the compressed activation map as an input. In this suggested model, two Keras-based ensemble CNN classifiers merge for activation function to identify DDoS attacks. The model's experimental results demonstrate that a CNN-based model can achieve excellent accuracy in a short amount of time with little computing cost. A deep CNN deep learning-based detection algorithm was suggested by Haider *et al*. (2020). The model's architecture includes four deep-learning algorithms: CNN, RNN, LSTM, and RL. To create a hybrid detection model, the first model combines CNN, RNN, and LSTM, while the second model combines RNN and LSTM, similar models. In this model, the activation functions Relu and sigmoid are employed and the activation functions activate the network's neurons. For the large-scale network system, the deep learning-based detection model guarantees efficiency. It's a low-overhead mitigation model that employs three distinct innovative methods. To begin, the table miss method is used to reduce communication bandwidth usage. Second, to conserve computing resources, the packet filtering method is utilized to filter out attack traffics. Finally, flow rules are maintained in a flow table to remove any unnecessary flow rules. To enhance the accuracy of the model, an SVM-based classifier is employed. The learning-based approach guarantees a shorter detection time while maintaining excellent accuracy. Many have proposed different clustering techniques for detecting DDoS assaults, even though current data indicate a rapid rise in DDoS attacks in recent years (Velliangiri and Premalatha, 2019).

Proposed SDN Architecture

An OpenFlow-based controller is the conventional SDN controller. The typical OpenFlow protocol-based architecture is shown in Fig. 1, along with its main duties to:

- Manage the flow of information
- Manage the apps

The flow control between switches is handled through a southbound API that monitors and generates flow rules for the physical layer. The suggested approach combines SFlow with an OpenFlow-based controller to decrease workload and enhance SDN controller efficiency. The OpenFlow-based controller will create flow rules while SFlow monitors the network and collects the required statistical data. It is possible to utilize SFlow data for attack detection and troubleshooting network-related issues. It is a method for monitoring, collecting, storing, and analyzing scalable network traffic. Specifically, it allows for the monitoring of tens of thousands of interfaces from a single place. Furthermore, it monitors connection speeds of up to 10 Gb/s and beyond without degrading the performance of core internet routers and switches or adding any major network burden to the system. The model has three modules such as The SFlow collector collects and analyses the traffic information and the logs are sent on to the cumulative sum detection model for further analysis.

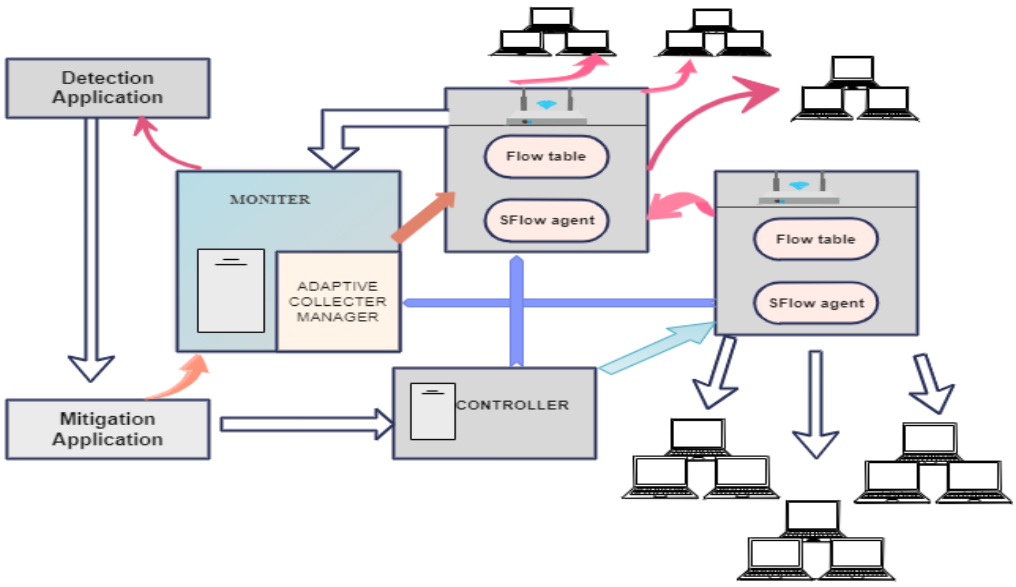

Fig. 1: Proposed SDN architecture

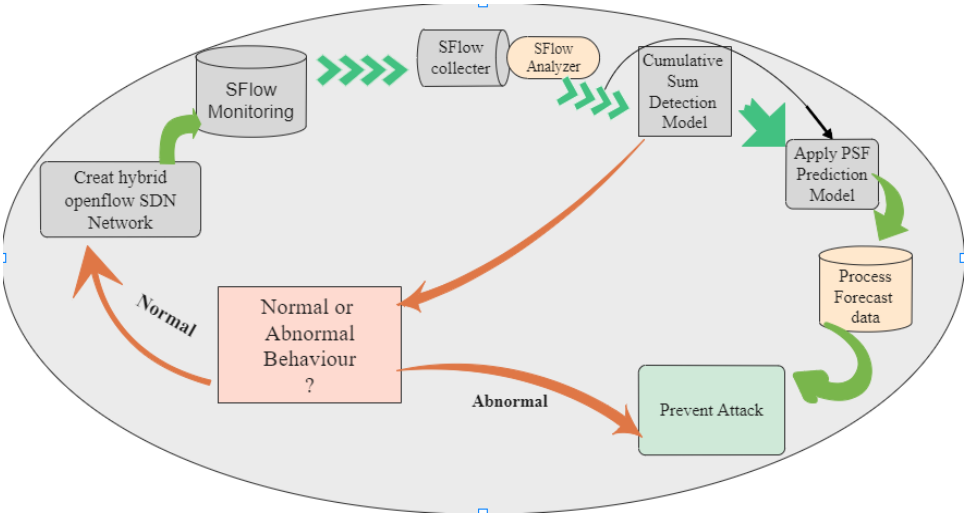

Fig. 2: IP Spoofing attack detection and prevention model

The detection model will determine if the traffic from a node pattern is normal or abnormal based on the pattern's characteristics. If everything is normal, the SFlow monitor will continue to measure the flow rate indefinitely.

Figure 2 represents the IP spoofing attack detection and prevention model. The Port ID (PID) of the aberrant node is detected and destroyed if this does not occur.

The information about the node is kept and tagged as an attacker. In parallel, the Pattern Sequence-based Forecasting (PSF) prediction model is being implemented to anticipate and prevent the assault before it occurs.

Monitoring, Detection and Prevention

The establishment of network topology is based on the suggested hybrid architecture to monitor the whole network. The SFlow adaptive collector manager, which is located in the control plane, is responsible for collecting and analyzing statistical data. In the analysis module, the statistical information is transmitted from SFlow-enabled switches to SFlow collectors; the analyzed samples are saved in a log for later use by the detection module Samples are collected for both normal and pathological network activity, and historical information is recorded in an analyzer along with a label indicating the network's behavior. The entire sampling and polling rates are started and the packet analyzer will wait for a packet to be received. It counts the packet in messages till it meets the sample count that has been specified.

Based on the sample sent from the monitoring process, the cumulative-SUM anomaly detection (clustering) model is developed to identify an abnormal node in a network and alert the network administrator. Then, with the assistance of the SFlow analyzer, the PID of the discovered node is determined, allowing the attacker node to be killed and the network from being attacked.

With the assistance of past network information, a forecasting model is being built in tandem to anticipate and prevent the assault before it occurs, in our forecasting module, we have implemented the Pattern Sequence-based Forecasting (PSF) prediction method. In general, the PSF model may be broken down into two processes: The first is the preprocessing and clustering of historical data and the second is the forecasting of future data based on clustered information.

DDoS Flooding Attack Detection

Figure 4 depicts the suggested approach for detecting and mitigating DDoS flooding attacks. The hybrid network topology started with the proposed detection model and the proactive attack-aware routing model is based on our proposed architecture. According to the suggested hybrid design, once the network topology is established, three processes will begin running in parallel at the same time. The SFlow collector begins collecting network statistics information and sends the information to the feature selection model, which then makes use of the information. An attack-aware load-balanced flow rule generator is used in the second phase, which is sent to the controller through a proactive routing protocol and then implemented.

Feature attribute selection is done from the input dataset. The input dataset is collected from the Canadian Institute for Cybersecurity. CICDoS2019 includes the most up-to-date and benign frequent DDoS assaults, which closely match the real-world data (PCAPs). The findings of a network traffic analysis using CICFlowMeter-V3 with labeled flows based on the time stamp, source, and destination IP addresses, source and destination ports, protocols, and attack methods are also included. The data logs collected from

SFlow collection will optimize and improve the accuracy of the attack detection model. The attribute selection process is done before the detection. To reduce the dimensionality of the SFlow collector's network traffic dataset and to select significant attributes, four different meta-heuristic attribute selection algorithms were tested and evaluated.

Binary Particle Swarm Optimization (BPSO) in Fig. 3 is a filter-based approach, which decides attributes based on their relevance and redundancy. This model can be applied to both discrete and continuous optimization problems. It is considered a tesseract; the element moves closer or corners of the tesseract by shifting the bits. The optimization state is limited to 0-1. BPSO is implemented with a sigmoid function, hence the probability of finding values nearer to 0 will not be perfect, because in some cases they will be considered as 0 in some other cases as 1. Due to this conflict attribute selection will not be perfect.

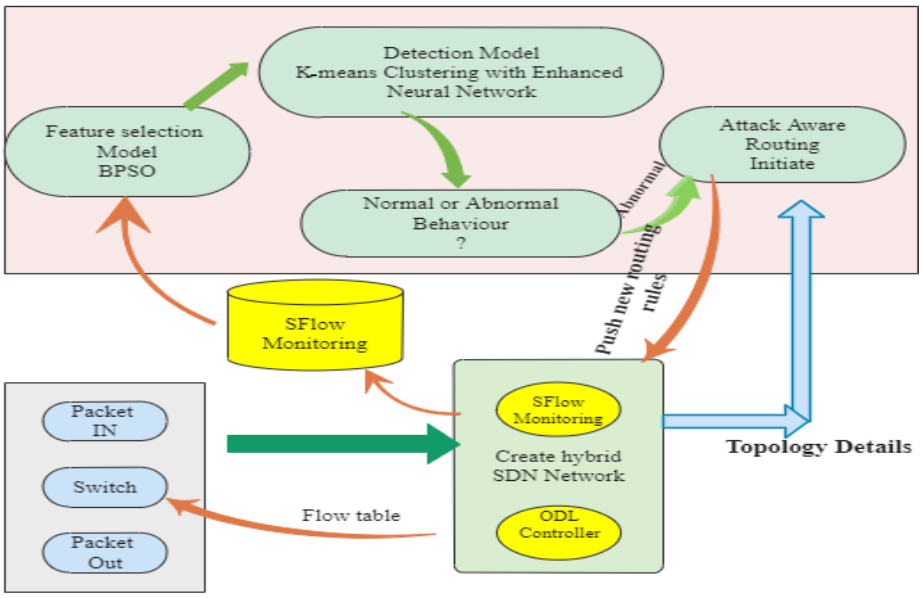

Fig. 3: Proposed methodology with k-means clustering and ENNA classifier

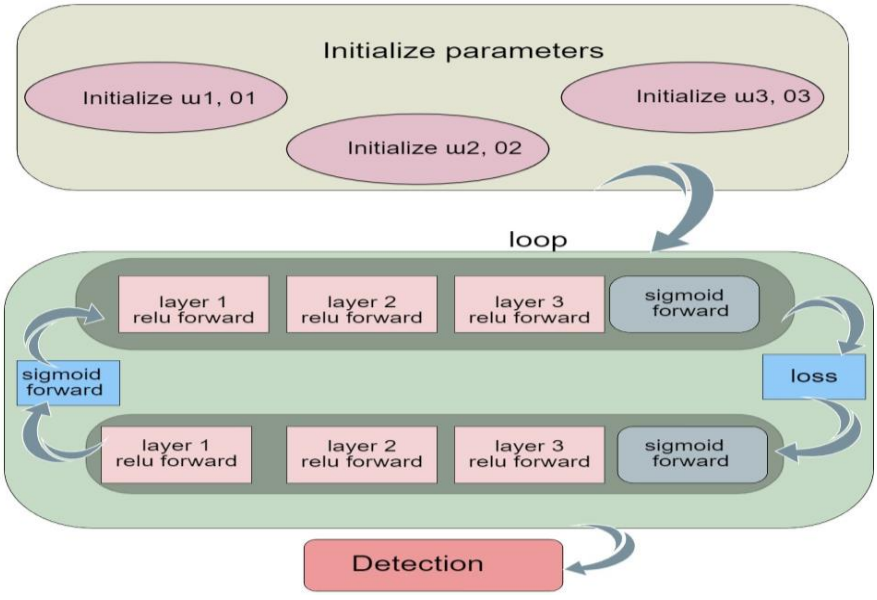

Fig. 4: Proposed ENN detection model

Intelligent Detection

The proposed Enhanced Neural Network (ENN) model consists of three hidden layers and one output layer enabled with the activation function. We implemented both forward and backward propagation with a decent gradient cost. Figure 4 shows the model of the proposed ENN detection model. The model with three hidden layers contains linear and Relu activation functions. One output layer includes the linear and sigmoid activation function in both forward and backward propagation. The cost is calculated for the loss value, which is given as input to the backpropagation; then, it will update the parameters. The newly updated parameters again initiate the forward propagation, looping this entire process until it reaches the global minimum value of gradient descent to improve our detection model's accuracy level.

The ENN model is designed with five layers including input, hidden and output layers, the entities of each layer are indicated by $n^{[l]}$, and the activation function of all layers is indicated by $a^{[l]}$.

Based on link bandwidth and the number of traffics, the link cost is calculated, accumulating the value of all links so for the particular path is the weight. Weight is calculated for all the generated possible paths. The lowest weighted paths are considered the best path on which any attacker node is present then the path is filtered out from the best path. Calculated new flow rules are pushed to the controller to update flow rules in a flow table. Based on network traffic, new flow rules are generated frequently. The switches will react based on the updated flow rules for the client request.

During the detection phase as the packets arrive, they are evaluated and assigned a score point based on the degree of relevance to attack characteristics possessed by them. This information is fed back to the set of attribute trees that are used for classifying the traffic. This positive feedback aids in fine-tuning the classifier to more correctly classify the traffic. The 86 packets that score above a predefined threshold value are suspected to be attack packets and are fed to the next level of classification. The cluster subsystem receives the suspected packets from the decision tree subsystem and classifies the packets based on the clustering index where they fit in. The attack packets are trained to get clustered separately for different types of DDoS attacks. In the proposed method, a semi-supervised k-means clustering technique is adopted to improve the clustering purity. The k-means algorithm assigns each multidimensional point to the cluster whose center (centroid) is nearest. The center is the average of all the points in the cluster, i.e., its coordinates are the arithmetic mean for each dimension separately over all the points in the cluster. The attack traffic is initially modeled as an array of trees, each of which stores and updates the data for a promising attribute of the packet toward efficient attack detection by mining the stored information dynamically. The trees are named attribute trees as they are used for manipulating the packet attributes and are structured as binary search trees and are populated with attribute values to aid in the effective detection of attack traffic. During the second phase of detection, the packets are modeled as objects in the appropriate clusters, and legitimacy is based on the degree of proximity to the matching pre-defined cluster relationships. The distance between the object *Obj* and the cluster head where it will be positioned is to be minimized and the average distance between the object *Obj* and other cluster heads is to be maximized when collisions occur to reduce the computation overhead involved in classifying an element, each cluster has one of its members selected as a cluster head. The cluster head is the element that represents the characteristics of its cluster elements. The proximity of a new member is determined based on the cluster head characteristics rather than considering all the members in the cluster.

Each packet is considered as an object '*Obj*' with k attributes. In a cluster of objects with similar characteristics, an object whose attributes lie closer to the respective attributes of the object that represents the mean value Obj_{mean}' is selected as the representative of the group called cluster head. To distinguish the contributory attributes among others, the cluster head periodically retrieves the weight values of the attributes computed by the decision tree subsystem and uses these values while classifying the incoming object. $'Obj_{mean}'$ may not be an existing object in the group and hence an object nearer to the mean object is selected as a representative object. The cluster head selection function is the weighted sum of the differences between two objects concerning their attributes as defined in Eq. (1). The cluster head is the object Obj_i whose difference $d(Obj_i, Obj_{mean})$ is minimum:

$$d\left(Obj_i, Obj_{mean}\right)=\sum_{1=1}^{k} W_1 \times \left|Obj_{i1}-Obj_{mean1}\right|, \text{ where } \sum W_1 = 1 \quad (1)$$

The following steps explain the cluster formation with a training set of packets:

Step 1: Select a subset of packets randomly from each of the sets of various DDoS attacks and normal traffic and initialize as many different clusters
Step 2: For every cluster compute the cluster heads as defined in Eq. (1)
Step 3: Place the next set of packets in the corresponding clusters and repeat step 2 to update the set of heads
Step 4: Repeat steps 2 and 3 until the heads stabilize or until a maximum number of iterations so that the selected heads always act as representative objects during attack classification

It demands the distance between the object *Obj* and the cluster head where it will be positioned, to be minimized and the average distance between the object *Obj* and other cluster heads to be maximized when collisions occur. Let the number of clusters be '*p*' and the cluster heads are defined as the set:

$$C=\{C1, C2,....CP\} \quad (2)$$

Choose the cluster *Cx* to satisfy the Eq. (3) to fit object *Obj* in cluster *Cx*:

$$CX \in C, \ni : d(Obj, CX) = \min_{Ci \in C} \left(d(Obj, Ci) \right) \quad (3)$$

If more than one cluster head is selected based on Eq. (3), then it is resolved using Eq. (4) fit *Obj* in:

$$CX \in C, \ni : d(Obj, CX) = \max \left\{ 1/\ p-1 \times \sum_{C \in C-(CX)} \left(d(Obj, Ci) \right) \right\} \quad (4)$$

The cluster where the packet is mapped determines the legitimacy or illegitimacy of the packet as the clusters are formed based on attack distinguishable characteristics.

Results

The network topology is emulated in a Mininet emulator with physical switches. Shell scripts are used to generate attack traffic and D-ITG traffic generators are used to test the performance of the network hosts during the emulation. Python is the programming language that is used to implement these algorithms. Our experimental setup tools are listed in Table 2.

The proposed hybrid architecture reduces system complexity by reducing CPU utilization and increasing scalability. The following graphs, Fig. 5(a), show the average CPU utilization percentage of the controller, and Fig. 5(b) shows in scalability point of view. Hybrid architecture reduces CPU utilization from more than 80% to less than 20%. Also, from the scalability point of view, the CPU utilization percentage remains less than 20%, even if we increase the switch's number.

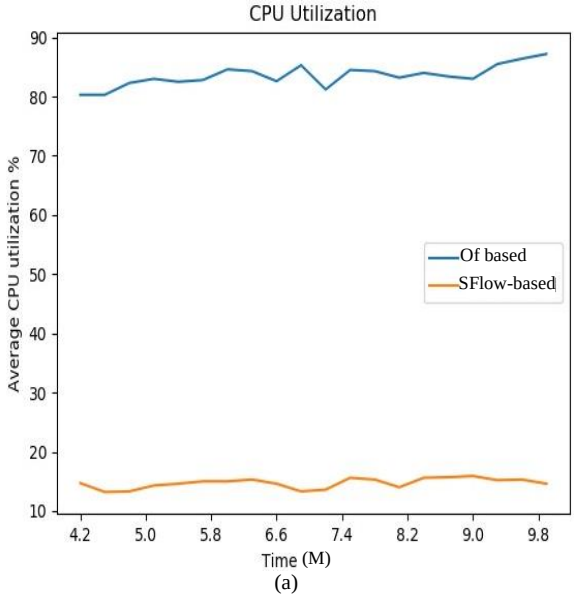

Fig. 5: Computational resource consumption of controllers

Table 2: Experimental setup tool

Tools	Name	Description
Emulator	Mininet 2.2.1	This emulator facilitates to test of complex topology without any physical devices and also integrates with the real network
Controllers	Open daylight and SFlow	Open Day Light (ODL) is an open-source java-based controller by the Linux foundation SFlow is a monitoring controller
Switches	OpenvSwitch2.11.0, HP3810M, HP2930F	OpenCV Switch is a multilayer software apache-licensed open-source switch HP3810M and HP2930F are the physical switches
Protocols	OpenFlow 1.3 and BGP	OpenFlow (OF) is a communication protocol it helps SDN controllers easily interact with the forwarding plane. Border Gateway Protocol (BGP) inter-domain routing protocol
Traffic generator	D-ITG 2.8.1	Distributed traffic generator to test the performance of the network
Attack generator	Shell script for IP spoofing. Shell script with hping3 for flooding	Hping3 is the free packet generator to introduce flooding attacks hoping is used in our script to emulate abnormal traffic
Algorithms	Python3.7	Python is an open-source lightweight programming language

Algorithms are implemented with the NumPy vectorization to reduce the time complexity of code running time in the proposed models. The time complexity of the vectorization method is $O(n)$, where n is the number of iterations. The default looping approach time complexity is $O(n*s*m)$, where n is the iterations, s is the data-set sample number, and m is the data-set feature number.

The vectorization approach cost 1.75 sec while the for-loop costs 525 sec. The vectorization approach is 300 times faster than the looping approach.

IP Spoofing Attack Prevention Results

Performance metrics are the major criteria to prove the algorithm with a better outcome. The performance metrics are classified as accuracy, sensitivity, false negative rate, false positive rate, precision, and recall rate. The calculation of the performance metrics is as follows:

$$Accuracy = \frac{True\ positive\ + True\ negative}{True\ positive + True\ negative + \ False\ Positive\ + \ False\ Negative} \quad (5)$$

$$Sensitivity = \frac{True\ positive}{True\ Positive + False\ Negative} \quad (6)$$

$$False\ Positive\ Rate = \frac{False\ \ positive}{False\ positive + True\ \ Negative} \quad (7)$$

$$False\ \ Negative\ \ Rate = \frac{False\ \ Negative}{False\ \ Negative + True\ \ Positive} \quad (8)$$

$$Precision\ Rate = \frac{True\ \ positive}{False\ \ Positive + False\ \ Positive} \quad (9)$$

The detection accuracy of the proposed clustering model in different test cases with some abnormal situations is tabulated in Table 3.

As time goes the number of samples getting increased and the accuracy is also increased to 100%. The false-positive rate is less than 6%.

The accuracy percentage is calculated based on the ratio between the total number of abnormalities and the total abnormal detected.

In parallel with the help of historical network information. We compared three different forecasting models, such as PSF, ARIMA, and ETS. The following Fig. 6 shows the performance of these models based on the original value and its predicted values.

Mean Absolute Error (MAE)

MAE is a measure of errors between paired observations and it is calculated based on original and predicted values with its number of samples. The following Table 4 shows the calculation of MAE. Compared to other algorithms, PSF is less in RMSE and MAE for cross-validation with different cases. The PSF model values clearly state that this model outperforms other forecasting methods. The following Tables 3-4 describe the performances of the prediction model. Based on all the above observations, the PSF prediction model-based forecasting is efficient for the data set.

The network performance was analyzed based on the proposed clustering model in terms of average loss percentage. Figure 7 shows the loss percentage of the network evaluated with and without a clustering model, after implementing the proposed model the network loss rate was reduced from more than 50% to less than 10%.

The proposed CLUSTERING model is compared with the BGPmon and the ARTEMIS detection model in terms of detection time. BGPmon is a monitoring-based model and ARTEMIS is a historical-based automatic real-time detection and mitigation model. Figure 8 shows the comparison of detection delay among different methods. BGPmon-based detection time is an average greater than 1 min. Whereas the ARTEMIS model's detection time is an average of less than or equal to 1 min and the detection time of the proposed clustering model is less than 40 sec.

Table 3: Performance metrics comparison with existing approaches and the proposed method

Performance measures	Learning percentage of proposed method	SVM	ANN	RF	KNN	Proposed
Accuracy	80	70	92	82	78	98.56
Sensitivity	60	36	73	80	59	97.52
Specificity	70	58	98	85	84	98.53
Precision	70	63	97	84	82	98.35
FPR	80	23	5	20	9	0.80
FNR	80	40	10	19	35	4.00
Recall	70	35	5	12	14	0.30

Table 4: Mean absolute error

Set	ETS	ARIMA	PSF
1	9.56	4.34	3.25
2	10.05	6.22	5.02
3	12.34	6.05	5.63
4	15.19	9.17	7.05

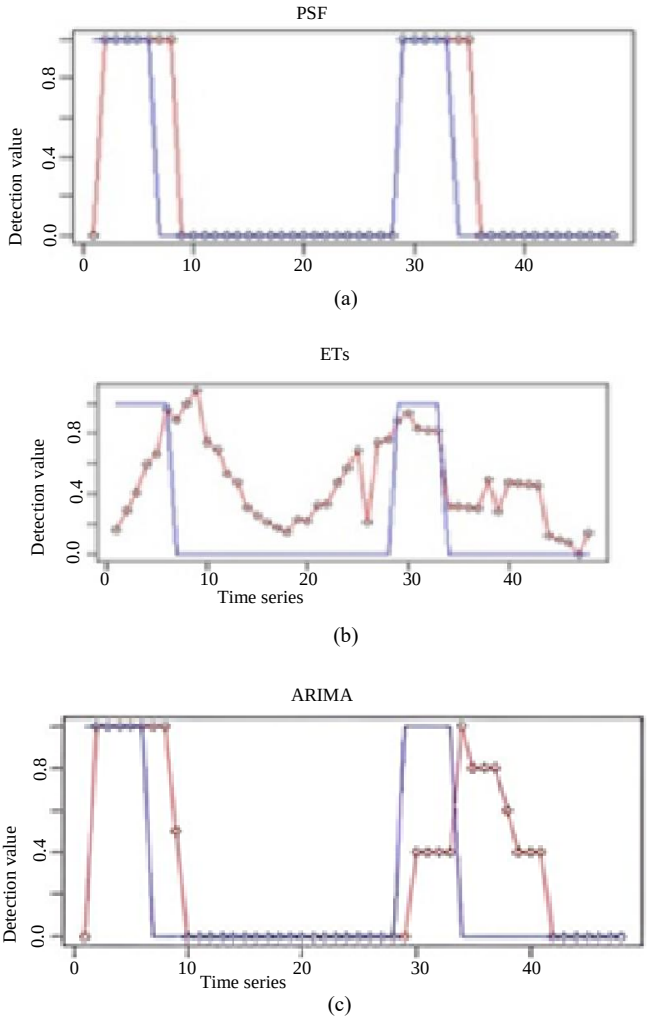

Fig. 6: Prediction analysis

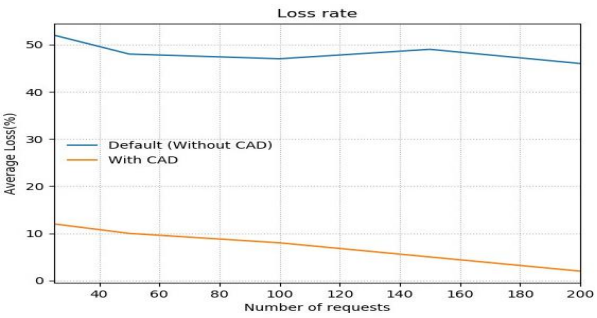

Fig. 7: Loss rate

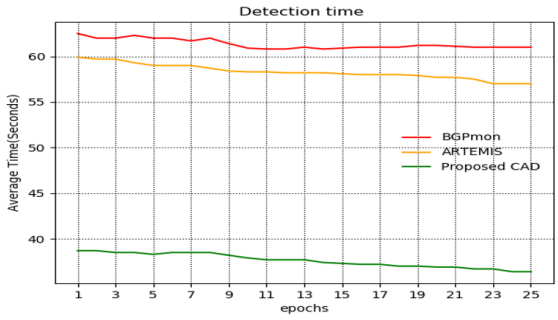

Fig. 8: Detection delay

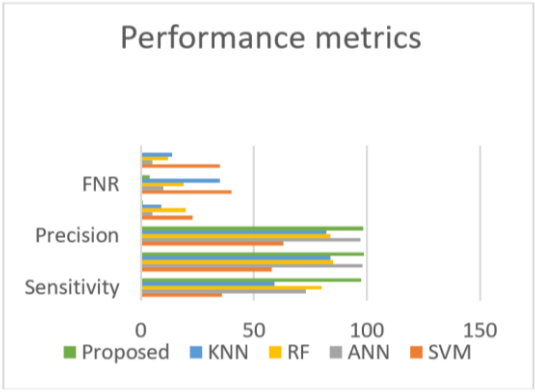

Fig. 9: Overall performance metrics

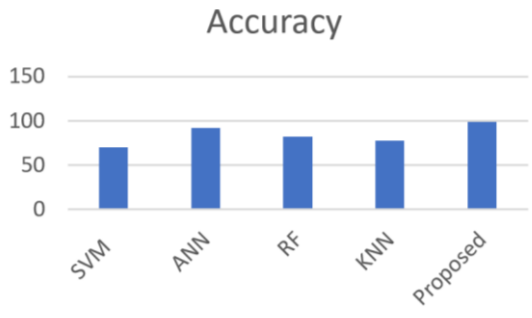

Fig. 10: Accuracy rate

Conclusion

There are many problems that may arise in a network, but network security is one of the most important responsibilities nowadays. DDoS attacks are the most

prevalent and dynamic attacks today; attackers are using a variety of dynamic methods to conduct DDoS attacks, which are becoming more effective. Traditional security systems are less effective and insufficient in dealing with these problems, according to the experts. Thus, with the help of the proposed system we have achieved the following results:

i) By using the hybrid k-means clustering with an enhanced neural network algorithm better performance can be achieved in the detection of a DDoS attack with decreasing processing loads and times in a promising manner
ii) The proposed model is compared with the existing models and the accuracy is achieved at 98.6% for DDoS detection
iii) False positive Rate is decreased to up to 6% only, also the precision, and recall rate is improved as compared to the existing models
iv) Also, the complexity and CPU utilization are reduced to a great extent as described and compared in the experimental results section. Also the accuracy rate as shown in Fig. 10 (see page no. 757) is highest in proposed model

Control plane and data plane work are no longer separated in the network thanks to the use of SDN, which has consolidated control plane work in a network controller. Traditional networking was based on devices that had control planes and data planes merged into a single device before the introduction of SDN, however, this was altered with the introduction of SDN.

One of the most significant benefits of SDN is the provision of security. Handling dynamic high-rate DDoS assaults has gotten easier and more practical with the implementation of SDN. The proposed clustering model detects the attack within the controller and prevents the attack using an enhanced neural network model. It was discovered that the findings of the experiments were obtained via the use of emulation tools such as Mininet and a real-time data packet analyzer (Wireshark). Once the ODL has been determined to be the most effective SDN controller, the susceptibility to DDoS attacks is assessed via the use of several DDoS penetration tools, such as hping3 and Nping, among others.

The proposed work thus obtains improved detection accuracy, specificity, sensitivity, precision, and false positive rates decreases as summarized in Table 3. The same has been depicted in Fig. 9 (see page no. 757) to compare the outputs of the other methods compared to proposed system.

In future work, it is possible to evaluate how the suggested solution would fare in the event of a subnet assault. If a whole subnet is under assault, instead of just one host, another intriguing area to pursue is the detection and prevention of DDoS attacks (DDoS). The use of a distributed controller design results in significantly improved load distribution, processing power, and dependability.

Acknowledgment

There is no conflict of interest and no financial support from any other party.

Funding Information

The authors themselves have funded this manuscript.

Author's Contributions

Hema Surendrakumar Dhadhal: Participated in all experiments, coordinate the data analysis, and contributed to the written of the manuscript.

Paresh P Kotak: Designed the research plan and helped to get the results.

Ethics

This article contains original and unpublished material. The authors confirm that no ethical issues are involved.

References

Aydin, B. (2019). *Identifying critical cybersecurity controls at country level* (Master's thesis, Fen Bilimleri Enstitüsü).

de Assis, M. V., Carvalho, L. F., Rodrigues, J. J., Lloret, J., & Proença Jr, M. L. (2020). Near real-time security system applied to SDN environments in IoT networks using convolutional neural network. *Computers & Electrical Engineering*, *86*, 106738. https://doi.org/10.1016/j.compeleceng.2020.106738

Elsayed, M. S., Le-Khac, N. A., Dev, S., & Jurcut, A. D. (2020, August). Ddosnet: A deep-learning model for detecting network attacks. In *2020 IEEE 21st International Symposium on" A World of Wireless, Mobile and Multimedia Networks" (WoWMoM)*, (pp. 391-396). IEEE. https://doi.org/10.1109/WoWMoM49955.2020.00072

Hu, D., P. Hong & Chen, Y. (2017). "FADM: DDoS Flooding Attack Detection and Mitigation System in Software-Defined Networking," GLOBECOM 2017. IEEE Global Communications Conference, Singapore, 2017, pp. 1-7, https://doi.org/10.1109/GLOCOM.2017.8254023

Haider, S., Akhunzada, A., Mustafa, I., Patel, T. B., Fernandez, A., Choo, K. K. R., & Iqbal, J. (2020). A deep CNN ensemble framework for efficient DDoS attack detection in software defined networks. *IEEE Access*, *8*, 53972-53983. https://doi.org/10.1109/ACCESS.2020.2976908

Iqbal, W., Abbas, H., Daneshmand, M., Rauf, B., & Bangash, Y. A. (2020). An in-depth analysis of IoT security requirements, challenges and their countermeasures via software-defined security. *IEEE Internet of Things Journal*, *7*(10), 10250-10276. https://doi.org/10.1109/JIOT.2020.2997651

Jalili, R., Imani-Mehr, F., Amini, M., & Shahriari, H. R. (2005). Detection of distributed denial of service attacks using statistical pre-processor and unsupervised neural networks. In *Information Security Practice and Experience: 1st International Conference, ISPEC 2005, Singapore, April 11-14, 2005. Proceedings 1* (pp. 192-203). Springer Berlin Heidelberg.

Kalliola, A. Lee, K. Lee., H. & Aura T. (2015). Flooding DDoS mitigation and traffic management with software-defined networking.248-254. https://doi.org/10.1109/CloudNet.2015.7335317

Kasim, Ö. (2020). An efficient and robust deep learning-based network anomaly detection against distributed denial of service attacks. *Computer Networks*, *180*, 107390. https://doi.org/10.1016/j.comnet.2020.107390

Li, L., & Lee, G. (2005). DDoS attack detection and wavelets. *Telecommunication Systems*, *28*, 435-451. https://doi.org/10.1007/s11235-004-5581-0

Ma, L., Chai, Y., Cui, L., Ma, D., Fu, Y., & Xiao, A. (2020, June). A deep learning-based DDoS detection framework for Internet of Things. In *ICC 2020-2020 IEEE International Conference on Communications (ICC)* (pp. 1-6). IEEE. https://doi.org/10.1109/ICC40277.2020.9148944

Macedo, R. de Castro, R., Santos, A., Ghamri-Daudane , Y., & Nogueira, M. (2016). Self-Organized SDN Controller Cluster Conformations against DDoS Attacks Effects. 1-6. https://doi.org/10.1109/GLOCOM.2016.7842259

Mehr, S. Y., & Ramamurthy, B. (2019, December). An SVM based DDoS attack detection method for Ryu SDN controller. In *Proceedings of the 15th International Conference on Emerging Networking Experiments and Technologies* (pp. 72-73). https://doi.org/10.1145/3360468.3368183

Rghioui, A., Khannous, A., & Bouhorma, M. (2014). Denial-of-Service attacks on 6LoWPAN-RPL networks: Threats and an intrusion detection system proposition. *Journal of Advanced Computer Science & Technology*, *3*(2), 143. https://doi.org/10.14419/jacst.v3i2.3321

Shin, S., Yegneswaran, V., Porras, P., & Gu, G. (2013, November). Avant-guard: Scalable and vigilant switch flow management in software-defined networks. In *Proceedings of the 2013 ACM SIGSAC Conference on COMPUTER & Communications Security* (pp. 413-424). https://doi.org/10.1145/2508859.2516684

Ujjan, R. M. A., Pervez, Z., Dahal, K., Bashir, A. K., Mumtaz, R., & González, J. (2020). Towards SFlow and adaptive polling sampling for deep learning-based DDoS detection in SDN. *Future Generation Computer Systems*, *111*, 763-779. https://doi.org/10.1016/j.future.2019.10.015

Ujjan, R. M. A., Pervez, Z., Dahal, K., Khan, W. A., Khattak, A. M., & Hayat, B. (2021). Entropy based features distribution for anti-ddos model in sdn. *Sustainability*, *13*(3), 1522. https://doi.org/10.3390/su13031522

Velliangiri, S., & Premalatha, J. (2019). Intrusion detection of distributed denial of service attack in cloud. *Cluster Computing*, *22*(Suppl 5), 10615-10623. https://doi.org/10.1007/s10586-017-1149-0

Xu, W., Hu, G., Ho, D. W., & Feng, Z. (2019). Distributed secure cooperative control under denial-of-service attacks from multiple adversaries. *IEEE Transactions on Cybernetics*, *50*(8), 3458-3467. https://doi.org/10.1109/TCYB.2019.2896160

Zargar, S. T., Joshi, J., & Tipper, D. (2013). A survey of defense mechanisms against distributed Denial of Service (DDoS) flooding attacks. *IEEE Communications Surveys & Tutorials*, *15*(4), 2046-2069. https://doi.org/10.1109/SURV.2013.031413.00127