Ensembled Machine Learning Techniques for DDoS Detection in SDN

Tarakanadha Reddy P[2][0000−0002−2046−4625], Shalini P V[1][0000−0002−4842−959X], and Radha V[2]

[1] Department of Computer Science and Engineering, National Institute of Technology, Warangal, India
[2] Centre for Cloud Computing, Institute for Development and Research in Banking Technology, Hyderabad, India

Abstract. Software Defined Networking(SDN) focuses on overcoming the drawbacks of traditional networks and offers the advantage of flexibility in managing the networks. On the other hand, this new paradigm makes networks susceptible to attacks. DDoS is one of those significant attacks. DDoS makes resources unavailable to legitimate users, and one of the mechanisms that attackers follow is the TCP-SYN flood to launch the DDoS attack. The TCP SYN flood attack takes advantage of the three-way handshake to exhaust the web server's resources. We proposed an approach to detect DDoS attacks in SDN based on an ensemble technique.Our proposed approach uses stacking model, combining bagging and boosting models as ensembled techniques. we implemented our proposed approach on dataset. We have generated our own dataset containing the required features. We show that our proposed approach gives better accuracy than existing models in the literature. We validated our proposed approach on both generated dataset and existing dataset.

Keywords: Software Defined Networks · DDoS Attacks · Ensembled Techniques · Stacking Model

1 Introduction

Distributed Denial of Service (DDoS) in Software Defined Networking (SDN) has become an emergent serious cyber security issue. DDoS attacks aim to drain the victim's resources by preventing legitimate users from accessing them. Even though SDN is a viable technology with a bright future, it is susceptible to DDoS attacks. Some of the key reasons that DDoS is hard to detect are: the attack's spreading nature, the attack's changeable time trend, the frequency of the attack, the use of spoofed IP addresses, and the complexity in recognizing traffic features [2].

In traditional networks, various strategies have been employed to mitigate the impact of DDoS attacks [7]. Packet analysis is resource-intensive in traditional networks. As a result, sampling approaches were used to verify the packets. To detect and mitigate DDoS attacks, there are statistical, machine learning, and deep learning methodologies [11].

Many studies have used OpenFlow's flow capabilities to identify attacks in SDN, especially DDoS [6]. The flow level information captured from all the switches is used to identify the existence of attacks in this study. Flow level information can be easily retrieved because SDN is built on a flow-based architecture. Flow analysis is more resource-efficient than packet analysis. TCP SYN flood consumes resources on the target web server by exploiting a portion of the regular TCP three-way handshake.

2 Related Work

In this section, we discuss schemes on security of SDN in recent years.

Dehkordi et al. [1] introduced a approach based on both statistical and machine learning techniques for classification. They used the statistical entropy value of the collected features for classification.

In the Scheme [12] Ye, Jin presented a classification approach based on SVM for identifying DDoS attack traces in the network. The authors have generated their own dataset using both attack and benign traffic in mininet environment. The dataset which is generated for their study is too small. They got an accuracy of 95.24%.

Chen et al. [4] proposed a boosting approach called XGBoost to detect DDoS attacks. They worked on cloud environment . Their scheme [4] got an accuracy of 98.53%

For detecting the TCP SYN flood attacks, Tuan et al. [10] applied KNN and XGBoost. They validated their model on CAIDA 2007 data set and a testbed environment. This scheme obtained an accuracy of 98%.

Diaz et al. [9] used six machine learning algorithms with random and grid search hyperparameter optimization techniques to detect DDoS attacks of low rate. They implemented their model on CIC DoS 2017 dataset and obtained an accuracy of 95%.

Deepa et al. [5] used ensembled techniques to detect DDoS attacks. Their scheme SVM combined with the self-organizing maps got an accuracy of 98.12%.

Ancy et al. [8] used all the existing classification models on their generated custom data set. Out of all those classification models, SVM achieved an accuracy of 99.73%.

Braga et al. [3] used self-organizing maps by extracting six features for detecting DDoS attacks. These six features are: Average Flow Duration, Average Packet Count, Pair Flows Percentage, Different Ports Growth, Single flows Growth, and Average Byte Count. Their scheme got an accuracy of 98.61%.

By performing a literature review, we know that most of the works are carried out on the datasets with existing features. Schemes [3],[8] show that the extracted features improve the accuracy over the existing features.

Our proposed work is focused on generating the dataset having the essential six features from the Table. 1 by using a testbed environment. Apart from this, we proposed using the stacking based ensembled model that combines different

Table 1. Previous studies

Previous Works	Features Considered
Braga et al. [3] Self Organizing Maps	Average of packets, Average of bytes, Pair flows Percentage, Average of duration of all the flows, Different ports growth, Single flows growth
Ancy et al. [8] SVM Classification	Count of Flows, Average byte count, Average packet count, Average duration, Entropy of protocol, Entropy of source IP, Entropy of destination IP

models at the base and high levels to improve the accuracy of the classifying the attack. Our proposed approach is validated with ”DDoS attack SDN Dataset”.

3 Proposed work

In this section, we discuss the proposed work. We created our own network topologies using mininet to run various scenarios. Pox is one of the open flow controllers used in Software Defined Networking to control the behavior of the networks. The controller connects to the OpenFlow switches in the network. These switches will have flow tables. A flow table is the collection of flow entries. OpenFlow switch acts like the forwarding medium of the packets based on the entries in the flow table. Packet in the network arrives first at the switch through one of the ports. After packet arrival, it will be matched with entries in the flow table. If a match to the incoming packet is found, then based on the actions associated with the flow entry, the packet will be forwarded. If a match to the incoming packet is not found, then that packet will be sent to the controller. These packets are called flow-initiations. Controller on receiving the flow-initiations, will associate the actions to be taken and add a flow entry to the switch's flow table as a response.

3.1 Feature Extraction from the Flows Collected

We collect the flows coming to the switches in our network. Flow is defined as the group of packets that are having the same src_ip, dst_ip, src_port, dst_port, and protocol. From these flows, we extract the required features and store them to create a dataset. Considering the previous works mentioned in Table. 1, the following six features: Average Flow Duration, Average Packet Count, Average Byte Count, Pair Flows Percentage, Different Ports Growth, and Flows Count [3] [8] are taken in our proposed approach. The six features used in our proposed approach are,

Average Packet Count : During the benign traffic, the flow of packets is high as nodes are intended to communicate. But during attack traffic, flow of packets is low as the attacker's goal is to exhaust as many ports as possible by sending a very minimal count of packets to the web server on a network. Here, finding the median value will be more effective than taking up the average when the flows have a high number of packets.

$$\text{Average Packet Count} = \begin{cases} \frac{C\left(\frac{N}{2}\right)+C\left(\frac{N+1}{2}\right)}{2} & \text{if N is even} \\ C\left(\frac{N+1}{2}\right) & \text{otherwise;} \end{cases}$$

Fig. 1. Average Packet Count

Here, C represents the number of packets of each flow and N represents the total count of flows in a network.

Average Byte Count : During attack traffic, attackers send minimal bytes to the web server to use the port space. The average byte count is calculated as shown in the Fig. 1, where C represents the total number of bytes per each flow and N represents the total count of flows in a network.

Average Flow Duration : In the same way, we can calculate the median value to the duration of all flows. Flow duration will be higher if there is a benign traffic, and it will be lower if there is an attack traffic as flows from other hosts in a network increases. The average flow duration is calculated as shown in the Fig. 1, where C represents the duration of flow, and N represents the total flows in a network.

Pair Flows Percentage : Pair flows percentage helps to determine the count of pair flows during an interval in a network. A flow is said to be a pair flow only when the following three conditions satisfy,

1. Source IP of Flow A is the same as Destination IP of Flow B,
2. Destination IP of Flow A is the same as Source IP of Flow B,
3. Both Flow A and Flow B have the same protocol used for communication.

There will be packets from spoofed IP's during the attack, which decreases the percentage of pair flows. Pair flows percentage is calculated using Equation 1.

$$\text{Pair Flows Percentage} = \frac{\left(2 * \text{Count of Pair Flows}\right)}{\text{Count of Flows}} \tag{1}$$

Different Ports Growth During attack traffic, the attackers target to exhaust the port space present on the web server, which increases the count of ports being used during the attack time. Calculating the different port's growth can be done using Equation 2.

$$\text{Different Ports Growth} = \frac{\text{Number of Ports}}{\text{Interval}} \tag{2}$$

Flows Count : During attack traffic, count of flows increases as the flows will be generated with packets from spoofed IP's. The number of flows taken from the network during attack will give the count of flows in a interval.

3.2 Classification

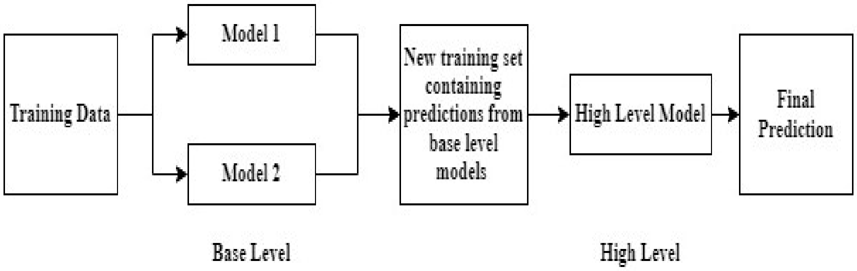

Fig. 2. Architecture of Stacking Model

Here, we proposed a stacking based ensembled approach to improve the accuracy in classifying the DDoS attacks. Stacking based ensembled approach contains base level and high level and it's architecture is given in Fig. 2. Ensembled approaches are of three types, they are Averaging, Boosting, and Bagging methods. Averaging is also known as stacking and it typically refers to the combination of different models and uses a meta-classifier to merge many basic classification models. This method uses a combination of traditional classifiers to create a generic machine learning model.

The initial stage in stacking is to aggregate every model's output in a base level which results into a new dataset. This dataset comprises the prediction of the every model in base level along with correct classification for every instance of the original dataset. The new dataset that is generated from base level models will be provided as input to the classifier in the high level. The architectures of the base and high-level classifiers are necessary for stacking-based ensemble approach. In this study, We taken the base classifiers as Adaboost and Extra Tree classifier, while Logistic Regression taken as high-level classifier.

4 Experimental setup

The dataset generation work has been carried out on the Lenovo Thinkpad E470, having 8 GB RAM and a 64-bit processor with Ubuntu operating system.

4.1 Dataset Generation:

A detailed description of the steps followed for generating dataset are given below

Topology Creation: Mininet is a emulator which builds network that is virtual in nature and similar to real world network topologies. This virtual network comprises hosts, switches, controllers, and connections. Mininet is used for R&D, learning, modeling, testing, debugging, and other activities related to the network. As part of our proposed approach, we created a few custom topologies and taken a few default topologies from Mininet. The topologies created are given in Table. 2

Table 2. Details of the Topologies created

Topology	No of Switches	No of Hosts	Web Server
1(Custom)	3	5	H5
2(Linear)	3	12	H6
3(Custom)	4	41	H29
4(Custom)	7	61	H42
5(Single)	1	15	H12
6(Custom)	7	80	H72
7(Custom)	3	5	H5
8(Custom)	13	36	H36
9(Linear)	3	6	H6
10(Custom)	9	120	H98
11(Custom)	12	160	H122
12(Minimal)	1	2	H1

Benign (Normal) Traffic Generation: Curl command is a command-line tool that transfers the data from/to the server. Curl command used in a script file generates benign traffic without user involvement. The script file sends requests from hosts to the web server at regular time intervals.

Collecting Flows of Benign Traffic: The flows present in the switches of network can be collected using commands. Using the commands, flows are collected at a regular interval between 60 to 100 secs(The hard time out to the flows is 60 secs).

Feature Extraction from Collected Flows: After collecting the flows from the switches. The steps to extract the values of the six features are given in Section 3.1. As this is benign traffic, label "0" is added and stored feature values on to a csv file.

DDoS Traffic Generation: Scapy is the tool used for packet generation, and it is one of the powerful packet manipulation program written in python. As scapy can create and send packets to the web server, it is considered for our attack generation along with Hping3. Hping3 is used to flood the victim and manages the amount, size and, fragmentation of packets. By considering size and amount, the attack can be performed from any host to the web server on a network. Scapy generates TCP packets with the source IP as random from the topology and the destination as the web server. As the server resources were being used to address the connection requests, legitimate users can not communicate with the server. During the attack, benign traffic has been generated and sent to the web server. The time taken to get acknowledgment from server to legitimate users is more when compared to the normal time and even some times, legitimate users are unable to get the acknowledgement.

Collecting Flows of DDoS Traffic: The flows present in the switches of network can be collected using commands. Using the commands, flows are collected at a regular interval between 60 to 100 secs(The hard time out to the flows is 60 secs).

Feature Extraction from Collected Flows: After collecting the flows from the switches. The steps to extract the values of the six features are given in Part 3.1. As this is benign traffic, label "1" is added and stored feature values on to a csv file.

The dataset that is generated has 24025 records. These records are extracted from an average of 5 lakh flows roughly. Out of 24025 records, 13753 records indicate benign traffic, and 10272 records indicate attack traffic. With this distribution of records, It seems to be that the generated dataset is balanced. As the next step, generated dataset is used to train a model based on our proposed approach for attack detection.

4.2 Classification:

During classification, partition of the dataset was 80% of the training set and 20% of the testing set. Classification algorithms like K Nearest Neighbor, Logistic Regression, Support Vector Machine, Gaussian Naive Bayes, and Linear Discriminant analysis were performed on the dataset. At last, one of model from bagging and boosting were taken as the base classifiers for the proposed stacking based ensembled approach, along with Logistic regression as the higher level classifier. We are combining base and higher-level classifiers in our proposed approach. This proposed approach evaluated with generated custom dataset. ”DDoS attack SDN dataset” was evaluated with the proposed approach. This dataset contain 22 features. Chi-Square, PCA, and Mutual Information(MI) were used to select the best features and feature subset of size(12) is constructed.

5 Results and Analysis

In this section, we discussed the results and analysis of our proposed approach. Result and analysis was done in two sections. In the first section 5.1, result analysis was done with generated dataset. In the second section 5.2, result analysis was done with ”DDoS attack SDN dataset”. The training and testing of the proposed approach carried out on Google Colab, having an Intel Xeon processor with a frequency of 2.20GHz and 13 GB of RAM. For measuring the model's performance, performance metrics like accuracy, recall, precision, and f1 score were used.

5.1 Result Analysis with Generated Dataset

The dataset with features Average Packet Count, Average Byte Count, Pair Flows Percentage and Flows Count was applied with the proposed approach. The performance metrics of the proposed approach were given in Table. 3. Out of all the techniques, GNB had less accuracy of 81.49%. KNN, SVM, LR, LDA were having accuracy of 99.93%, 99.61%, 99.25%, 99.09% respectively. Our Proposed approach (Base Learners - AdaBoost, Extra Tree & High level Learner - Logistic Regression) was having higher accuracy of 99.98% over the other techniques.

Table 3. Classifiers' performance by considering all the features

Classifier	Accuracy(%)	Recall(%)	Precision (%)	F1 Score(%)
SVM	99.611	99.187	99.901	99.543
GNB	81.492	69.713	99.607	82.021
KNN	99.93	99.901	99.934	99.918
LDA	99.098	98.57	99.312	98.94
Logistic Regression	99.25	98.733	99.509	99.119
Proposed Approach	**99.986**	**100**	**99.967**	**99.983**

5.2 Result Analysis with ”DDoS attack SDN dataset”

”DDoS attack SDN Dataset” was an emulated dataset. The performance metrics of the proposed approach were given in Table. 4. The proposed approach with Mutual Information & Chi-Square provided an accuracy of 99.2%. By selecting all the features, the proposed approach offered an accuracy of 99.83%. Using the feature selection methods, the proposed approach given the similar results by reducing the dimensionality.

Table 4. Our proposed approach performance with DDoS Attack SDN Dataset

Method	Accuracy(%)	Recall(%)	Precision (%)	F1 Score(%)
All Features	**99.837**	**99.983**	**99.593**	**99.837**
Mutual Information	99.20	99.923	97.973	98.938
Chi-Square	99.242	99.629	98.396	99.009
PCA	97.875	96.182	98.38	97.269

6 Conclusion and Future Work

DDoS (Distributed Denial of Service) had become an emergent serious cyber security concern. Even while Software Defined Networking (SDN) makes network management more manageable, it was /prone/ subject to DDoS attacks. In traditional networks, a different range of approaches had been used to detect and mitigate attacks. In this study, flow characteristics collected from network were used to detect DDoS attacks. In our work, flow collection was done, followed by feature extraction, made an emulated dataset. This dataset was used to train the classifier. The classifier taken up for this work was based on stacking based ensembled approach. The proposed approach was an effective strategy that integrated different learning algorithms at the base level and high level with improved attack detection accuracy. The proposed approach combined several classifiers into a single composite model that was more accurate in detection. The goal of this research was to see that the proposed ensembled approach would outperform the classification models in terms of accuracy by employing extracted features from the flows. By seeing the results of the classifiers, there was an increase in the accuracy of the proposed ensembled approach when compared to the other techniques used. As part of future work, We want to include other types of mechanisms used to launch DDoS attacks in our dataset, increasing the dataset size which will help to ensure that the proposed approach performs as expected.

References

1. Afsaneh Banitalebi Dehkordi, MohammadReza Soltanaghaei, and Farsad Zamani Boroujeni. “The DDoS attacks detection through machine learning and statistical methods in SDN”. In: The Journal of Supercomputing 77.3 (2021), pp. 2383–2415.
2. Kriti Bhushan and Brij B Gupta. “Distributed denial of service (DDoS) attack mitigation in software defined network (SDN)-based cloud computing environment”. In: Journal of Ambient Intelligence and Humanized Computing 10.5 (2019), pp. 1985–1997.
3. Rodrigo Braga, Edjard Mota, and Alexandre Passito. “Lightweight DDoS flooding attack detection using NOX/OpenFlow”. In: IEEE Local Computer Network Conference. IEEE. 2010, pp. 408–415.
4. Zhuo Chen et al. “XGBoost classifier for DDoS attack detection and analysis in SDN-based cloud”. In: 2018 IEEE international conference on big data and smart computing (bigcomp). IEEE. 2018, pp. 251–256.

5. V Deepa, K Muthamil Sudar, and P Deepalakshmi. “Design of ensemble learning methods for ddos detection in sdn environment”. In: 2019 International Conference on Vision Towards Emerging Trends in Communication and Networking (ViTECoN). IEEE. 2019, pp. 1–6.
6. Marinos Dimolianis et al. “Mitigation of multi-vector network attacks via orchestration of distributed rule placement”. In: 2019 IFIP/IEEE Symposium on Integrated Network and Service Management (IM). IEEE. 2019, pp. 162–170.
7. Yosr Jarraya, Taous Madi, and Mourad Debbabi. “A survey and a layered taxonomy of software-defined networking”. In: IEEE communications surveys & tutorials 16.4 (2014), pp. 1955–1980.
8. Ancy Sherin Jose, Latha R Nair, and Varghese Paul. “Towards Detecting Flooding DDOS Attacks Over Software Defined Networks Using Machine Learning Techniques”. In: REVISTA GEINTEC-GESTAO INOVACAO E TECNOLOGIAS 11.4 (2021), pp. 3837–3865.
9. Jesus Arturo Perez-Diaz et al. “A flexible SDN-based architecture for identifying and mitigating low-rate DDoS attacks using machine learning”. In: IEEE Access 8 (2020), pp. 155859–155872.
10. Nguyen Ngoc Tuan et al. “A DDoS attack mitigation scheme in ISP networks using machine learning based on SDN”. In: Electronics 9.3 (2020),p. 413.
11. Rui Wang, Zhiping Jia, and Lei Ju. “An entropy-based distributed DDoS detection mechanism in software-defined networking”. In: 2015 IEEE Trustcom/BigDataSE/ISPA. Vol. 1. IEEE. 2015, pp. 310–317.
12. Jin Ye et al. “A DDoS attack detection method based on SVM in software defined network”. In: Security and Communication Networks 2018 (2018).