Detecting DoS attack in SDN using Machine Learning

Aswin P J
Cyber Security Systems and Networks
Amrita School of Engineering, Amritapuri, India
Kollam, India
amenp2csn21005@am.students.amrita.edu

***Abstract*—Software-defined Network (SDN) is the network architecture design that makes a network more efficient and easier to manage. The controller is the brain of the SDN and DoS is the major threat to this architecture. During a DoS attack, the controller is overloaded with spoofed packets and it won't be able to take the decision which affects the packet flow of the entire network. A network intrusion detection system (NIDS) on the SDN controller is the solution proposed in this paper for detecting DoS attacks in the SDN network. A machine learning model trained with a classifier algorithm is used to detect DoS symptoms in the network as a part of NIDS. Evaluated a few models trained with different classifier algorithms to select the best algorithm among them. The obtained result shows decision tree has a high recall value of 0.78 which is less in the real situation. Adaboost is having high precision(0.969) and high accuracy(0.897).**

***Index Terms*—SDN, DoS, Machine Learning, NIDS**

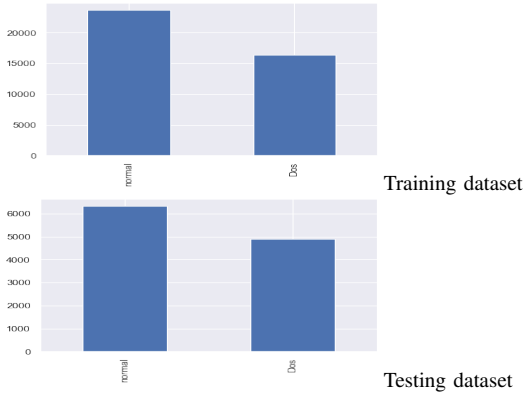


Fig. 1: Class Distribution

I. Introduction

Software defined network is a network architecture that provides a centralized programmable network that is easy to handle. It has a lot of benefits compared to a traditional system like great reliability, better network management, and less costly and highly scalable. The main threat of the SDN is the SDN controller failure. The controller is the brain of SDN and due to the DoS attack lot of issues occurs like resource exhaustion, packet drops, processing spoofed packet sent by the attacker, no new flow instructions, etc. DoS (Denial of Service) is a cyber-attack that makes a resource or service unusable. These DoS attacks are very common in SDN systems by overflowing the SDN controller with heavy traffic that makes the controller inactive or affects its functionality. It will completely affect the network traffic and packets in the network. A DoS attack on an organization that follows SDN architecture will face access issues to the internet, email, or cloud services and they need to detect these kinds of attacks without delay and resolve it as early as possible.

Machine learning algorithms can be used to detect and mitigate DoS attacks. Some papers [6] implement models to control DoS attacks in SDN using algorithms like Support Vector Machines, Random forest, KNN, etc. In [3], the author is proposing a system with K-mean and KNN algorithm to detect the attack, and in [10] they extracted six tuple fixtures of the controller and analyzed it using the SVM algorithm to detect DoS attack and [7] also used SVM for detection of the attack. In another work [9], they are comparing different ML algorithms in a dataset from SDN and it shows that the J48 algorithm has only 17 seconds to train the model and 5 seconds to test it which is very less compared to other models. They are also introducing a mitigating script that blocks network ports where the attack is happening.

This paper is proposing a method of implementing a network intrusion detection system(detect malicious packets in a network) on the controller of SDN. Using machine learning classifier algorithms like decision tree classifier, adaboost classifier it is easy to detect the attack. Machine learning is a suitable solution for this because it can classify a huge number of packets in real-time which is a difficult manual task. Even though all algorithms have good precision and accuracy score they fail to have a good recall score. The decision tree is the suggestible algorithm.

II. Related work

Research on the topic of security of SDN is increased over years. Researchers are approaching prevention methods for different cyber attacks with different trending technologies for better optimum solutions. In the paper [4], the authors are using the IP filter technique on open flow protocol to observe and detect malicious user packets. If an average number of connections of an IP address and a minimum number of packets from an IP address is less than a value specified by the author, it is considered a DoS attack. The model is not efficient when traffic is very high. Another [8] is introducing a solution

for detecting DoS attacks using the entropy difference of the destination IP address. Entropy will decrease if several hosts face a DoS attack and if the entropy value went down to an already fixed threshold value it is considered a DoS attack. The experiment result of the model is 96% accuracy in detecting DoS attacks. This model is not analysing each packet, so it drops legitimate packets along with malicious ones. From the above works, we can understand that a machine learning model can overcome these limitations and as mentioned in [4], the count of connections is an important feature for detecting the attack.

The common solution used by most of the papers in this field is machine learning. Machine learning is a suitable solution for this because it can classify a huge number of packets in real-time which is a difficult manual task. Also, the large, labelled dataset is available for training and testing the model. The paper [7] is proposing a framework which collects traffic, analyses and detects fake packets using the SVM model, and controls packet flow to mitigate the attack according to the model's decision. The experiment is conducted using the KD99 data set which is the older version of our data set, NSL-KDD. The accuracy of the experiment result is 99.8% but the SVM is taking a lot of time to train and test the model. So, I'm not using that model in this project.

NSL-KDD is a commonly used dataset for the network IDS. The papers [5] and [1] are working on the same data set with different methodologies. Paper [5] is explaining deeply about the data set features and analysing them using different machine learning classification models(J48, SVM, Naive Bayes). The data set contain 4 types of attack packets and normal packets. The result shows that the J48 algorithm classifies the data set with 99.8% accuracy. In [1], the author is deploying a machine learning model which is trained with a Naive Bayes algorithm using the NSL-KDD dataset and testing the model on a real-time DoS attack. The precision of the experimental result is 92% but recall is 66% which is less compared to the precision.

Few researchers have compared different machine learning algorithms and analysed the performance of each comparing others. The [2] is using different classification algorithms like decision trees and fuzzy logic for training the model and comparing its metric values. The author is suggesting to use a decision tree for classifying the packets which have 94% precision which is a better value compared to other algorithms. The decision tree is the primary algorithm which I'm using in this paper. Similarly, in the paper [9], authors are comparing J48, random forest, SVM, and KNN to detect fake packets and using a mitigation script model that will instruct the controller to block the DoS port. The metric result shows that precision, accuracy and recall values are similar for all algorithms but the training and testing time of J48 is 17 seconds and 3 seconds which is very less compared to other algorithms. The mitigation idea used in this paper is not effective and also closing a port for a long time will drop normal packets to the network. J48 is also a decision tree algorithm. The authors of the paper [12] are also comparing

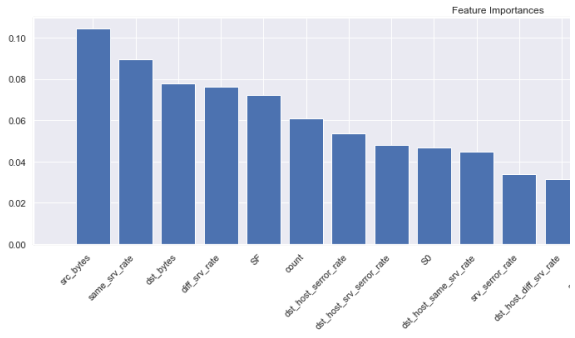


Fig. 2: Feature importance bar graph

the performance of different machine learning algorithms in the NSL-KDD dataset. The author's selected 6 features from the dataset which easily extractable from the SDN network. I'm also using the same features for training the model. From the above discussion, most of the machine learning papers are showing better results for detecting DoS attacks in SDN networks. The decision tree is the recommended algorithm for classifying packets from most of the research papers. This paper will also follow the decision tree and compare it with other classification algorithms.

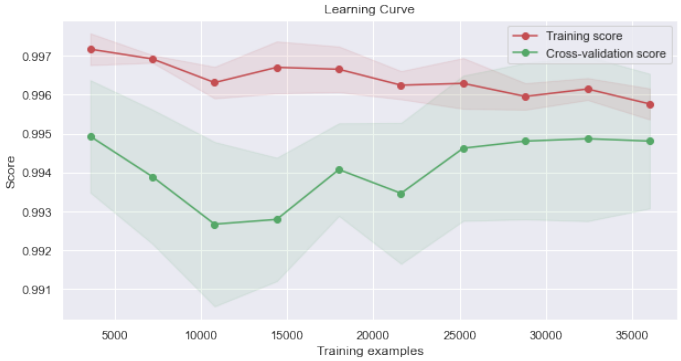
III. Methodology

A. Dataset

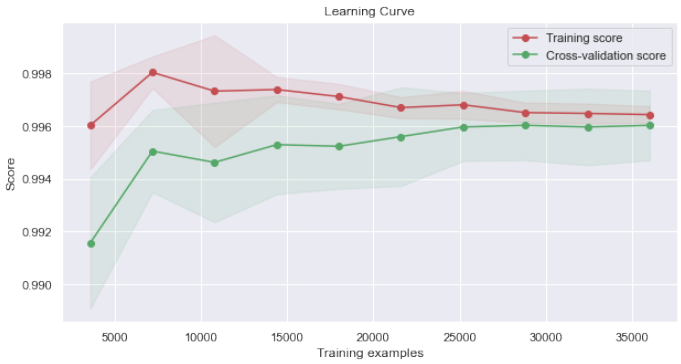
In this paper, I'm using the NSL-KDD dataset which is an upgraded of KDD99 dataset which contain data samples of internet traffic data. This is the common dataset used for training intrusion detecting systems. KDD99 dataset is created from internet traffic records collected in a data mining competition held in 1999. The cleaned-up version of KDD99 is the NSL-KDD dataset. This dataset is proposed in the paper [11] and is available on the UNB website. The training dataset and testing dataset are provided in a separate CSV file, so we don't need to split the dataset. There are no null/NAN values in the dataset. The dataset is classified into 5 different categories normal, DOS, R2L, U2R, and Probe. For this project, we are focusing only on DOS attacks. So, we can remove R2l, U2R and Probe samples from the dataset. After removing it, we have 113270 samples in the training dataset and 17169 samples in the testing dataset. In the training dataset, there are 67343 normal samples and 45927 DOS samples and in the testing dataset, 9711 normal and 7458 DOS is available. Both CSV files contain 42 extracted features which are done by the dataset author.

In paper [12], they are using only six features from this dataset which are easily available from the SDN controller also. Like that paper, I'm also using the same six features. These features are count, protocol type, destination byte, source byte, same service rate and duration (Table 1).

Data preprocessing is required before doing any processing function and I'm doing all preprocessing steps in both training and testing files. Due to high resource consumption and time



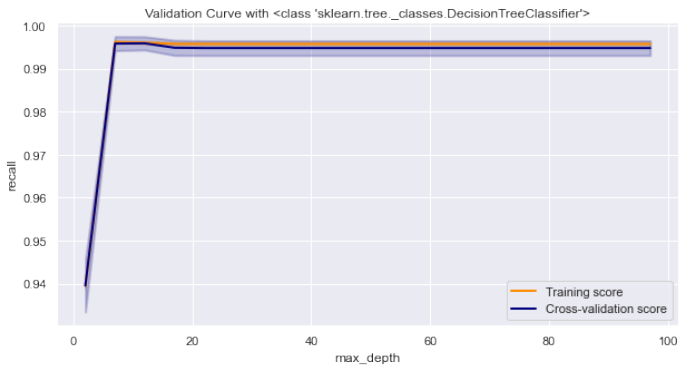
(a) Decision Tree



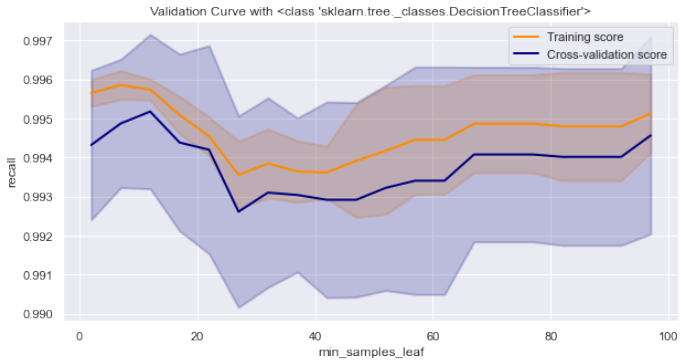
(b) Random Forest

(c) Adaboost

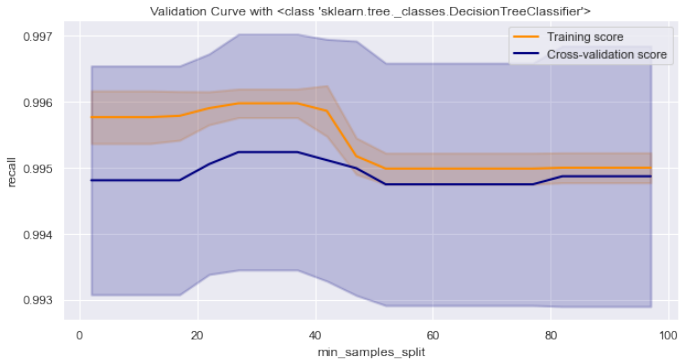
Fig. 3: Learning Curve of three models



(a) max_depth



(b) min_samples_leaf



(c) min_samples_split

Fig. 4: Validation curve of Decision Tree

limits dataset is reduced to 40024 training data and 11211 testing data. All features are in object type and need to convert to float type except for categorical data. There is a total of 3 categorical attributes in the dataset and one hot coding is required for converting them. num-outbound-cmds, a feature in the dataset is have 0 value for all columns and it is not required for the training model. For comparing the performance of features selected based on SDN with the feature that has more relevance in the dataset, the importance score is calculated for all features in the dataset. Dataset is scaled using a standard scaler before calculating the importance score.

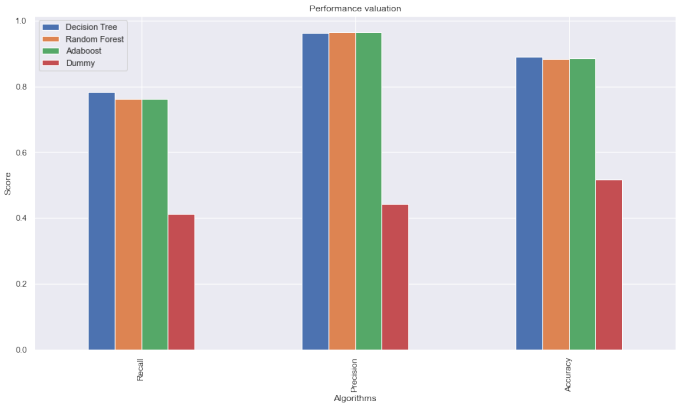


Fig. 5: Comparison of metric values of each algorithm

TABLE I: Feature selection

Feature	Description	Relevance
Count	Total number of connections to same destination	Increase in number of connections to the same destination is a sign of DOS
Protocol type	Protocol of connection	To identify UDP flood, TCP flood and ICMP flood attack
Destination byte	Total byte from destination	Destination shows number byte flow in connection. An abnormal increase in number has the potential for a DOS attack.
Source byte	Total byte from source	Source byte shows number byte flow in connection. An abnormal increase in number has the potential for a DOS attack.
Same service rate	Total percentage of connection to same service	An increase in the number of connections to the same service, DOS chance is high
Duration	total duration of connection total duration of connection	The total duration of connection will be usual high for DoS attack as there is a delay of response in connection.

TABLE II: Hyperparameters

Algorithm	Hyper Parameters	Values
3*Decision Tree	max_depth	39
	min_samples_leaf	3
	min_samples_split	2
3*Random Forest	max_depth	11
	min_samples_leaf	4
	min_samples_split	46
2*AdaBoost	learning_rate	0.70000
	n_estimators	160

TABLE III: Model comparison

Metrics	SDN extractable features	Features selection model
Recall	0.7815246	0.789495
Precision	0.9615287	0.9693851
Accuracy	0.8909999	0.8972437

B. Experiments Run

In this machine learning project, three classification algorithm is used for training and testing the dataset. They are Decision tree, Random Forest and AdaBoost. A decision tree algorithm is a classification algorithm which takes decisions based on nodes and leaves. An algorithm which works with multiple decision trees is the random forest classifier. It also helps to avoid overfitting in the dataset. Adaboost is an ensemble used to boost the performance of classification algorithms.

For hyper parameter tuning, the random search algorithm is used which can provide a better result than grid search due to its randomisation character. K-fold is the cross-validation technique used with a random search algorithm. Hyperparameter details are summarized in TABLE II. The optimal model for the dataset is selected based on the result of the random search algorithm. The validation curve and learning curve are also plotted for choosing the best hyperparameter with less chance of overfitting and underfitting. For comparing the model performance, a baseline model is also declared and it is compared with three algorithms used here.

As per the paper [12], the authors have already mentioned the features which are easily extractable from SDN networks. However, to compare it with features with more impact on the dataset, we need to do the feature selection based on the feature importance score. Using a random-forest classifier with default parameters, the scaled dataset is trained and plotted bar graph based on the features importance score(Fig 2). The performance of the models is evaluated using different metrics.

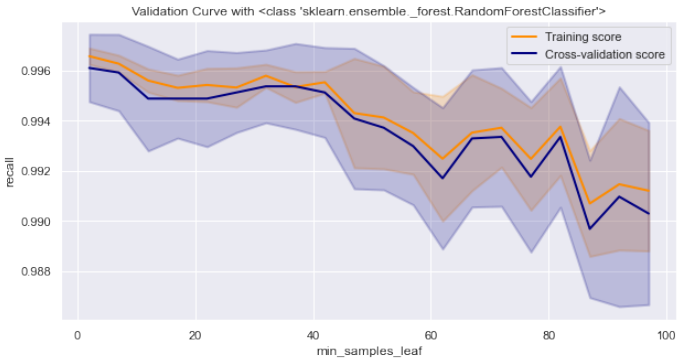
C. Evaluation Metrics

The priority order of metrics to evaluate our model is recall, precision and accuracy. The recall is a predicted DOS attack versus no attacks that happened, and Precision is the percentage of total predicted DOS attacks that are real DOS attacks. I'm giving more priority to recall value because I don't want my model to classify a DOS packet as a normal packet. Precision is given second priority to getting an idea about the false-positive classification of our model. The last metric used in this project is accuracy which is the percentage of total positive prediction(true positive + false positive). It is used to compare the predicted classification of my model with true classification.

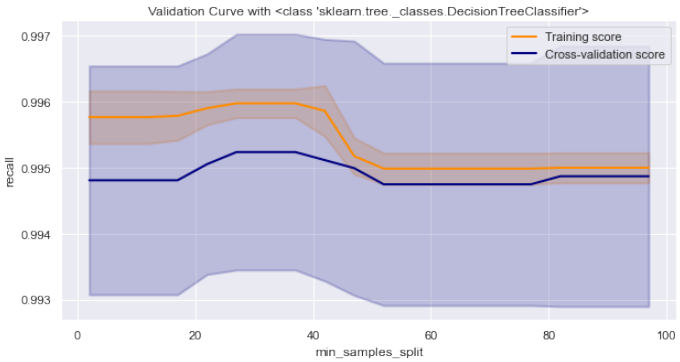
IV. Results and Analysis

All experiments were conducted on a Windows 11 system with processor Intel core I5 9th gen processor and 8GB RAM. Algorithms and methods used in this project are imported from scikit-learn libraries. The performance of each algorithm in the NSL-KDD dataset is evaluated by comparing metric values like recall, precision and accuracy.

(a) max_depth

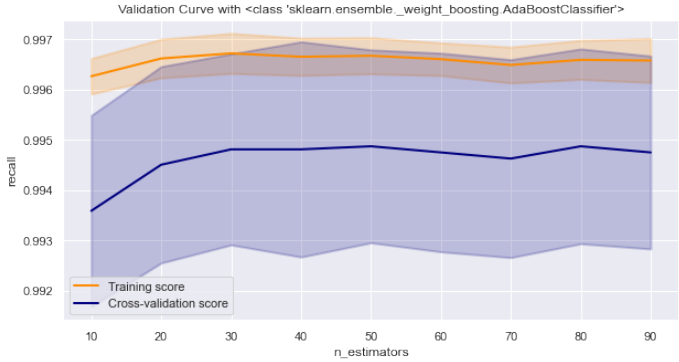


(b) min_samples_leaf

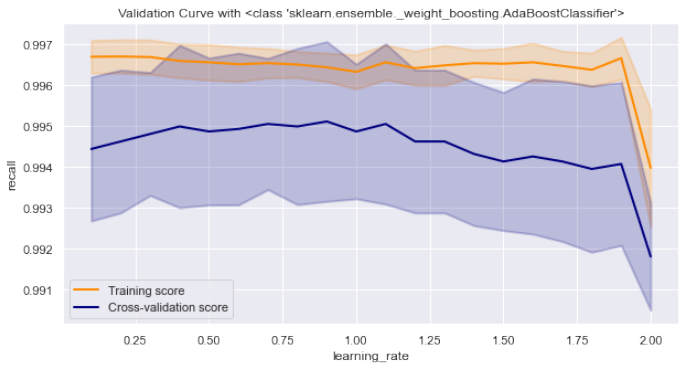


(c) min_samples_split

Fig. 6: Validation curve of Random Forest



(a) n_estimators



(b) learning_rate

Fig. 7: Validation curve of Adaboost

For estimating underfitting/overfitting and selecting the best hyper parameter, the learning curve and validation curve are plotted. The learning curve(Fig 3) of the three models shows that they are underfitting. As already said random forests have less overfitting/underfitting, the figure also proves that it has better plotting. Adaboost has more underfitting than the decision tree. In validation curves of all algorithms look like there is no notable overfitting. But for max-depth of random forest and decision tree training score and validation score is going together. It shows that max-depth parameters don't have an impact on decision-making.

Summarizing Fig. 5, all the models have high precision value and accuracy. But recall values of all three models are below 0.8. Relatively, the primary algorithm of this paper, the decision tree has a better recall value. Even though this model is detecting most of the malicious records, it also marks a few records as legitimate which are malicious. Also, all three algorithms beat the baseline model with a big difference. Comparing decision tree models of SDN features and features selected based on importance score both have a similar metric score(TABLE III). It shows that common features of both scenarios have more impact on decision-making.

In the result of the paper [1], they are getting high precision(92%) and low recall(64%) using the Naive Bayes algorithm with the same dataset. In the paper [13] also they are getting low recall with this dataset. Due to this low recall value, it is not a perfect model. But it has less significant overfitting/underfitting makes it better.

From the above discussion, the NSL-KDD dataset has a low recall value, so choosing a better internet record dataset can have a better recall value. Due to hardware constraints, the dataset is reduced and few machine learning algorithms which have high training time (SVM, KNN) didn't become part of this experiment.

V. Conclusion

This paper is proposing a method of implementing a network intrusion detection system(detect malicious packets in a network) also known as NIDS on the controller of SDN. Packets will be passing through NIDS before reaching the SDN controller. Machine learning classifier algorithms like

decision tree classifier, adaboost and random forest classifier are used in this paper to detect DOS symptoms in the network as a part of NIDS. Results show that the model shows a low recall score for three algorithms but has good precision and accuracy. The decision tree algorithm is slightly better compared to other algorithms. The random forest model shows low underfitting/overfitting among the three. Dataset has low recall value with most of the algorithms, so need to train the model with a dataset which will have a good recall score. Testing with a real-time attack simulator will help to understand the performance of the model.

REFERENCES

[1] Alshamrani, A., Chowdhary, A., Pisharody, S., Lu, D., Huang, D.: A defense system for defeating ddos attacks in sdn based networks. In: Proceedings of the 15th ACM international symposium on mobility management and wireless access. pp. 83–92 (2017)

[2] Ashraf, J., Latif, S.: Handling intrusion and ddos attacks in software defined networks using machine learning techniques. In: 2014 National software engineering conference. pp. 55–60. IEEE (2014)

[3] Banitalebi Dehkordi, A., Soltanaghaei, M., Boroujeni, F.Z.: The ddos attacks detection through machine learning and statistical methods in sdn. The Journal of Supercomputing 77(3), 2383–2415 (2021)

[4] Dao, N.N., Park, J., Park, M., Cho, S.: A feasible method to combat against ddos attack in sdn network. In: 2015 International Conference on Information Networking (ICOIN). pp. 309–311. IEEE (2015)

[5] Dhanabal, L., Shantharajah, S.: A study on nsl-kdd dataset for intrusion detection system based on classification algorithms. International journal of advanced research in computer and communication engineering 4(6), 446–452 (2015)

[6] Elsayed, M.S., Le-Khac, N.A., Dev, S., Jurcut, A.D.: Machine-learning techniques for detecting attacks in sdn. In: 2019 IEEE 7th International Conference on Computer Science and Network Technology (ICCSNT). pp. 277–281. IEEE (2019)

[7] Kokila, R., Selvi, S.T., Govindarajan, K.: Ddos detection and analysis in sdn-based environment using support vector machine classifier. In: 2014 Sixth International Conference on Advanced Computing (ICoAC). pp. 205–210. IEEE (2014)

[8] Mousavi, S.M., St-Hilaire, M.: Early detection of ddos attacks against sdn controllers. In: 2015 international conference on computing, networking and communications (ICNC). pp. 77–81. IEEE (2015)

[9] Rahman, O., Quraishi, M.A.G., Lung, C.H.: Ddos attacks detection and mitigation in sdn using machine learning. In: 2019 IEEE world congress on services (SERVICES). vol. 2642, pp. 184–189. IEEE (2019)

[10] Ye, J., Cheng, X., Zhu, J., Feng, L., Song, L.: A ddos attack detection method based on svm in software defined network. Security and Communication Networks 2018 (2018)

[11] Tavallaee, M., Bagheri, E., Lu, W., Ghorbani, A. A. (2009, July). A detailed analysis of the KDD CUP 99 data set. In 2009 IEEE symposium on computational intelligence for security and defense applications (pp. 1-6). Ieee.

[12] Tang, T. A., Mhamdi, L., McLernon, D., Zaidi, S. A. R., Ghogho, M. (2016, October). Deep learning approach for network intrusion detection in software defined networking. In 2016 international conference on wireless networks and mobile communications (WINCOM) (pp. 258-263). IEEE.

[13] Mohammed, S.S., Hussain, R., Senko, O., Bimaganbetov, B., Lee, J., Hussain, F., Kerrache, C.A., Barka, E. and Bhuiyan, M.Z.A., 2018, October. A new machine learning-based collaborative DDoS mitigation mechanism in software-defined network. In 2018 14th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob) (pp. 1-8). IEEE.