Published by :
http://www.ijert.org

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 9 Issue 09, September-2020

Machine Learning based Intrusion Detection System for Software Defined Networks

Oqbah Ghassan Abbas[1]
[1]Department of Telecommunication Engineering,
Higher Institute for Applied Sciences
and Technology, Damascus, Syria

Khaldoun Khorzom[2]
[2]Department of Telecommunication Engineering,
Higher Institute for Applied Sciences
and Technology, Damascus, Syria

Mohammed Assora[3]
[3]Department of Telecommunication Engineering,
Higher Institute for Applied Sciences
and Technology, Damascus, Syria

***Abstract*— In the last few years, big companies have been depending more and more on Software defined Networks "SDN" to fulfil their needs for programmable networks. But like other networks, SDN has some security problems. Many technologies are used to solve such problems, and machine learning is considered one of the best. Machine learning has proved its ability to find data patterns when other technologies failed. This makes it a perfect choice for anomaly-based detection and intrusion detection system "IDS" in general. In this research, we propose a new anomaly-based IDS that benefits from the ability of SDN to provide statistical features about every flow that passes through the network, and passes these features to a voting system that consists of several machine learning algorithms, which gives the system the ability to study the users' behaviour and predict any possible intrusion. The voting system is trained and tested using NSL-KDD and KDDCup99 datasets and the results shows increasing in accuracy and decreasing in false positive rate.**

***Keywords*— SDN, IDS, Machine Learning, NSL-KDD**

I. INTRODUCTION

The need for programmable networks has drawn the attention of data-centres and big-data companies to new paradigms like software defined network "SDN", which becomes a trend in the last few years. SDN's main goal is to separate the control and the data planes, with the controller being able to generate and modify the flow tables to suit the network services which are running as applications within the controller. But like every new paradigm, SDN faces many challenges, especially with security [4]. Intrusion detection systems "IDS" are considered as one of the best solutions for network security, as it's able to predict and alarm the network administrator about possible intrusions. In the last decade, IDSs are more interested in studying the users' behaviour to predict intrusions. Using new technologies like machine learning, which is perfect for such problems as it is able to extract new information with every interaction between the users and the network, and it is able to find patterns in this information which helps studying the users' behaviour [2]. The problem of using machine learning with network security is the need of data extraction from the network, which increases the overhead in it, but after SDN was introduced, it is possible to benefit from its properties, as it provides statistical features about the traffic that passes the network [1]. In addition to the possibility to add the IDS as a program within the SDN controller [3], which makes it possible to implement an IDS without any additional cost, and without increasing the network latency or decreasing its bandwidth.

The rest of the paper is organized as follows. In section II we give a brief introduction to SDN, IDS and machine learning. Related works are introduced in section III. Our proposed IDS in addition to KDDCup99 and NSL_KDD datasets are introduced in section IV. The performance of our proposed IDS is analysed in section V. Finally, we draw conclusions in section VI.

II. BACKGROUND

In this section we briefly present SDN, IDS and machine learning.

A. Software Defined Networks

Software defined network is a new paradigm in networking that was developed around 2008 [5], where SDN-Controller and SDN-Switches are the main devices in the network, instead of routers and switches. The key concept of SDN is the separation of the control plane and the data plane, as the controller generates the flow tables and distributes them to the switches, which are only responsible of forwarding the data packets. In addition, the switches calculate statistical information about flows and forward this information to the controller. "TABLE I," presents this information [10]

The main benefit of SDN is the ability to add services to the network as applications that run within the controller without adding any devices or modifying the network's topology [3]. This opens a new horizon for improving networks with minimum cost.

TABLE I. FEATURES DESCRIPTION

Feature Name	***Description***
duration	Length (in seconds) of the connection
protocol_type	Type of protocol, e.g. tcp, udp, etc.
src_bytes	Number of data bytes from source to destination.
dst_bytes	Number of data bytes from destination to source.
Count	Number of connections to the same host as the current connection in the past two seconds.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 9 Issue 09, September-2020

B. Intrusion Detection Systems

The intrusion detection is the process of monitoring and analysing any network event to detect any possible intrusion. It can be classified as Host-based when it runs on a network host, or Network-Based when it monitors network packets [6]. For detection strategy, IDS can be classified as signature-based detection, where it looks for patterns that matches known attacks pattern, and anomaly-based detection, where it looks for patterns the doesn't match normal behaviour [6]. Many metrics are used to evaluate the performance of an IDS, but we will focus on accuracy and false positive rate. These two metrics can be calculated from the confusion matrix which has: True Positive (TP) the number of attack samples that were classified correctly, True Negative (TN) the number of normal samples that were classified correctly, False positive (FP) the number of normal samples that were classified incorrectly, and False Negative (FN) the number of attack samples that were classified incorrectly. Accuracy and false positive rate are given in "Equation (1)," and "Equation (2),":

$$\text{Accuracy} = (TP + TN) / (TP + FP + TN + FN) \quad (1)$$

$$FPR = FP / (FP + TN) \quad (2)$$

C. Machine Learning

Machine learning is a subfield of computer science that evolved from the study of pattern recognition and computational learning theory in artificial intelligence. It explores the construction and study of algorithms that can learn from, and make predictions on data. Machine learning is classified as supervised learning when training data is labeled, unsupervised learning when training data is unlabeled and semi-supervised when training data is a mixture between labeled and unlabeled data. Many machine learning algorithms have been developed and improved in the last two decades, from these algorithms we will use the following:

- **Decision Tree (DT):** The algorithm chooses the feature with the highest information gain to be the root node, then the 'gini index' is calculated to find the best partition, then the process is repeated till reaching the specified maximum depth.
- **Random Forest (RF):** A number of decision trees are built depending on a different subset of the dataset for each of them, then the performance of all the trees is averaged to get the final result of the algorithm.
- **XGBoost:** A decision tree is built by using a subset of the data set to get a level1 decision tree, then other subsets will be used consecutively till reaching a specified level. The algorithm could start with many initial decision trees and choose the best one after a certain level.
- **Support Vector Machine (SVM):** Depends on finding the best geometric separator between the classes of the dataset by finding the distances between the points of the dataset. The kernel trick could be used by using kernel functions that measures the distances between the dataset points after projecting them to another dimension.
- **Deep Neural Network (DNN):** Similar to simple neural networks but has multiple hidden layers.

III. PREVIOUS WORK

Vigneswaran and Poornachandran [7], introduce an anomaly-based IDS that works in traditional networks and depends on deep neural network model, they use KDDCup99 dataset to train and test the model. The proposed solution gives an accuracy of 93%. They use KDDCup99 dataset, which suffers from imbalance classes and redundant records which affects the reliability of the results. Ajaeiya and Adalian [8], suggest an anomaly-based IDS that works in SDN and only uses the features provided by it. They compare the results of multiple machine learning algorithms. Random Forest algorithm gives the best results, where the true positive rate is 96.3% and the false positive rate is 0.009. The results show the efficiency of depending on the probabilistic distribution using algorithms like Random Forest. However, in their research, they do not use a standard dataset, which raises some concerns about the validity of their results. Abubakar and Pranggono [9], propose an IDS that works in SDN and consists of a signature-based IDS, and an anomaly-based IDS that depends on deep neural network model and uses NSL-KDD dataset for training and testing. The detection accuracy is 97.4%. However, intrusions detected by the signature-based part are not separated from intrusions detected by the anomaly-based part. So, the accuracy of the anomaly-based part cannot be isolated. Tang and Mhamdi [10], suggest an anomaly-based IDS that works in SDN and only uses the features provided by it. They compare the results of multiple machine learning algorithms. Using a deep neural network with three hidden layers gives the best results with an accuracy of 75.75%. However, the parameters of the used algorithms are not provided. So, it is hard to propose a better tuning for the algorithms.

IV. PROPOSED SOLUTION

In this section we present the architecture of the proposed IDS, and the datasets used to train and test the machine learning part.

A. Architecture

In this section we focus on the design of the IDS. As the input of the IDS is not the whole packets, we need the first part to extract the features. These features can be extracted via the SDN-Switches. As the extracted features do not need preprocessing, we can directly pass them to the second part, which is a machine learning-based voting system that will produce a prediction on whether the flow is normal or abnormal. This part consists of several machine learning algorithms trained using KDDCup99 and NSL-KDD datasets, Support Vector machine algorithm is chosen to cover the possibility of the dataset being geometrically separable. Decision Tree, Random Forest and XGBoost are chosen in case we can obtain better predictions depending on the probabilistic distribution of the features. Deep Neural Network is also chosen because it proved its dominance in finding patterns in datasets. The third part is about adding rules that prevents malicious flows from passing the network. "Fig. 1," shows the architecture of the proposed solution

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 9 Issue 09, September-2020

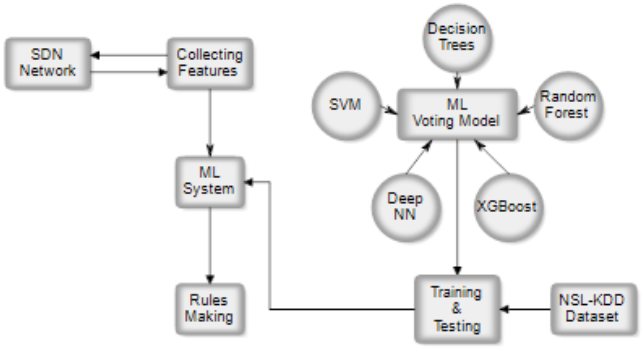

Fig. 1. The architecture of the proposed solution

B. Implementation

As we have seen, the proposed solution consists of 3 parts:

- **Extracting features**: In this part we emulate an SDN network using GNS3. The data plane consists of two hosts connected to an Openvswitch, and OpenDayLight controller that runs on Ubuntu 16.04 Linux distribution, and it is connected to the Openvswitch through openflow1.3 protocol. Every two seconds the controller requests the features collected by the switch using a feature-req message (openflow message), and the switch replies with a feature-reply message containing the features presented in "TABLE I".
- **Voting system**: This part receives the extracted features and runs them through a set of previously trained machine learning algorithms to calculate the probability of these features being associated to a possible intrusion. The system uses the machine learning algorithms that collaborates to calculate the final decision. Every algorithm predicts new samples individually, and the final prediction is made using the "Equation 3"

$$p = (\sum p_i * acc_i)/(\sum acc_i) \quad (3)$$

Where:

p: probability of a test sample being associated to an intrusion.

p_i: probability of a test sample being associated to an intrusion according to model (i).

acc_i: accuracy of model (i) during testing phase.

By using "Equation 3," to calculate the final prediction, we have a voting system that gives higher priority to the algorithms that performed better during the testing phase.

- **Adding Rules:** Every time a flow marked as an intrusion, a new rule is added to the Openvswitch that prevents this flow from passing through the network. It should be noted that adding this part to the system turns it into an intrusion prevention system "IPS" instead of an IDS.

V. EXPERIMENTAL RESULTS

We implemented a machine learning based IDS in SDN for 2-class classification (normal and abnormal), with the aim of increasing the accuracy and decreasing the FPR. All mentioned algorithms are built, trained and tested using Keras and SKLearn libraries within PyCharm IDE. "TABLE II," shows the parameters used to implement each algorithm.

TABLE II. PARAMETERS VALUES FOR MACHINE LEARNING ALGORITHMS

Algorithm	Parameter	Value
Decision Tree	Max_depth	2
Random Forest	N_estimators	100
XGBoost	Max_features	Sqrt
SVM	Objective	binary: hinge
DNN	Gamma	1.0
Decision Tree	Learning rate	0.06

At first, we use KDDcup99 dataset to test these algorithms separately, and the results are presented in "TABLE III,".

From the results in "TABLE III," we notice:

- Decision Tree gives better results than Random Forest and XGBoost, which are considered to be improvements to the Decision Tree algorithm, we justify that by the number of the features we use. KDDcup99 provide 41 features, we only use the 6 features that are compatible with SDN. XGBoost and Random Forest use subsets of the data sets for each sub-tree, and each of these subsets have a maximum of 3 features (best case scenario), making the sub-trees unable to learn enough from these subsets.
- SVM fails to give good results compared to other algorithms, which means our dataset isn't separable geometrically.
- DNN gives the best accuracy, as this kind of algorithms proved its ability to find patterns other algorithms fail to find.
- Compared to [7] and [8] we have better accuracy and FPR.

Because KDDCup99 dataset has several drawbacks [11][12], we also use NSL-KDD dataset to evaluate the machine learning algorithms, and the results are presented in "TABLE IV,"

TABLE III. RESULTS OF THE MACHINE LEARNING ALGORITHMS WITH KDDCUP99

Algorithm	Accuracy %	FPR
Decision Tree	99.4	0.0009
Random Forest	98.54	0.05
XGBoost	99.05	0.01
SVM	89.68	0.012
DNN	99.5	0.0009
Decision Tree	99.4	0.0009

TABLE IV. RESULTS OF THE MACHINE LEARNING ALGORITHMS WITH NSL-KDD

Algorithm	Accuracy %	FPR
Decision Tree	82.72	0.05
Random Forest	77.40	0.03
XGBoost	79.61	0.03
SVM	73.22	0.038
DNN	83.8	0.07
Decision Tree	82.72	0.05

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 9 Issue 09, September-2020

From the results in "TABLE IV," we notice:

- The results pattern of "TABLE III," is repeated in "TABLE IV,", where DNN gives the best accuracy, followed by Decision Tree, Random Forest and XGBoost, and SVM gives the lowest accuracy.
- Compared to [10] we have better accuracy, but compared to [9] their solution gives better accuracy as their solution uses a signature-based IDS in addition to the anomaly-based IDS.
- KDDCup99 gives better results than NSL-KDD, this is because the redundant samples in the KDDCup99 and the imbalance of the number of samples for each class in KDDCup99, which makes NSL-KDD more reliable.

Depending on the results of "TABLE III," and "TABLE IV,", and the fact that NSL-KDD is more reliable than KDDCup99, the voting system is implemented using the models resulted from using NSL-KDD dataset. By applying "Equation 3" to the results shown in "TABLE IV,", we get the results represented in "TABLE V,"

From the results in "TABLE V", we notice that the voting system averages the accuracy of the used algorithms and gives the best FPR compared to the individual results of the used algorithms, as the algorithms with lower accuracy come together to change the vote of the algorithms with the higher accuracy to improve the overall FPR, but this happens at the expense of the overall accuracy.

It is always possible to add or remove any algorithm to the voting system in order to trade-off between the accuracy and FPR.

TABLE V. RESULTS OF THE VOTING SYSTEM

Evaluation Metric	Result
Accuracy	79.6%
FPR	0.03

VI. CONCLUSIONS

In this paper we proposed a machine learning based NIDS for software defined networks. A voting system is implemented using several machine learning algorithms. This system receives the features provided by SDN, so no more equipment is added to the network, and no overhead packets which increase the network's latency and decrease its bandwidth. Finally, the results show an increasing in the overall accuracy to 79.6% and decreasing in FPR to 0.03, with the ability to trade-off one at the expense of the other, by adding or removing algorithms from the voting system.

REFERENCES

[1] A. da Silva and C. Machado, "Identification and selection of flow features for accurate traffic classification in SDN," in 14th International Symposium on Network Computing and Applications, Cambridge, 2015.

[2] A. Ushmani, "Machine learning pattern matching," International Journal of Computer Science Trends and Technology, vol. 7, no. 2, 2019.

[3] D. Hoang and M. Pham, "On software-defined networking and the design of SDN controllers," in 2015 6th International Conference on the Network of the Future (NOF), MONTREAL, 2015.

[4] N. Bindra and M. Sood, "Is SDN the real solution to security threats in networks?," Indian Journal of Science and Technology, vol. 9, no. 32, 2016.

[5] B. Hartpence and R. Rosario, "Software defined networking for systems and network administration programs," The USENIX Journal of Education in System Administration, vol. 2, no. 1, 2016.

[6] N. Sultana and N. Chilamkurti, "Survey on SDN based network intrusion detection system using machine learning approaches," Springer, vol. 12, no. 2, pp. 493-501, 2017.

[7] R. Vigneswaran and P. Poornachandran, "Evaluating shallow and deep neural networks for network intrusion detection systems in cyber security," in 9th International Conference on Computing, Communication and Networking Technologies (ICCCNT), Bengaluru, 2018.

[8] G. Ajaeiya and N. Adalian, "Flow-Based intrusion detection system for SDN," in 2017 IEEE Symposium on Computers and Communications (ISCC), Heraklion, 2017.

[9] A. Abubakar and B. Pranggono, "Machine learning based intrusion detection system for software defined networks," in Seventh International Conference on Emerging Security Technologies (EST), Canterbury, 2017.

[10] T. Tang and L. Mhamdi, "Deep learning approach for network intrusion detection in software defined networking," in International Conference on Wireless Networks and Mobile Communications (WINCOM), Morocco, 2016.

[11] "http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html," 28 10 1999. [Online]. Available: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html.

[12] "https://www.unb.ca/cic/datasets/nsl.html," [Online]. Available: https://www.unb.ca/cic/datasets/nsl.html.