A Machine Learning Approach for Predicting DDoS Traffic in Software Defined Networks

Kshira Sagar Sahoo
Department of Computer Science and Engineering
Madanapalle Institute of Technology
and Science, AP, India, 517325
Email:kshirasagar12@gmail.com

Amaan Iqbal
Department of Computer Science and Engineering
National Institute of Technology,
Rourkela, India, 769008
Email: amaaniqbal2786@gmail.com

Prasenjit Maiti
Department of Computer Science and Engineering
National Institute of Technology,
Rourkela, India, 769008
Email: pmaiti1287@gmail.com

Bibhudatta Sahoo
Department of Computer Science and Engineering
National Institute of Technology,
Rourkela, India, 769008
Email: bibhudatta.sahoo@gmail.com

***Abstract*—Software Defined Networks (SDN) paradigm was introduced to overcome the limitations of the traditional network such as vendor dependencies, inconsistency policies, etc. It becomes a promising network architecture that provides the operators more control over the network infrastructure. The controller also called the operating system of the SDN has the centralized control over the network. Despite all its capabilities, the introduction of various architectural entities poses many security threats to SDN layers. Among many such security issues, Distributed Denial of Services (DDoS) is a rapidly growing attack that poses a tremendous threat to SDN. It targets to the availability of the network, by flooding the controller with spoofed packets. It causes the controller to become paralyzed, and thereby the entire network becomes destabilize. Therefore, it is essential to design a robust DDoS detection mechanism to prevent the control plane attack. In this regard, we have used seven Machine Learning techniques to accurately classify and predict different DDoS attacks like Smurf, UDP flood, and HTTP flood. Experimental results with proper analysis have been presented in this work.**

I. Introduction

The goal of Software Defined Networks (SDN) is to decouple the control plane from the forwarding plane [1]–[3]. Additionally, this architecture allows more flexible network management to the network operator [4]–[6]. All the routing decision and controlling mechanism are controlled by a central device called controller. The controller sends command to the forwarding devices such as router and switch for managing the data packets. In SDN, the control plane may consist of one or more than one controller depending upon the size and usage of the network. In control plane, the controller provides a distributed policy information consistently throughout the network through a standard protocol. The OpenFlow is a well known protocol that used to make a secured connection among the network devices and the controller to determine the best path for the various application running on the top of the controller. In spite of numerous benefits provided by SDN, security is still a major concern among the enterprises and research communities. The delivery of full packet information from the data plane to the control plane is not supported especially in OpenFlow based SDN architecture. Meanwhile, network intrusion prevention system for SDN applications requires a full packet inspection against every single packet passing through the data plane. So, enabling the full packet delivery to the controller is one challenging issue for SDN.
Furthermore, the fundamental architectural changes and the introduction of various design entities pose new security concerns to the SDN platform. Moreover, the Distributed Denial-of-Service (DDoS) is a rapidly growing attack that poses a tremendous threat to the future generation Internet technology. The multitude of this attack is quickly becoming more and more complex. With the advancement in network technologies, SDN architecture increase the chances to defeat DDoS attacks cause the several points of the network. On one way, the centralized control provisioning and enhanced visibility makes SDN, easier to detect the DDoS Attacks. But, on another hand, it becomes a victim of DDoS attacks due to the potential vulnerabilities exist across various SDN layers. Therefore, an efficient detection technique must be designed for a secured SDN control plane. In order to detect attacks, Intrusion Detection Systems (IDS) follow two different approaches: signature-based or anomaly based detection [7]. The anomaly based IDS is based on the concept of a baseline for network behavior. The Machine Learning (ML) approach that helps in implementing the network behavior that can learn from historical data and provide a prediction for the upcoming packets based on the training data. ML based methods have shown notable potential in the classification of the legitimate traffic and the attack traffic. Also, these techniques do not need to check the packet payload, rather they require a particular set of features of the incoming flows like source and destination IP addresses and port addresses, number of packets contain in the flows etc. [8]. Hence, a lower computational cost incurs by ML techniques as compared to Deep Packet Inspection (DPI) based techniques. Although the experimental outcomes achieved by different researchers show significant improvements in the detection

978-1-7281-0259-7/18/$31.00 ©2018 IEEE
DOI 10.1109/ICIT.2018.00049

of DDoS attack in SDN [9], [10], the anomaly detection problem is always an open-ended research area. These works usually need a large volume of network traffic data, large dimensional training dataset in a constantly changing network environment. Besides the relevance of choosing the most suitable features from the dataset, setting the performance parameters of the implemented algorithms with the optimal value is another important factor, which influences to design an efficient detection model. The main contribution of this paper is given below:

- This work leverages ML techniques for classifying attack and non-attack traffic.
- Proposing an effective solution for predicting DDoS attack using seven different widely used ML algorithms.
- Validate our proposed system through numerous simulations.

The rest of the paper is organized as follows: Section II describes the related work, Section III presents an insight into the ML algorithms applied in the work. Then the proposed mechanism is illustrated in Section IV. Section presents the experimental results and discusses the findings. Finally, Section VI presents the conclusions and future works.

II. Related Work

A wide range of techniques have been introduced to tackle DDoS flooding attack in the conventional networks. However, a limited study has done in the area of security challenges of SDN environments. A comprehensive survey on SDN security has presented by Ali *et al.* [11]. This paper figured out a number of challenges and also provide solutions to curb the network threats. The anomaly detection mechanism has carried out through OpenFlow and sFlow by Giotis et al. [12]. For detection purpose, statistical entropy method has used. Kokila et al. [13], has used Support Vector Machine for detecting the attack traffic but used the traditional features of the traffic. In [9], authors suggest a new detection scheme using Self-Organizing Map (SOM) with six different attributes. The collection of flow entries from the switch has performed in a pre-determined time interval. Authors in [14], uses various sampling rates to capture the attack traffic. An attack detection trigger technique at the initial phase of the attack was proposed in [15]. Along with the anomaly detection, they have proposed a packet traceback mechanism using a trained Back Propagation Neural Network (BPNN) model to find the result. Various ML algorithms have used by Ashraf *et al.* to handle DDoS in SDN [16]. Although there are various research work has carried out on SDN, DDoS attack detection is an open-ended problem [17]. Hence, in this paper, by utilizing the SDN controller capability we adopt the different ML approaches with different traffic features to curb the potential attack. The proposed method can be used by the controller to define security rules to block an entire subnet for possible attack by the attacker.

III. Design Principle

To predict the possible DDoS attack to SDN controller, we have used machine learning (ML) technique which relies on the historical attack data. We have used seven different ML algorithms for predicting the attack host, and compare their performance in terms of different performance metrics. Figure 1 describe an overview of the proposed detection scheme.

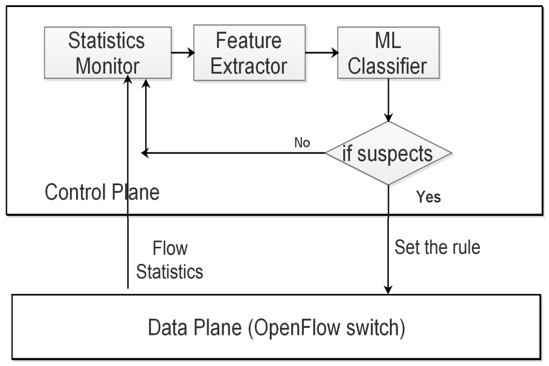

Fig. 1. Flowchart of the proposed scheme

Here, we discuss each module in details.

- *Statistics Monitor:* The statistics monitor sends Flow start request to the OF switches and as a result, receives the flow statistics information.
- *Feature extractor:* This module is responsible for extracting the features that are significant to DDoS attack detection. Next, all extracted feature will feed to the classifier.
- *ML Classifier:* The ML classifier, classify the traffic as per the training information. To obtain a classifiers for predicting possible attack hosts accurately, historical data is required to train detection model. The training phase helps the model to learn and obtain a better accurate result.

In this approach, we have not fixed the ML classifier. Any learning method can be used as per the requirement. The Algorithm 1 summarizes the proposed approach

Algorithm 1 SDN_ML_DDoS_PROC

```
1: Choose the ML classifier
2: Train the ML classifier with the selected features within the controller
3: OF Switches ← send Controller($Flow\_Start\_Request$)
4: Collect flow statistics $DDoSDetectionProcessStart$
5: if ($Classifier\ predict\ attack$) then
6:     Action $Delete\_Flow\_Entry$
7: else
8:     Allow flow to access the host
9: end if
```

IV. Machine Learning Algorithms

There are basically two types of machine learning techniques are used such as supervised learning and unsupervised learning algorithms. In supervised learning algorithms, each input data is associated with a class which is called label. During testing the machine predicts the class of input data based on the training sample. This is called supervised because we know the class of training sample during learning phase of the machine and the output of the algorithm is the trained classes. Now, we will discuss some ML algorithms used in this work.

- *k-Nearest Neighbor (kNN):* It is a non-parametric and lazy learning classification algorithm. The term lazy indicates that it does not make generalization using the training data. Let's each sample in the set has n attributes considered to be independent variables, and combine to form an n-dimensional vector. $\vec{x} = (x_1, x_2, ..., x_n)$. Another attribute say y, depends on the other n attributes. Suppose a set of such vector V; given together with their corresponding classes: $x^{(i)}, y^{(i)}, \quad for\ i = 1, 2, ..., V$.
  The idea behind kNN is to identify k samples in the training set whose independent variables (x) are related to new samples (u). Then use these k samples to classify the new sample into a class (v). When we discuss about neighbors it implies that there is a distance (dissimilarity) measure that can be calculated between independent variables. The most common measure of distance is Euclidean distance, is described below.
  $$d(x, u) = \sqrt{\sum_{i=1}^{n}(x_i - u_i)^2} \qquad (1)$$
- *Naïve Bayes (NB):* This algorithm is based on Bayes' theorem. It assumes that the presence of a feature in a class is completely unrelated to the any other features present in the class. It is easy to build Naïve Bayes model and it is helpful in large datasets. It performs well than highly sophisticated classification techniques. It can be applied to inferential statistics and decision making problems that deals with probability inference. The Equation 2 shows the probability calculation of a class belonging to a sample.
  $$P(c|x) = \frac{P(x|c).P(c)}{P(x)} \qquad (2)$$
  Here, $P(c|x)$ is the posterior probability of class,$P(c)$ is the prior probability of class, $P(x|c)$ is the probability of likelihood of a given class and $P(x)$ is the prior probability of the predictor.
- *Support Vector Machine (SVM):* SVM is considered as a highest accuracy based classifier in the field of ML. It is a set of related supervised learning methods used for classification [18]. Given a set of training samples, each sample labeled as different categories. An SVM algorithm develops a model that predicts whether a new sample falls into one of the categories.
  Let's a given training data set $S = \{(\vec{x}_1, y_1), (\vec{x}_2, y_2), ..., (\vec{x}_n, y_n)\}$, where $\vec{x}_i \in \mathcal{R}^n$ and $y \in \{+1, -1\}$. From the inputs, SVM draws an optimal hyperplane $\mathcal{H}$ that separates the data into different classes. The hyperplane $\mathcal{H}$ can be defined as:
  $$\vec{x}_i \in \mathcal{R}^n : (\vec{w}, \vec{x}) + b = 0, \vec{w} \in \mathcal{R}^n, b \in R \qquad (3)$$
  The algorithm is based on finding the hyperplane which gives the maximum distance of separation between training samples using the following function.
  $$f(\vec{x}) = sign\{(\vec{w}, \vec{x}) + b\} \qquad (4)$$
  For detecting attacked traffic, two linearly separable data is considered. Hence, the optimal hyperplane would be:
  $$y_i\{(\vec{w}, \vec{x}) + b\} \geq 1, \ s.t.\ i = 1, ..., n \qquad (5)$$
- *Random Forest (RF):* Created by LEO Breiman and Adele Cutler,these classifiers join different decision trees to anticipate new unlabeled information, every decision tree is available in the forest and its quality is subject to the quantity of trees in the forest [19]. For every tree,random attributes are chosen, each number of trees speaks to a single forest and each forest is a predation class for new unlabeled information. In this algorithm, random feature selection is done for each individual tree.Then a collection learning algorithm is utilized for classification and prediction of the outputs in light of an individual number of trees. Utilizing this strategy, numerous classification trees are created, and every free tree is built by an alternate piece of the general dataset. After each tree is classified in an unlabeled class another question will be actualized under each tree vote in favor of choice. The forest picked as the victor depends on the most noteworthy number of votes recorded.
- *Linear Regression (LR):* Linear regression is commonly used in predictive analysis. It is a model that assumes a linear relationship between the input and a output variable. Specifically, the output can be calculated from a linear combination of the input variables. For a single input variable, LR is referred to as simple linear regression, whereas in case of multiple input variables, it refers to as multiple linear regression. In LR we take the output of the linear function and fix the value within the range of $[0, 1]$ using the sigmoid function.

V. Result Analysis

The performance of the classifier has evaluated based on primary performance indicators based on confusion matrix shown in Table I. Where, TP and FP denotes True Positive and False Positive respectively. Similarly, TN and FN represents the True Negative and False Negative. Here, we will discuss some performance measures used in this work.

TABLE I
CONFUSION MATRIX

		Predicted	
		positive	Negative
Actual	Positive	TP	TN
	Negative	FP	TN

- Accuracy : It's the measure of classified dataset features to the total dataset, which can be described in Equation 6.
  $$\frac{TP + FN}{TN + TP + FN + FP} \qquad (6)$$

- Precision : This measure is predominantly used when the dataset is imbalanced. Its the ratio of correctly classified data to the sum of correctly classified and incorrectly classified.

$$\frac{TP}{TP+FP} \tag{7}$$

- Recall : It is the ratio of correctly classified data to the sum of significant attacks. It is additionally called positive sensitivity value, which can be figured by the Equation 8.

$$\frac{TP}{TP+FN} \tag{8}$$

The above described classification algorithms, run on a machine having 4GB RAM with Ubuntu 14.04 with 64 bit Operating System. To evaluate the accuracy of different ML techniques, we have used Python based Sci-kit learn tool. For training and testing we use a publicly available modern DDoS dataset developed by Mouhammd *et al.* [20]. This dataset is a labeled and non-redundant, which contains five classes, 27 features and 21,60,668 records. Table II, depicts the distributions of the various DDoS records in the dataset. In their work, for accurate prediction, authors have used various ML techniques such as MLP, RF and NB. They have showed that the overall accuracy level of RF and MLP is 98.02% and 98.63% respectively. In machine learning, usually the data set has split into two subsets, such as training and testing data. Then we fit the model on the train data, in order to make predictions on the test data. In this experiment, to train the model, we run the classifier for 30 times with different combinations of training set and testing set such as 70:30, 80:20, and 90:10. The prediction accuracy and total time taken by different classifiers are tabulated in Table III. The classification accuracy of classifier depends upon the different parameter settings. For example, the parameter like C and γ plays an important role for better accuracy in SVM. For SVM, after a number of observations we found that the classification accuracy is higher when the γ is set to 0.025 and C is set to 1. Similarly for DT the max_depth parameter has set to 2 and for RF the $n\_estimators$ has fixed to 20. As compared to previous result [20], in this experiment Linear Regression (LR) shows a significant improvement in terms of prediction accuracy. In all split ratios both KNN and ANN exhibit relatively better accuracy than other ML techniques. As far as training/testing time concerned NB is taking lesser time whereas KNN and SVM both are taking higher time. On the other hand, RF takes lesser time i.e. 176.89 sec compared to LR classifier. Further, the change in the training/testing split ratio does not bring much significant change in the prediction accuracy, due to the nature of the dataset.

TABLE II
DISTRIBUTION OF ATTACK RECORDS IN THE DATASET

Attack Type	Number of Records
UDP Flood	201344
HTTP Flood	4110
Smurf	12590
SiDDoS	6665

TABLE III
PREDICTION ACCURACY(IN %) AND TOTAL TIME (IN SEC.) TAKEN BY DIFFERENT ML CLASSIFIERS

ML	70:30		80:20		90:10	
methods	Accuracy	Time	Accuracy	Time	Accuracy	Time
LR	98.652	188.50	98.643	200.55	98.617	225.84
kNN	98.633	4505.58	98.664	5522.28	98.656	4564.16
NB	97.640	1.24	97.416	1.42	97.451	1.78
DT	97.241	162.75	97.255	238.67	97.242	224.61
RF	98.409	176.89	98.495	205.5	98.453	264.52
ANN	98.645	111.15	98.616	126.13	98.656	180.59
SVM	98.153	2306.19	98.175	2511.92	98.224	2592.37

Next, we demonstrate the confusing matrix of the LR classifier. Table IV, Table V, and Table VI show the confusion matrix of LR classifier with 90:10, 80:20, 70:30 split ratio respectively.

TABLE IV
CONFUSION MATRIX (90:10)

	HTTP-Flood	Normal	SiDDoS	Smurf	UDP Flood
HTTP-Flood	392	5	29	0	0
Normal	5	193491	16	0	0
SiDDoS	0	38	636	0	0
Smurf	4	799	36	410	0
UDP Flood	0	2056	0	0	18150

TABLE V
CONFUSION MATRIX (80:20)

	HTTP-Flood	Normal	SiDDoS	Smurf	UDP-Flood
HTTP-Flood	707	7	39	0	0
Normal	8	386950	43	0	0
SiDDoS	0	87	1304	0	0
Smurf	2	1686	62	801	0
UDP-Flood	0	3928	0	0	36510

TABLE VI
CONFUSION MATRIX (70:30)

	HTTP-Flood	Normal	SiDDoS	Smurf	UDP Flood
HTTP-Flood	1250	8	62	0	0
Normal	20	580762	46	0	0
SiDDoS	0	105	1931	0	0
Smurf	9	2440	103	1183	0
UDP Flood	0	5946	0	0	54336

It can be observed from Figure 3 and Figure 2 that all the classifiers achieved higher precession and recall value for both normal and UDP-traffic class. It is also observed that classifying the Smurf class is the most challenging task for all classifiers. Because, in Smurf attack, a large volume of ICMP echo packets are being sent, which is hard to classify as normal or abnormal traffic.

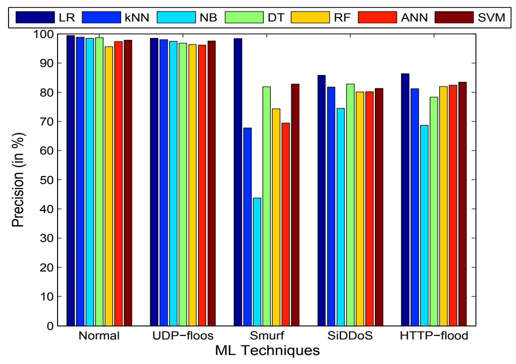

Fig. 2. Precession results of different classifiers

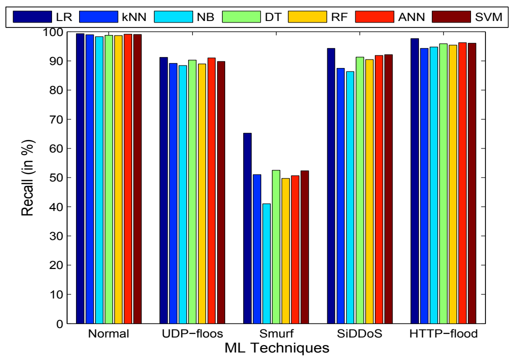

Fig. 3. Recall results of different classifiers

Among seven classifier LR achieved high accuracy, precision, recall results, while NB is showing worst result. RB and NB shows poor result for the Smurf class, on the other hand, LR, DT, and SVM achieved a high precision rate than others. Moreover, DT, RF takes lesser time compared to LR. In general, it can be conferred that there is a trade-off between prediction accuracy and total execution time, which should be taken into consideration while selecting a ML technique for DDoS traffic detection in SDN. Leveraging the classifier's outcome, we can set different security rules on the controller to check the potential attackers by blocking a subnet of the network.

VI. Conclusion

In this paper, we have used ML approaches to predict the DDoS attack in SDN network on a different set of traffic features. Leveraging the usage of different ML algorithms, the security rules defined by the controller can check the malicious attack. Experimental results showed that the selection of proper ML algorithms could help to predict the attack accurately and define the security rules for the potential attacker. The average prediction accuracy achieved by LR is 98.652% which means this classifier can predict the malicious traffic accurately. On the other hand, RF achieved 98.409% with less execution time than LR. As the future work, we will focus on higher testing accuracy for Smurf and UDP-Flood traffic and will compare with other ML techniques.

References

[1] Y. Jarraya, T. Madi, and M. Debbabi, "A survey and a layered taxonomy of software-defined networking," *IEEE communications surveys & tutorials*, vol. 16, no. 4, pp. 1955–1980, 2014.

[2] K. S. Sahoo, S. Mohanty, M. Tiwary, B. K. Mishra, and B. Sahoo, "A comprehensive tutorial on software defined network: The driving force for the future internet technology," in *Proceedings of the International Conference on Advances in Information Communication Technology & Computing*. ACM, 2016, p. 114.

[3] P. Maiti, J. Shukla, B. Sahoo, and A. K. Turuk, "Qos-aware fog nodes placement," in *2018 4th International Conference on Recent Advances in Information Technology (RAIT)*. IEEE, 2018, pp. 1–6.

[4] ——, "Mathematical modeling of qos-aware fog computing architecture for iot services," in *Emerging Technologies in Data Mining and Information Security*. Springer, 2019, pp. 13–21.

[5] M. Tiwary, D. Puthal, K. S. Sahoo, B. Sahoo, and L. T. Yang, "Response time optimization for cloudlets in mobile edge computing," *Journal of Parallel and Distributed Computing*, vol. 119, pp. 81–91, 2018.

[6] K. S. Sahoo, D. Puthal, M. Tiwary, J. J. Rodrigues, B. Sahoo, and R. Dash, "An early detection of low rate ddos attack to sdn based data center networks using information distance metrics," *Future Generation Computer Systems*, vol. 89, pp. 685–697, 2018.

[7] A. K. Sahoo, K. S. Sahoo, and M. Tiwary, "Signature based malware detection for unstructured data in hadoop," in *Advances in Electronics, Computers and Communications (ICAECC), 2014 International Conference on*. IEEE, 2014, pp. 1–6.

[8] Z. A. Qazi, J. Lee, T. Jin, G. Bellala, M. Arndt, and G. Noubir, "Application-awareness in sdn," in *ACM SIGCOMM computer communication review*, vol. 43, no. 4. ACM, 2013, pp. 487–488.

[9] R. Braga, E. Mota, and A. Passito, "Lightweight ddos flooding attack detection using nox/openflow," in *Local Computer Networks (LCN), 2010 IEEE 35th Conference on*. IEEE, 2010, pp. 408–415.

[10] Y. Zhang, "An adaptive flow counting method for anomaly detection in sdn," in *Proceedings of the ninth ACM conference on Emerging networking experiments and technologies*. ACM, 2013, pp. 25–30.

[11] S. T. Ali, V. Sivaraman, A. Radford, and S. Jha, "A survey of securing networks using software defined networking." *IEEE Trans. Reliability*, vol. 64, no. 3, pp. 1086–1097, 2015.

[12] K. Giotis, C. Argyropoulos, G. Androulidakis, D. Kalogeras, and V. Maglaris, "Combining openflow and sflow for an effective and scalable anomaly detection and mitigation mechanism on sdn environments," *Computer Networks*, vol. 62, pp. 122–136, 2014.

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

[14] R. Miao, M. Yu, and N. Jain, "Nimbus: cloud-scale attack detection and mitigation," in *Acm sigcomm computer communication review*, vol. 44, no. 4. ACM, 2014, pp. 121–122.

[15] Y. Cui, L. Yan, S. Li, H. Xing, W. Pan, J. Zhu, and X. Zheng, "Sd-anti-ddos: Fast and efficient ddos defense in software-defined networks," *Journal of Network and Computer Applications*, vol. 68, pp. 65–79, 2016.

[16] J. Ashraf and S. Latif, "Handling intrusion and ddos attacks in software defined networks using machine learning techniques," in *Software Engineering Conference (NSEC), 2014 National*. IEEE, 2014, pp. 55–60.

[17] K. S. Sahoo, M. Tiwary, and B. Sahoo, "Detection of high rate ddos attack from flash events using information metrics in software defined networks," in *Communication Systems & Networks (COMSNETS), 2018 10th International Conference on*. IEEE, 2018, pp. 421–424.

[18] S. S. Keerthi, S. K. Shevade, C. Bhattacharyya, and K. R. Murthy, "A fast iterative nearest point algorithm for support vector machine classifier design," *IEEE transactions on neural networks*, vol. 11, no. 1, pp. 124–136, 2000.

[19] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[20] M. Alkasassbeh, G. Al-Naymat, A. Hassanat, and M. Almseidin, "Detecting distributed denial of service attacks using data mining techniques," *International Journal of Advanced Computer Science and Applications*, vol. 7, no. 1, 2016.