IIETA International Information and Engineering Technology Association
Advancing the World of Information and Engineering

Revue d'Intelligence Artificielle
Vol. 36, No. 2, April, 2022, pp. 233-240
Journal homepage: http://iieta.org/journals/ria

# Performance Evaluation of SDN DDoS Attack Detection and Mitigation Based Random Forest and K-Nearest Neighbors Machine Learning Algorithms

Mayadah A. Mohsin*, Ali H. Hamad

Department of Information and Communication Engineering, University of Baghdad, Baghdad 10011, Iraq

Corresponding Author Email: mayadahabdalmohsin@gmail.com

https://doi.org/10.18280/ria.360207

**Received:** 12 January 2022
**Accepted:** 16 March 2022

***Keywords:***
*distributed denial of service, K-nearest neighbor, machine learning, Mininet, random forest, RYU controller, SDN security, software-defined network*

**ABSTRACT**

Software-defined networks (SDN) have a centralized control architecture that makes them a tempting target for cyber attackers. One of the major threats is distributed denial of service (DDoS) attacks. It aims to exhaust network resources to make its services unavailable to legitimate users. DDoS attack detection based on machine learning algorithms is considered one of the most used techniques in SDN security. In this paper, four machine learning techniques (Random Forest, K-nearest neighbors, Naive Bayes, and Logistic Regression) have been tested to detect DDoS attacks. Also, a mitigation technique has been used to eliminate the attack effect on SDN. RF and KNN were selected because of their high accuracy results. Three types of network topology have been generated to observe the effectiveness of proposed algorithms on different network architectures. The results reveal that RF performs better than KNN in a single topology, and both have close performance in other topologies.

## 1. INTRODUCTION

SDN design aims to address the shortcomings of the existing traditional network architecture through a new architectural design that is based on the decoupling of the control plane from the data plane, allowing the controller to be directly programmable and have full management. So, the SDN environment provides great reliability, simplicity, and flexibility for network management [1]. But SDN poses additional security issues due to the centralized controller, which is a vulnerable point that attackers target.

One of the most common attacks on SDN networks is the Distributed Denial of Service (DDoS) attack. This attack aims to make network services unavailable to legitimate users by sending large amounts of malicious traffic to exhaust the network resources. It launches his attack by taking control of several compromised hosts and making them part of a network called a botnet [2, 3]. This attack should be detected early before bringing the entire network down by causing damage to the controller. So, SDN should be equipped with efficient detection and mitigation techniques to ensure the safety of its resources and data in the face of various DDoS attacks.

Several detection techniques have been used to defend the SDN network against DDoS attack: entropy, machine learning algorithm, Traffic pattern analysis, intrusion detection system (IDS) like SNORT [4]. Machine Learning (ML) is mainly a classifier that classifies network traffic into normal and attacks. First, it builds a classifier model based on sample data called the training data set. Then, it tests the model in order to make predictions [5, 6]. There are many mitigation techniques provided through the use of OpenFlow protocol in SDN networks: drop packets, block port, redirection, control bandwidth, deep packet inspection, network reconfiguration, and topology change; each solution has its advantages and disadvantages [7]. The most fast and simple mitigation technique is the drop packet and the block ports. They completely block the attack sources, which may result in dropping the legitimate traffic if the legitimate host is compromised or in the case of a false alarm. So, a timeout is added to the blocking rule to ensure a non-permanent block for a real user in this work.

In this context, this paper presents a DDoS attack detection and mitigation technique in an SDN environment using two machine learning algorithms. The main contributions of this paper can be summarized as follow:

- Apply machine learning algorithms on three testing environments: single, linear with one controller, and linear with multi-controller SDN networks.
- Generate SDN dataset using Mininet emulator for each scenario.
- Evaluate and compare four different ML algorithms (RF, KNN, NB, LR) accuracy.
- Implement a machine learning model to detect DDoS attacks. Two ML algorithms have been selected: RF and KNN.
- Implement attack mitigation technique in both algorithm types.

The rest of this paper is organized as follow: in section 2, a related work for DDoS detection techniques based on machine learning in SDN are introduced. In section 3, a theoretical background of the machine learning techniques is discussed. Section 4 presents the proposed algorithms implementation and environment setup. Section 5 shows the results and discussion. Finally, a conclusion introduced in section 6.

## 2. RELATED WORKS

Several works have been done in the scope of DDoS detection and mitigation in SDN network using machine learning techniques we study some of these works we found

interesting.

Ye et al. [8] Proposed a DDoS attack detection system based on an SVM algorithm that classifies traffic based on 6-tuple characteristics values related to DDoS attack that previously gathered from switch flow table. Sahoo et al. [9] Compare seven machine learning techniques to protect the SDN against DDoS attacks, KNN, NB, SVM, RF and LR. Experimental results showed that LR and RF have the most accurate detection. However, while the results are presented in the publication, the tools used for implementation and simulation are not described in full, and hence this study cannot be compared to our work. Prakash and Priyadarshini [10] proposed an evaluation for ML algorithms such as NB, KNN and SVM to defend SDN in the network layer against DDoS attacks, the work results show that the KNN performed better than other algorithms. Le et al. [11] tested six different ML-algorithms including RF, NB, KNN, SVM, Multiple Layer Perceptron (MLP), and DT, to classify DDoS attacks. The work proved that DT and NB have high accuracy, fast processing time and consume less resource compared to other algorithms. Karan et al. [12] implement SNORT intrusion detection system and ML algorithm to detect DDoS attack in SDN environment. The work compares SVM and Deep Neural Network (DNN) for their detection performance. the results found the DNN is having the higher accuracy. Rahman et al. [13] evaluated some machine learning algorithms which are SVM, RF, J48 and KNN for DDoS detection and mitigation in SDN network. The work found that J48 algorithm is the best for the system. Polat et al. [14] Proposed machine learning models supported with feature selection methods to detect DDoS attacks, the work tested different ML algorithms (SVM, NB, ANN KNN) and found that the KNN classifier achieved the highest accuracy rate in DDoS attack detection. Santos et al. [15] compare between four machine learning techniques to solve the problem of DDoS attacks in the SDN environment, the result found that RF algorithm had the best accuracy, and DT had the best processing time. Table 1 shows a comparative study for the related works compared to our work.

**Table 1.** Comparative study of related works

| Ref. | Detection algorithm | Dataset type | Mitigation | Controller | Network Topology | Network Scale |
|---|---|---|---|---|---|---|
| **[6]** | SVM | No dataset Flow table features extraction | × | Floodlight | Linear | 5 hosts |
| **[7]** | SVM/KNN/NB/RF/LR | Public | × | × | × | × |
| **[8]** | NB/KNN/SVM | Generated by Simulation | Drop packet | Floodlight | Linear | Not mentioned |
| **[9]** | RF/DT/SVM/KNN/MLP | Generated by Simulation | × | RYU | Single | 3 hosts |
| **[10]** | SVM/DNN | Public KDD dataset | × | RYU | Single | 4 hosts |
| **[11]** | J48 | Generated by Simulation | Block port with timeout | RYU | Single | 5 hosts |
| **[12]** | SVM/NB/ANN/KNN | Generated by Simulation | × | POX | Single | 6 hosts |
| **[13]** | SVM/MLP/DT/RF | Generated by Simulation | × | POX | Single | 6 hosts |
| **Our paper** | RF/KNN/NB/LR | Generated by Simulation | Block port with timeout | RYU | Single, linear, linear with multi-controller | 64 hosts /8 switches-64 hosts /8 switches-64 hosts |

## 3. BACKGROUND

Machine learning algorithms be divided into four categories: supervised, unsupervised, semi-supervised, and reinforcement learning. Table 2 explains the main differences between them. In supervised learning algorithms, the input is combined with a label that represents a classification class and during the testing phase, the machine predicts the class of that input data based on the training sample. This is called supervised because we pre-trained the model on a known classed training sample [16]. In this work the supervised learning has been used since the generated data is labeled to 1 and 0 to indicate a normal or attack flow.

**Table 2.** Machine learning techniques

| ML class | Learning | Example |
|---|---|---|
| **Supervised** | Labeled dataset | SVM, DT, KNN… |
| **Unsupervised** | Unlabeled dataset | K-means, SVD, PCA |
| **Semi-supervised** | Some labeled and other unlabeled | linear regression, logistic regression |
| **Reinforcement** | Trial and error (no dataset) | Markov Decision Support, Q-learning |

Next is a further explanation of some of the supervised algorithms that have been used in this work:

*Random Forest (RF):* Also known as a random decision forest, because it is constructs from many decision trees (forest). It is one of the popular supervised learning algorithms that used both in classification and regression problems. The accuracy of the algorithm depends on the number of trees used [17]. To build the final decision about the new unclassified sample each decision tree will make a vote about this sample and the final prediction is the class with the most votes among the tree outputs.

*K-nearest neighbor (KNN):* It is a simple, easy-to-implement supervised machine learning algorithm that can be used to solve both classification and regression problems. It assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories [18].

*Naive Bayes (NB):* It is one of the simplest supervised ML algorithms that used to solve the classification problem. It based on the Bayes theorem that uses conditional probability to describe the relationship between statistical quantities. Bayes theorem uses conditional probability of an event which based on the assumption that predictors or input features are

independent to each other [19].

*Logistic Regression (LR):* it is a supervised ML algorithm used for the classification problem based on the logistic function or sigmoid function which is a mathematical function used to map the predicted values to probabilities and have value between 0 and 1, different from linear function that is used in linear regression that can gives value above 1 or below 0 [20]. Logistic regression uses a threshold value, so as values above the threshold tends to 1, and a value below the threshold tends to 0.

## 4. PROPOSED ALGORITHMS IMPLEMENTATION AND EXPERIMENTAL ENVIRONMENT SETUP

In this work four machine learning algorithms has been proposed to secure SDN networks from DDoS attacks, RF, KNN, NB and LR. The proposed system works as illustrated in Figure 1, the controller extracts the network traffic characteristics from switch flow table and uses it as input to the selected ML classifier to identify the traffic as normal or attack. The experimental setup done on an HP laptop with Core i7 CPU and 16 GB RAM under Linux Ubuntu 16.04 LTS OS. The testbed has been done using Mininet emulator and RYU controller. Open VS witches with OpenFlow protocol version 1.3 have been used in this work.

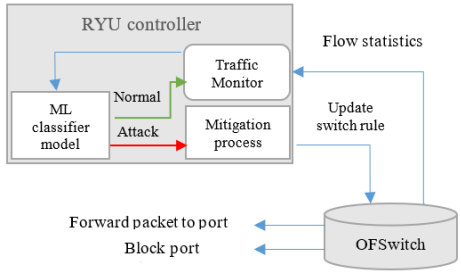
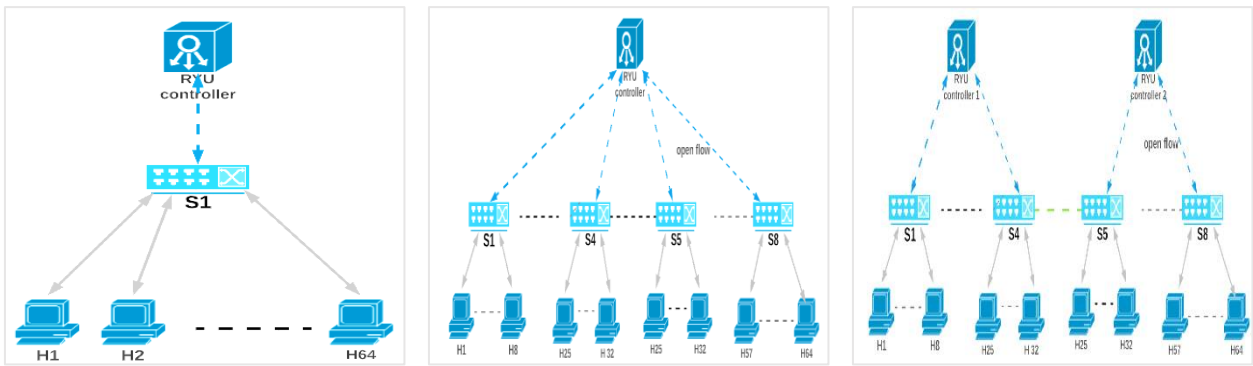



**Figure 1.** Proposed system

Different steps used to construct the system as follow.

### 4.1 Network scenarios construction

Three different network topologies constructed using python script to evaluate the effect of changing topology type on the detection and mitigation techniques efficiency. Single topology is the first network was implemented with 64 hosts, one switch and one controller. Linear with one controller topology was the second network implemented with also 64 hosts but with eight switches and one controller. Finally, a multi-controller Linear topology was implemented, which is the same as the linear but with two-controllers Figure 2 illustrate the topologies architecture.




(a) Single topology (b) Linear topology (c) Multi-Controller topology

**Figure 2.** Simulation network scenarios

### 4.2 Dataset generation

According to the scenarios showed in Figure 2 three datasets have been generated using the simulation instead of using the publicly available datasets that are unrealistic and lack many extracted data features. The datasets generated by collecting the features of network flows during normal and attack traffic using Mininet emulator. Table 3 listed the extracted features of the flow.

A python script responsible for creating the topology will randomly generate the normal ICMP traffic between hosts in the topology using the "Ping" traffic generation command tool. Another python script will generate the ICMP flood traffic, which is a DDoS traffic, between hosts in the topology randomly using the "Hping3" traffic generation command tool. These are the standard command tools for traffic generation in a simulation test. The generation launch 6 hours of normal traffic and 8 minutes of attack traffic for each type of topology which generate data of a (43.7 M Byte/32619 flow) for single topology, (344.4 M Byte/2176918 flow) for linear topology and (259.4 M Byte/1791308 flow) for linear with multi-controller topology.

### 4.3 Model training

After generating the datasets, the machine learning classifier model is trained on each one. At the training, the dataset split into 75% training and 25% testing sets. The trained model will be called in the testing phase by the controller during real-time traffic to predict whether the flow is normal or attack. The confusion matrix and training time are used to evaluate the trained model. Eq. (1) calculate the accuracy of the model depending on confusion matrix.

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN} \times 100\% \qquad (1)$$

where, (TP) is the true positive, (FP) is the false positive, (FN) is the false negative (FN), and (TN) is the true negative which all construct the confusion matrix. Table 4 illustrated the confusion matrix parameters and training time results. FP and FN indicate false alarms. Figure 3 shows the obtained accuracy for each classifier.

**Table 3.** Dataset features

| Field name | Description |
|---|---|
| Timestamp | Arrival time of packet |
| datapath_id | Switch ID |
| Flow_id | Flow identifier |
| ip_src | Source IP address |
| tp_src | Source port |
| ip_dst | Destination IP address |
| tp_dst | Destination port |
| ip_proto | protocol |
| icmp_code | ICMP code |
| icmp_type | ICMP type |
| flow_duration_sec | The duration of flow in seconds |
| flow_duration_nsec | Th duration of flow in nano seconds |
| Idle_timeout | Flow entry timeout after inactivity |
| hard_timeout | Flow expiration time |
| flag | Modify the way flow inputs are handled |
| packet_count | Number of packets |
| byte_count | Number of bytes |
| packet_count_per_second | Number of packets per second |
| packet_count_per_nsecond | Number of packets per nano second |
| byte_count_per_second | Number of bytes per second |
| byte_count_per_nsecond | Number of bytes per nano second |
| label | 0 for normal, 1 for attack |

**Table 4.** Machine learning training results.

| Algorithm | Network Topology | TP | FP | FN | TN | Training Time (sec.) |
|---|---|---|---|---|---|---|
| RF | Single | 25796 | 0 | 0 | 55859 | 27 |
| | Linear | 220747 | 0 | 0 | 323483 | 171 |
| | Multi-controller | 32501 | 0 | 0 | 415326 | 125 |
| KNN | Single | 25796 | 0 | 2 | 55859 | 53 |
| | Linear | 220747 | 0 | 3 | 323483 | 372 |
| | Multi-controller | 32501 | 0 | 0 | 415326 | 275 |
| NB | Single | 25796 | 0 | 15926 | 39933 | 4.9 |
| | Linear | 220747 | 0 | 143764 | 179719 | 16.6 |
| | Multi-controller | 32501 | 0 | 124918 | 290408 | 13.8 |
| LR | Single | 0 | 25796 | 0 | 55859 | 10.6 |
| | Linear | 0 | 220747 | 0 | 323483 | 29.8 |
| | Multi-controller | 0 | 32501 | 0 | 415326 | 27 |

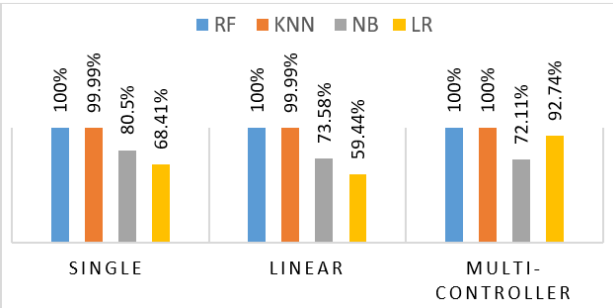



**Figure 3.** Machine learning model accuracy

### 4.4 Model testing

The training results showed that NB and LR have less accuracy and they are both not good for the system. So, farther test step will be implemented only using RF and KNN since they have a good accuracy.

### Traffic generation

The normal traffic is generated using the ping command tool, it sends a request over the network to a specific host. A successful ping results in a response from the host that was pinged, back to the originating host. The attack traffic is generated using Hping3 which is a network tool able to send custom ICMP/UDP/TCP packets. This tool has many options to allow to control the size, quantity, and fragmentation of packets in order to overload the target. Table 5 illustrate normal and attack traffic specification in this system.

**Table 5.** Normal and attack traffic specification

| Traffic parameters | Normal | Attack |
|---|---|---|
| Packet type | ICMP | ICMP Flood |
| Source IP | 10.0.0.1 or any host | Spoofed Random IP |
| Destination IP | 10.0.0.2 or any host | 10.0.0.2 |
| Traffic rate (packets/1 sec.) | 2 | More than 1000 |

### Detection and mitigation process

After training ends, the system will be ready to detect the traffic generated by any host in the topology and classify it. RYU controller python code will call the trained model and start testing it on the real-time traffic. It will get the flow statistics from all the switches every 3 seconds and pass them as input to the classifier model. The classifier predicts whether

it is an attack or normal traffic and then displays the result on the controller terminal. If the traffic is malicious (DDoS), the controller also displays the victim, and then the mitigation process starts.

When generating normal traffic between any two hosts it has been observed that RYU terminal correctly detects a legitimate traffic happening. Figure 4 shows the correct detection in the multi-controller topology. The two other topologies also display a correct detection, but we eliminate their results for abbreviation.

While normal traffic still running, we test the detection of an attack traffic. A DDoS attack traffic is generated with random spoofed IPs using hping3 tool between two hosts for each topology, in all topologies host 64 will attack host 2. After few seconds, RYU will detect attack traffic and print the victim host. Figure 5 shows the result of DDoS detection only for multi-controller topology for abbreviation also.

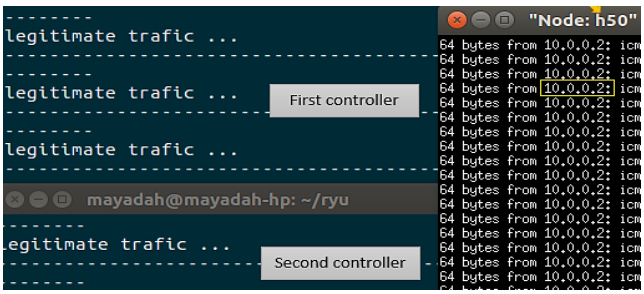



**Figure 4.** Detecting normal traffic in Ryu terminal in multi-controller topology

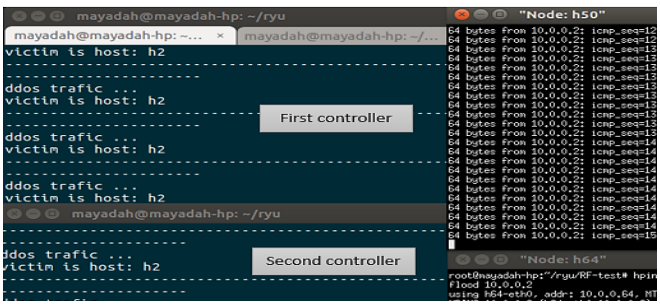


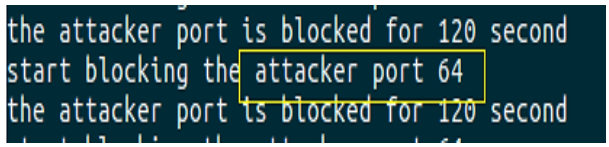

**Figure 5.** Detecting attack traffic in Ryu terminal in multi-controller topology

Next, mitigation process starts to stop the attack traffic from exhaustion the network resources. It will find the source port of the attacker and then block this port for 120 sec. The drop packet rule will be updated by the controller in the switch flow-table. The detection and mitigation process steps are explained in algorithm 1.

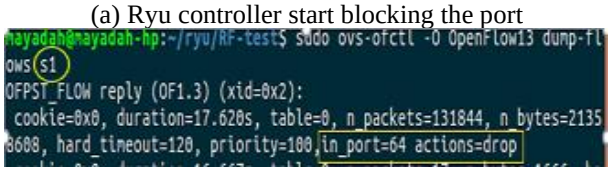
***Algorithm 1: Detection and Mitigation Process***

**Select the classifier (RF or KNN)**
**hard_time = t #block port timeout**
**mitigation = 0 (before detection)**
**1: Create a trained model based on generated dataset**
**2: Capture the packets every 3 second and process it to get necessary fields (*collect_flow_stat*)**
**3: Classify the packet using the model**
**4: if classifier classifies the flow as anomaly, then**
**5: Display the traffic as attack and specify the victim**
**6: mitigation =1 #activate mitigation process**
**7: Start: DDoS attack detected from host x**
**8: if (mitigation =1):**
**9: "Check the IP address of host x"**
**10: if (Packet_in of source IP = random):**
**11: implement block_port function**
***12: Block_port (switch_id, src_port)***
**13: "Port will be blocked for t second"**
**14: else:**
**15: mitigation =0**
**16: else**
**17: Displays the traffic as legitimate**

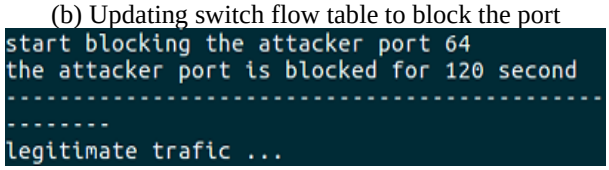
After mitigation done correctly the network will return to its normal traffic. The real time results during mitigation at single and linear topologies are shown in Figures 6 and 7. We should mention that in linear topology and multi-controller topology, host 64 which is the attacker host is located at switch 10 port 11.




(a) Ryu controller start blocking the port

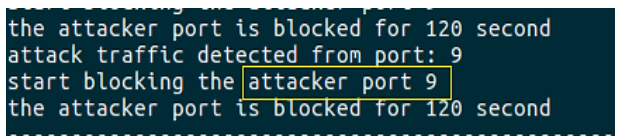



(b) Updating switch flow table to block the port

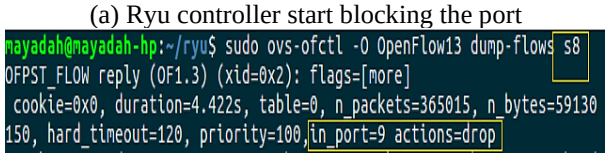



(c) Return to normal traffic after mitigating the attack

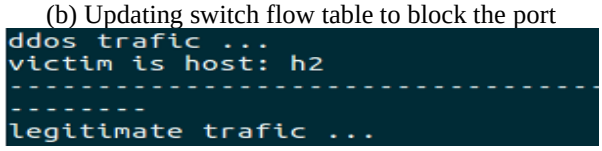
**Figure 6.** Mitigation results in single topology




(a) Ryu controller start blocking the port




(b) Updating switch flow table to block the port




(c) Return to normal traffic after mitigating the attack

**Figure 7.** Mitigation results in linear topology

In the linear with multi-controller topology as it has two domains as shown in Figure 8, the port that receives traffic from the other domain will be monitored and if an attack is detected this port it will be blocked for 30 seconds to give time for the other controller to start to block the attacker port at his

domain, this small-time block for the traffic between the 2 domains will not block the normal traffic between domains for a lot of time and ensure that the DDoS traffic will not harm the target domain if the attacker domain controller delayed in mitigating the attack. The real time results during mitigation in this scenario shown in Figure 9.

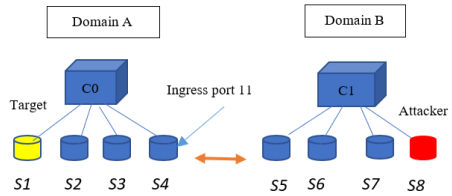
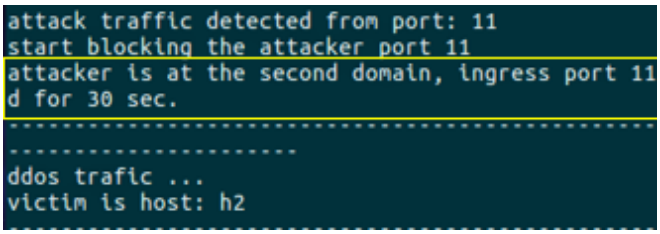
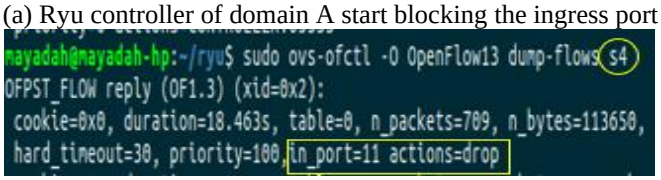


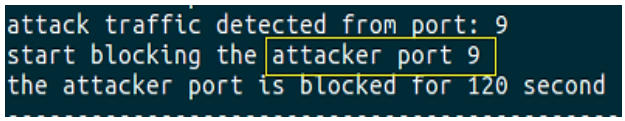

**Figure 8.** Linear with 2-controllers topology as a two-domain structure



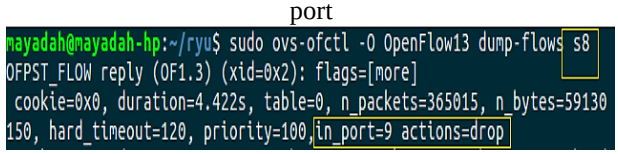

(a) Ryu controller of domain A start blocking the ingress port



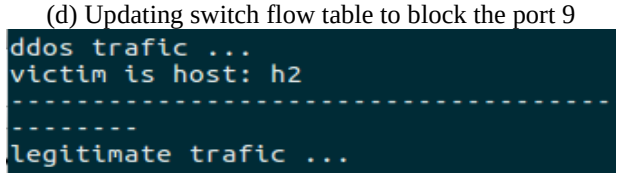

(b) Updating switch flow table to block the port 11




(c) Ryu controller of domain B start blocking the attacker port




(d) Updating switch flow table to block the port 9




(e) Return to normal traffic in the domains after mitigating the attack

**Figure 9.** Mitigation results in multi-controller topology

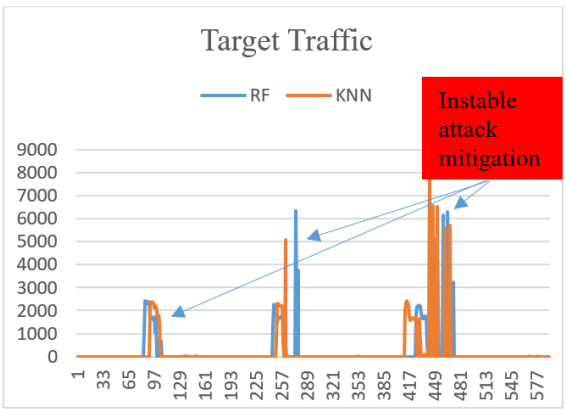
## 5. RESULTS AND DISCUSSION

The proposed algorithms tested for their ability to detect and mitigate DDoS attack in the three simulation networks and to prove its accurate classification for the incoming traffic to the controller. The monitoring metrics during the evaluation are the packets rate of the victim and controller, as well as the attack detection and mitigation time. Table 6 shows the results collected during the run test of all network topology using RF and KNN classifiers and during three DDoS attack strikes.

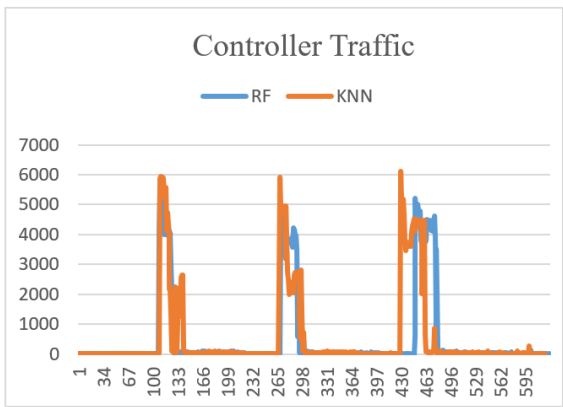
**Table 6.** Simulation results of RF and KNN models

| Test metrics | RF | | | KNN | | |
|---|---|---|---|---|---|---|
| | single | linear | 2-controller | single | linear | 2-controller |
| **Normal Tx packet** | 580 | 580 | 480 | 580 | 480 | 480 |
| **Normal Rx packet** | 416 | 577 | 418 | 351 | 480 | 450 |
| **Packet loss** | 28% | 0% | 12% | 39% | 0% | 6% |
| **Attack Tx packet** | 34487300 | 26242144 | 24647452 | 33910143 | 24104797 | 25092838 |
| **Detection time** | 12 | 5 | 3 | 10 | 5 | 3 |
| ***Max. Mitigation time** | 22/32/48 | 8/11/10 | 5/7/4 | 14/12/58 | 9/10/8 | 6/4/6 |
| ***Target max. Packet rate during attack** | 2425/6344/6313 | 314/183/184 | 220/276/244 | 2394/5092/8605 | 178/185/212 | 224/227/295 |

*The values (x/x/x) are for the three attack strikes.

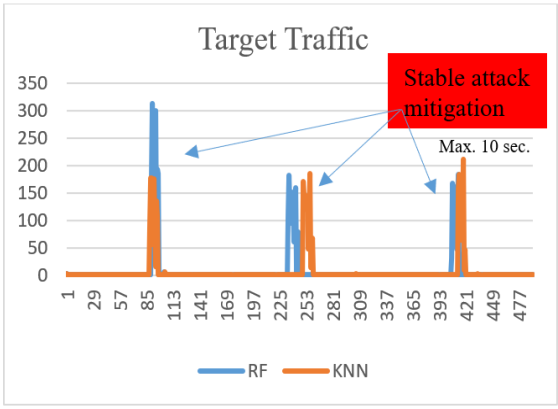



(a) Target packets rate

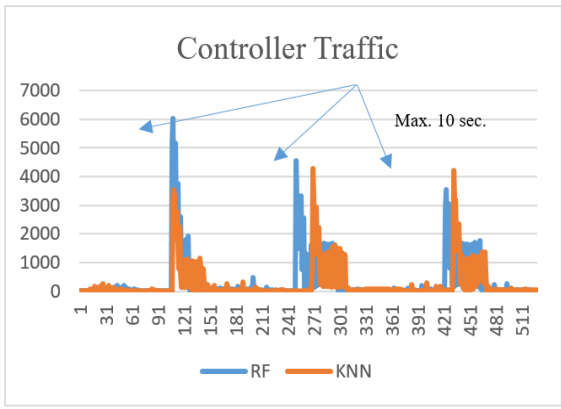



(b) Ryu controller packets rate

**Figure 10.** System results in single topology during detection and mitigation of DDoS attack using RF and KNN classifiers

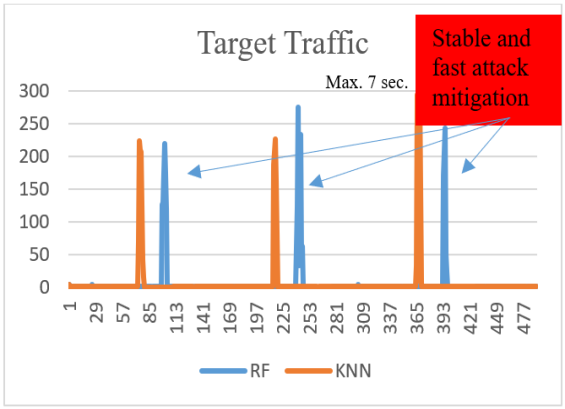



(a) Target packets rate

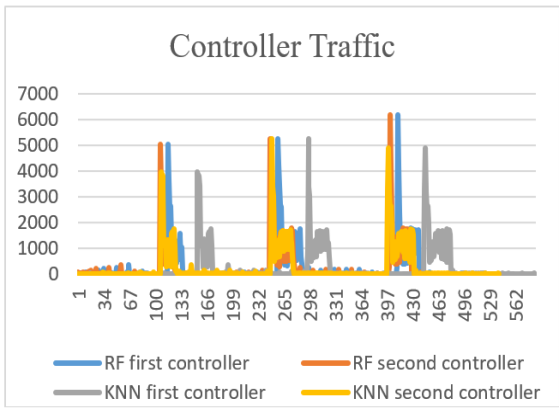



(b) Ryu controller packets rate

**Figure 11.** System results in linear topology during detection and mitigation of DDoS attack using RF and KNN classifiers




(a) Target packets rate




(b) Ryu controller packets rate

**Figure 12.** System results in multi-controller linear topology during detection and mitigation of DDoS attack using RF

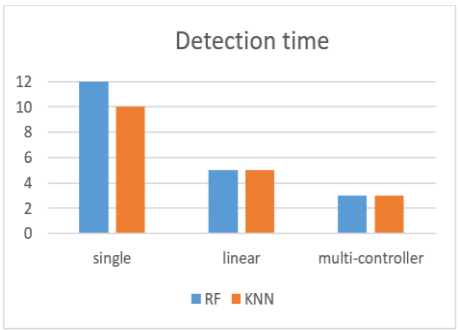



**Figure 13.** Detection time for all tests

We can observe that:

- There is a high packet loss at the single topology (28 % at RF, 39% at KNN).
- The mitigation time is high and not stable in single topology as shown in Figure 10. However, in linear and multi-controller topologies it is lower and optimum as shows in Figures 11 and 12.
- No packet loss in linear topology and a little loss in multi-controllers.
- The target exposed to a high packet rate during attack strikes in single topology reach up to 6300 packets during RF test and up to 8600 packets during KNN test, which is very high traffic that saturate target resources and caused a packet loss for legitimate traffic.
- The packet loss in the multi-controller topology is due to the 30-second port blocking between the two domains during the attack and not because of the classifier efficiency.
- Detection time is low and the single topology has the maximum detection time (12 sec.) as shown in Figure 13.

From observing the target and controller traffic we can find that the packets per second are significantly increased during the launch of DDoS attack, and it is mostly affecting the controller traffic because since the source addresses of the attack are spoofed random IPs, the switch will forward all them. When mitigation started, the target will return to the normal traffic faster than the controller does.

## 6. CONCLUSIONS

DDoS attack detection is one of the main challenges of SDN security today. One of the great enhancements to network management that emerged in SDN is the ability to build an application to mitigate security issues or to propose new software solutions to the network environment, especially with modern intelligent technologies. Machine learning algorithms are one of these solutions that can significantly improve the quality of SDN security. RF, KNN, NB, and LR are proposed in this work as supervised machine learning algorithms to detect a DDoS attack in three network architectures which are single topology, linear topology, and multi-controller topology. The models have been trained on the datasets generated in a simulated SDN environment using Mininet emulator and RYU controller. The simulation results show that NB and LR have low accuracy rates and produce many wrong predictions. On the other hand, RF and KNN produce high accuracy rates and

can be effectively used as prediction models for this work.

From monitoring network traffic during the attack, it can be observed that this attack is mainly attempting to exhaust the controller and bring it down by flooding the switch flow table with spoofed IP addresses requests. It also caused some loss in normal packet flow by busying the controller with these faked requests. This effect is mostly appeared in the single topology unlike linear and multi-controller topologies, which means that increasing the switches number lowers the load and helps to eliminate the attack effect fast. Also, increasing network switches minimizes the detection and mitigation time. Moreover, increasing the number of controllers enhances the detection and mitigation process by minimizing the error rate, detection and mitigation time.

Finally, the proposed mitigation technique is accurately implemented to stop the attack before it harms the controller by blocking the attacker port for 120 second.

For future work we suggest implementing the system in real network instead of virtual and adopting more SDN controllers in the multi-controller topology. Also, other types of machine learning algorithms such as reinforcement which doesn't need a dataset can be tested.

## REFERENCES


[1] Gupta, S., Grover, D. (2021). A comprehensive review on detection of DDoS attacks using ML in SDN environment. In 2021 International Conference on Artificial Intelligence and Smart Systems (ICAIS), pp. 1158-1163. https://doi.org/10.1109/ICAIS50930.2021.9395987

[2] Deepa, V., Sudar, K.M., Deepalakshmi, P. (2018). Detection of DDoS attack on SDN control plane using hybrid machine learning techniques. In 2018 International Conference on Smart Systems and Inventive Technology (ICSSIT), pp. 299-303. https://doi.org/10.1109/ICSSIT.2018.8748836

[3] Mohsin, M.A., Hamad, A.H. (2022). Implementation of entropy-based DDoS attack detection method in different SDN topologies. American Academic Scientific Research Journal for Engineering, Technology, and Sciences, 86(1): 63-76.

[4] Bawany, N.Z., Shamsi, J.A., Salah, K. (2017). DDoS attack detection and mitigation using SDN: Methods, practices, and solutions. Arabian Journal for Science and Engineering, 42(2): 425-441. https://doi.org/10.1007/s13369-017-2414-5

[5] Tuan, T.A., Long, H.V., Son, L.H., Kumar, R., Priyadarshini, I., Son, N.T.K. (2020). Performance evaluation of Botnet DDoS attack detection using machine learning. Evolutionary Intelligence, 13(2): 283-294. https://doi.org/10.1007/s12065-019-00310-w

[6] Hamad, A.H. (2021). Smart campus monitoring based video surveillance using haar like features and K-Nearest neighbour. International Journal of Computing and Digital Systems, 10(1): 863-870. http://dx.doi.org/10.12785/ijcds/100179

[7] Beslin Pajila, P.J., Golden Julie, E. (2019). Detection of DDoS attack using SDN in IoT: A survey. In Intelligent Communication Technologies and Virtual Mobile Networks, pp. 438-452. https://doi.org/10.1007/978-3-030-28364-3_44

[8] Ye, J., Cheng, X., Zhu, J., Feng, L., Song, L. (2018). A DDoS attack detection method based on SVM in software defined network. Security and Communication Networks. https://doi.org/10.1155/2018/9804061

[9] Sahoo, K.S., Iqbal, A., Maiti, P., Sahoo, B. (2018). A machine learning approach for predicting DDoS traffic in software defined networks. In 2018 International Conference on Information Technology (ICIT), pp. 199-203. https://doi.org/10.1109/icit.2018.00049

[10] Prakash, A., Priyadarshini, R. (2018). An intelligent software defined network controller for preventing distributed denial of service attack. In 2018 Second International Conference on Inventive Communication and Computational Technologies (ICICCT), pp. 585-589. https://doi.org/10.1109/icicct.2018.8473340

[11] Le, D.T., Dao, M.H., Nguyen, Q.L.T. (2020). Comparison of machine learning algorithms for DDoS attack detection in SDN. Информационно-управляющие системы, 3(106): 59-70. https://doi.org/10.31799/1684-8853-2020-3-59-70

[12] Karan, B.V., Narayan, D.G., Hiremath, P.S. (2018). Detection of DDoS attacks in software defined networks. In 2018 3rd International Conference on Computational Systems and Information Technology for Sustainable Solutions (CSITSS), pp. 265-270. https://doi.org/10.1109/csitss.2018.8768551

[13] Rahman, O., Quraishi, M.A.G., Lung, C.H. (2019). DDoS attacks detection and mitigation in SDN using machine learning. In 2019 IEEE World Congress on Services (SERVICES), 2642: 184-189. https://doi.org/10.1109/services.2019.00051

[14] Polat, H., Polat, O., Cetin, A. (2020). Detecting DDoS attacks in software-defined networks through feature selection methods and machine learning models. Sustainability, 12(3): 1035. https://doi.org/10.3390/su12031035

[15] Santos, R., Souza, D., Santo, W., Ribeiro, A., Moreno, E. (2020). Machine learning algorithms to detect DDoS attacks in SDN. Concurrency and Computation: Practice and Experience, 32(16): e5402. https://doi.org/10.1002/cpe.5402

[16] Aljuhani, A. (2021). Machine learning approaches for combating distributed denial of service attacks in modern networking environments. IEEE Access, 9: 42236-42264. https://doi.org/10.1109/access.2021.3062909

[17] Aljuhani, A. (2021). Machine learning approaches for combating distributed denial of service attacks in modern networking environments. IEEE Access, 9: 42236-42264. https://doi.org/10.1109/access.2021.3062909

[18] Zhao, Y., Li, Y., Zhang, X., Geng, G., Zhang, W., Sun, Y. (2019). A survey of networking applications applying the software defined networking concept based on machine learning. IEEE Access, 7: 95397-95417. https://doi.org/10.1109/access.2019.2928564

[19] Sangodoyin, A.O., Akinsolu, M.O., Pillai, P., Grout, V. (2021). Detection and classification of DDoS flooding attacks on software-defined networks: A case study for the application of machine learning. IEEE Access, 9: 122495-122508. https://doi.org/10.1109/ACCESS.2021.3109490

[20] Amin, R., Rojas, E., Aqdus, A., Ramzan, S., Casillas-Perez, D., Arco, J.M. (2021). A survey on machine learning techniques for routing optimization in SDN. IEEE Access. https://doi.org/10.1109/access.2021.3099092