Comparative Analysis of DDoS Detection Techniques Based on Machine Learning in OpenFlow Network

Fauzi Dwi Setiawan Sumadi
Informatics Department
Universitas Muhammadiyah Malang
Malang, Indonesia
fauzisumadi@umm.ac.id

Christian Sri Kusuma Aditya
Informatics Department
Universitas Muhammadiyah Malang
Malang, Indonesia
christianskaditya@umm.ac.id

2020 3rd International Seminar on Research of Information Technology and Intelligent Systems (ISRITI) | 978-1-7281-8406-7/20/$31.00 ©2020 IEEE | DOI: 10.1109/ISRITI51436.2020.9315510

***Abstract*— Software Defined Network (SDN) allows the separation of a control layer and data forwarding at two different layers. However, centralized control systems in SDN is vulnerable to attacks namely distributed denial of service (DDoS). Therefore, it is necessary for developing a solution based on reactive applications that can identify, detect, as well as mitigate the attacks comprehensively. In this paper, an application has been built based on machine learning methods including, Support Vector Machine (SVM) using Linear and Radial Basis Function kernel, K-Nearest Neighbor (KNN), Decision Tree (DTC), Random Forest (RFC), Multi-Layer Perceptron (MLP), and Gaussian Naïve Bayes (GNB). The paper also proposed a new scheme of DDOS dataset in SDN by gathering considerably static data form using the port statistic. SVM became the most efficient method for identifying DDoS attack successfully proved by the accuracy, precision, and recall approximately 100 % which could be considered as the primary algorithm for detecting DDoS. In term of the promptness, KNN had the slowest rate for the whole process, while the fastest was depicted by GNB.**

Keywords—SDN, Machine Learning, DDoS, Detection, Reactive

I. INTRODUCTION

The development of computer network technology is pushing some of the core sectors in the network to make radical changes in order to meet the trends of computer networks that are needed today including agile, flexible, and versatile. One common problem that must be analysed carefully is network management. With the continued increase in computer network devices (Internet of Things), efforts in managing network traffic will increase linearly. In a traditional computer network, a network administrator must manually configure each layer 2 and layer 3 device if there are problems with the device. This must be done because there are different configurations set by each network device vendor with specific procedures [1]. For example, to configure a router with a Cisco vendor it will be different from a Juniper router. Complex computer network trends can trigger misconfiguration or human errors.

In order to resolve the specified problems, computer network paradigm was developed called SDN. SDN separates vertical abstractions in traditional network devices consisting of 2 core layers, the controller layer and the forwarding layer [2]. The controller acts as the centre of the network settings that are connected directly to the device. Various network management functions can be flexibly managed by installing applications, for example, the routing process, the mapping of network topology, the intrusion detection system, the discovery of the Internet Protocol (IP), the discovery of the Address Resolution Protocol (ARP), and other special functions built with program lines in the application layer on the controller. The application that has been installed in the controller will process each incoming packet and respond according to the algorithm that has been defined. There is a southbound API protocol that functions to standardised the communication so that the layer forwarding device can perform the forwarding function compiled by the controller.

The implementation of a centralized control system provides a security loophole that can be exploited by irresponsible parties to be able to carry out attacks aimed at broadly impacting the topology that is directly connected to a controller. One example of an attack that can be carried out both locally and globally is DDoS. This type of attack can saturate links that are connected directly to the controller. The controller will indirectly process the incoming new packet to the topology. This is a major weakness in implementing a centralized control system. With the number of new packages that exceed the normal limit will force the controller to process the package by utilizing all available resources so that it can bring the controller in an unstable condition.

Several methods have been implemented before by integrating machine learning. However, its implementation is only based on the process of detection and identification of attacks and used the available dataset that did not maintain the feature specifically for SDN environment. In the process of detection, several previous studies applied a proactive scheme by utilizing flow integration tools for packet headers without involving controllers. But in this method, there are still scalability issues where the system can only be implemented on forwarding devices that have adequate resources. [3] using the C4.5 algorithm, Bayesian Network, Decision Table, and Naïve Bayes in the DDoS detection process. The system is developed reactively by providing certain IP subnet block actions, with a dataset that has 4 features, including, the IP of the attacker, the destination host, the attack number, and the timestamp of the attack. This research has not discussed in detail the port and protocol-based attacks. In research [4], the authors used the SVM, Naïve Bayes, and Neural Network methods to identify DDoS attacks with a selection feature based only on the number of hosts making requests every second without using IP and

2020 3rd International Seminar on Research of Information Technology and Intelligent Systems (ISRITI) | 978-1-7281-8406-7/20/$31.00 ©2020 IEEE | DOI: 10.1109/ISRITI51436.2020.9315510

protocol. Whereas in [5], the authors compared several classification methods with datasets obtained from DARPA without mitigating and were not accompanied by a description of the mechanism of comparison of the use of methods to the use of existing resources in SDN. Moreover, in [6] the authors created an ML server which could extract the dataset from the controller based on the NL-KDD dataset which gained 98% in accuracy and defend proactively. [7] proposed machine learning and deep learning detection without implementing the mitigation method for SDN. The authors used NSL-KDD dataset which gained accuracy of 88% for Deep Neural Network method. [8] used the combination of 3 distinct available online datasets where the feature extracted using sFlow gained 96.5% for the detection rate utilising Random Forest Algorithm.

In different mechanism, several papers conducted their experiment by using their own dataset extraction process which was considered compatible with the SDN environment such as flow rule statistic. Paper [9] used sFlow dockers to extract the flow data from the SDN switches and utilised the feature extraction process gaining 98.3% accuracy for the KNN classifier. [10] proposed an almost similar pattern of a dataset, using 4 main features (flow length, flow duration, flow size, and flow ratio). The researchers developed an improved KNN method and achieved around 99.4% for the recall variable. Moreover, paper [11] also implemented flow stat request for generating 6 features as the main resource to classify using SVM resulting 95.24% for the accuracy of the detection rate. In similar pattern, paper [12] used the flow statistic mechanism for developing dataset features in nmeta2 architecture which then classified by using SVM, KNN, and Random Forest. The highest accuracy depicted by KNN at around 93.5%.

Based on the previous research that has been done before, this research was focused to develop and analyse applications to detect, identify, and mitigate DDoS attacks on SDN networks from various machine learning classification methods including, RBF SVM, Linear SVM, KNN, DTC, RFC, MLP, and GNB reactively. This research also proposed new feature scheme for the dataset which could create an almost static form of dataset increasing to the accuracy variable. Later, each method will be analysed based on the accuracy of the detection and mitigation results as well as the accumulative data of resource usage from the controller when running the application. In addition, the application will implement the DDoS mitigation scheme by utilizing the priority rule feature provided by the OpenFlow standard. So packets that are classified as malicious packets will automatically be blocked. The details of the application scheme method and the research method are explained further in the research method section.

II. Research Method

The research was emulated using Mininet [13] as the emulator software. The topology consisted of three main SDN OvS switches [14], one RYU [15] controller as the centre of the networking management process, and six Hosts. Fig. 1 illustrates the overview of the specified topology. The main concern of the attacker was directed to perform a DDoS attack directly to the controller by sending randomly generated data test packets. The H1 acted as the attacker by sending a large number of packets to the H4.

The controller by default implemented learning switch mechanism for mapping the network environment. The flow of the proposed reactive machine learning application is described in Fig. 2 and Fig. 3. Upon receiving new packets, the switch will directly filter the packet's header with the available flow selector installed on the switch. If there is no packet's selector that can filter the incoming packet the switch will encapsulate the incoming packet in OPFT_PACKET_IN message to the controller [16].

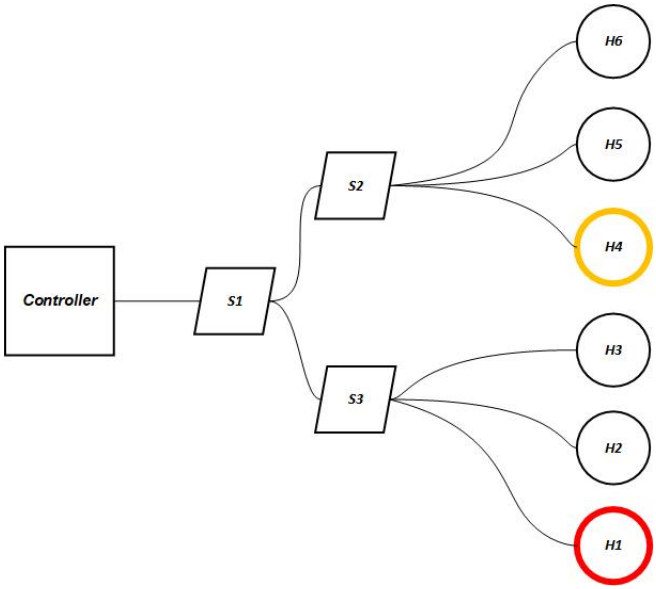

Fig. 1. Emulation's topology

Since the construction of the packet consisted of either random source port or IP source, the SDN switch would automatically forward the incoming DDoS packet to the controller for further assessment. Otherwise, the SDN switch will perform the traffic treatment specified by the controller, such as forwarding, dropping, crafting, etc.

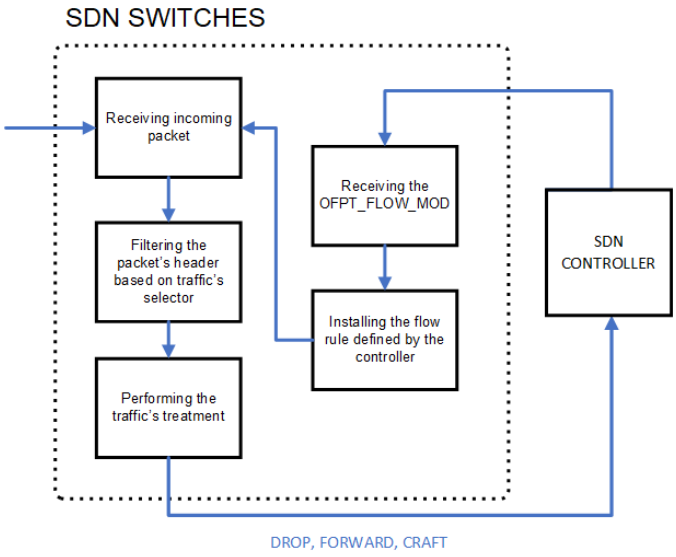

Fig. 2. SDN switch's block diagram

After the controller receiving the OFPT_PACKET_IN message, the machine learning app will parse the packet's header corresponding to the list of features used on the

classification model (the details are explained in section III). Subsequently, the application processes the header's information and converts it into float format using a standard scaler. Furthermore, the application applies the classification model to predict the result of the current packet's header information. If the result states that the packet is considered as the normal packet, the application will transfer the packet processing mechanism to learning switch application. On the other hand, if the packet is identified as DDoS packet, the controller creates OFPT_FLOW_MOD message consists of flow rule construction command for blocking the DDoS attack based on the identified protocol's type then sends it to the corresponding SDN switch (directly connected to the attacker). Therefore, the attack will be blocked or dropped by the SDN switch using an idle time period (60s).

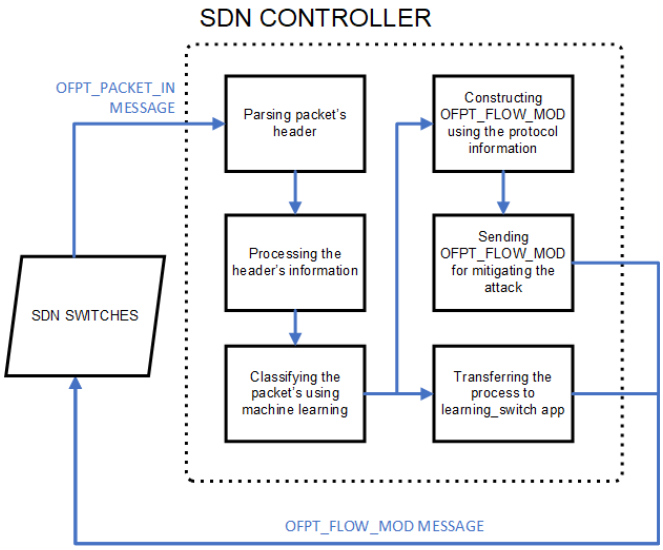

Fig. 3. SDN controller's block diagram

The classification algorithms that were used during the experiment consisted of RBF SVM, Linear SVM, KNN, DTC, RFC, MLP, QDA, and GNB. Each of the algorithms has its own hyperparameter which is illustrated in Table I. In order to analyse the effectiveness of the specified algorithms, several variables were used to measure which was the most accurate algorithm for classifying DDoS attack. The variables included accuracy, precision, recall, and F1 score.

TABLE I. HYPERPARAMETERS USED IN CLASSIFICATION

Algorithm	Hyperparameters
RBF SVM	Kernel: RBF Kernel Coefficient: 0.01 Lambda: 10
Linear SVM	Kernel: Linear Function Lambda: 0.025
KNN	Number of Neighbours: 2 Weight Function: Uniform Computational Algorithm: Brute-Force
DTC	Split Function: Gini Information Gain: Entropy Maximum Tree's Depth: 5
RFC	Number of Trees: 10 Split Function: Gini Information Gain: Entropy Maximum Tree's Depth: 5
MLP	Hidden Layer: 1 Activation Function: Relu Solver: Adam Maximum Iteration: 200
ADC	Maximum Number of Estimator: 50 Boosting Algorithm: SAMME.R
GNB	the largest variance: 1e-09 Prior Probabilities: Adjusted Based on Data

III. RESEACH DATASET

Instead of using flow statistic data which was introduced in [9-12], the experiment was carried out using different dataset structure that could be categorized as a static form of data. The dataset can be found in [17]. The data extraction process used the IPv4, ICMP, TCP, and UDP header information added by some information regarding to the port's statistic extracted from the reply of OFPMP_PORT_STATS requests. The list of features is described in Table II.

TABLE II. FEATURE LIST

Feature's Name	Feature's Origin
datapath_id	OFPT_PACKET_IN
version	IPv4's Header
header_length	IPv4's Header
tos	IPv4's Header
total_length	IPv4's Header
flags	IPv4's Header
offset	IPv4's Header
ttl	IPv4's Header
proto	IPv4's Header
csum	IPv4's Header
src_ip	IPv4's Header
dst_ip	IPv4's Header
src_port	UDP's/TCP's Header
dst_port	UDP's/TCP's Header
tcp_flag	TCP's Header
type_icmp	ICMP's Header
code_icmp	ICMP's Header
csum_icmp	ICMP's Header
port_no	OFPPortStatsReply
rx_bytes_ave	OFPPortStatsReply (rx_bytes / rx_packets)
rx_error_ave	OFPPortStatsReply (rx_errors / rx_packets)
rx_dropped_ave	OFPPortStatsReply (rx_dropped / rx_packets)
tx_bytes_ave	OFPPortStatsReply (tx_bytes / tx_packets)
tx_error_ave	OFPPortStatsReply (tx_errors / tx_packets)
tx_dropped_ave	OFPPortStatsReply (tx_dropped / tx_packets)

Data extraction process (Fig. 4) started from the attacker by sending two *.pcap files that contained the train data and test data. The flooding process was performed individually for both files. The authors configured the SDN switches for

passing the incoming packet directly to the controller. Simple switch application in RYU had been extended as well in order to retrieve the packet's header information –

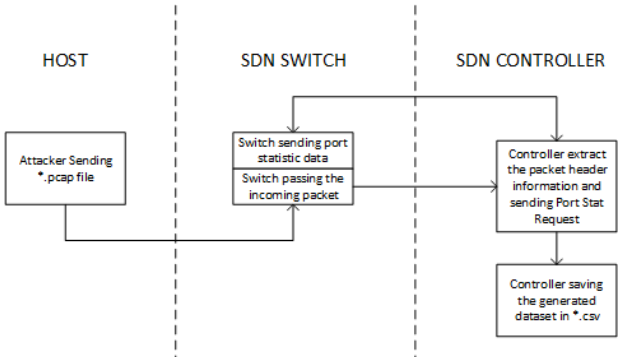

Fig. 4. Dataset extraction process

concerning the IPv4, TCP, UDP, and ICMP as well as sending OFPPortStatsRequest to the corresponding switch for acquiring the port statistic. The average number of bytes of the incoming packet was calculated by differentiating the number of bytes received and the number of packets received (rx_bytes_ave). The attack was conducted by sending 3 distinct types of attacks including the ICMP, TCP, and UDP flood attack. The available class consisted of 6 class which included the DDOS_ICMP, DDOS_TCP, DDOS_UDP, NORMAL_ICMP, NORMAL_TCP, and NORMAL_UDP. In term of the data proportion, the number of train data is 420,0000 data while the test data is divided into two distinct number including the 18,000 and 36,000 data tests. Both data types (train and test) were generated on different occasion and consisted of different structures.

IV. Results and Discussion

The experiment's results showed several variables that could be used to measure the capability of each proposed algorithm for detecting and identifying the DDoS attack. Table III showed the training time for each algorithm to create the classifier model based on two distinct data proportion (420,000:18,000 and 420,000:36,000). The longest time was depicted by KNN followed by MLP and SVM respectively while the other algorithms could finish the model before or equal to 1 s. This behaviour might occur since the SVM used Linear and RBF kernel which were considered as the complex kernel. KNN also took a considerably long time to develop a model because of the calculation of the distance between data point.

TABLE III. Training Time

Classifier Type	Training Time (s)
SVM (RBF)	6.514
SVM (LIN)	3.085
KNN	85.781
DTC	1
RFC	0.7
MLP	18.971
GNB	0.1

Moreover, the experiment was conducted into two different scenarios which included the prediction scheme without involving the SDN controller and generating the prediction using the SDN controller. In term of the first scenario portrayed by Table VI and Table V, the accuracy for DTC were decreasing because the uniformity of data test dropping along with the growth of the number of data test. The positive trend illustrated by SVM, KNN, RFC, MLP, and GNB. The results showed the highest accuracy depicted by both Linear SVM and GNB for all data test size. However, this trend was not described in the second scenario due to data redundancy.

TABLE IV. Experiment's Result For 18000 Dataset Without SDN

Classifier Type	Accuracy (18000) %	Precision (18000) %	Recall (18000) %	F1 (18000) %
SVM (RBF)	99.3	99.3	99.3	99.3
SVM (LIN)	99.9	99.9	99.9	99.9
KNN	99	99	99	99
DTC	100	100	100	100
RFC	69.5	80	69.5	62.7
MLP	35.6	33.2	35.6	22.3
GNB	99.7	99.7	99.7	99.7

TABLE V. Experiment's Result For 36000 Dataset Without SDN

Classifier Type	Accuracy (36000) %	Precision (36000) %	Recall (36000) %	F1 (36000) %
SVM (RBF)	100	100	100	100
SVM (LIN)	100	100	100	100
KNN	99.4	99.4	99.4	99.4
DTC	83.3	75	83.3	77.7
RFC	97	97.5	97	97
MLP	50.4	50.4	50.4	50.4
GNB	100	100	100	100

Surprisingly, the pattern did not occur for some classification algorithm on the second scenario since the prediction results appeared to be redundant. This circumstance might occur because there was a lot of feature processing mechanism that should be maintained by the SDN controller. Therefore, the percentage of prediction loss was pointed at around 79.8% for 36,000 data tests which meant only 7,272 data successfully classified illustrated in Table VIII. In details, the algorithm that could maintain its accuracy variable was RBF SVM and Linear SVM approximately at 100% indicating that the increase of the data test did not reduce the robustness of the classification process (Table VI and Table VII). The increasing trend in accuracy also produced by KNN. The remaining algorithms including DTC, RFC, MLP, and GNB generated a decreasing percentage for all variables which indicated that the algorithms could not handle the classification efficiently.

TABLE VI. Experiment's Result For 18000 Dataset In SDN

Classifier Type	Accuracy (18000) %	Precision (18000) %	Recall (18000) %	F1 (18000) %
SVM (RBF)	100	100	100	100

Classifier Type	Accuracy (18000) %	Precision (18000) %	Recall (18000) %	F1 (18000) %
SVM (LIN)	100	100	100	100
KNN	87.8	93.1	88.2	86.5
DTC	100	100	100	100
RFC	84.2	86.4	84.6	83.1
MLP	34.5	32.5	36.3	23
GNB	93.9	95	94.1	93.5

TABLE VII. EXPERIMENT'S RESULT FOR 36000 DATASET IN SDN

Classifier Type	Accuracy (36000) %	Precision (36000) %	Recall (36000) %	F1 (36000) %
SVM (RBF)	100	100	100	100
SVM (LIN)	100	100	100	100
KNN	95.6	89.2	92.9	87.5
DTC	89.7	78.4	83.3	80.4
RFC	80.7	88.4	84.4	82.5
MLP	65.3	52.4	66.6	54.2
GNB	59.1	68.6	83.1	70

TABLE VIII. PERCENTAGE OF PREDICTION LOSS IN SDN

Classifier Type	Prediction Loss (18000) in %	Prediction Loss (36000) in %
SVM (RBF)	2.7	79.8
SVM (LIN)	2.7	79.8
KNN	2.7	79.8
DTC	2.7	79.8
RFC	2.7	79.8
MLP	2.7	79.8
GNB	2.7	79.8

In addition, Table IX shows the most significant features (top 10) that influences the classification process. The selection was done by utilizing Random Forest Regressor. The Table also informs the source of the feature which mainly are extracted from IPv4's packet header as well as the data from sending OFPMP_PORT_STATS message directly to the SDN controller.

TABLE IX. TOP 10 FEATURES FOR CLASSIFICATION

Feature Name	Feature's Source
total_length	IPv4
ttl	IPv4
proto	IPv4
csum	IPv4
src_ip	IPv4
src_port	IPv4
dst_port	IPv4
tcp_flag	IPv4
port_no	Port_Stat
rx_bytes_ave	Port_Stat

V. RESULTS AND DISCUSSION

Overall, DDoS still became one of the most significant issues in SDN in regard to centralized control management. The authors proposed new scheme of dataset which utilized the packet's library and the port statistic request for extracting 25 features in total. Machine learning methods was comparatively analyzed and can be deduced that the SVM whether used the Linear or RBF kernel could produce the highest accuracy among several similar researches. The authors will try to analyze the significance of Deep Learning method for detecting the similar attacks in the future.

ACKNOWLEDGMENT

The authors would like to say their profound gratitude for the Universitas Muhammadiyah Malang as the main sponsor for this research.

REFERENCES

[1] Sezer S, Scott-Hayward S, Chouhan PK, Fraser B, Lake D, Finnegan J, et al. Are we ready for SDN? Implementation challenges for software-defined networks. IEEE Communications Magazine. 2013;51(7):36-43.

[2] Software Defined Networking: The Norm For Networks (2012).

[3] Nanda S, Zafari F, DeCusatis C, Wedaa E, Yang B, editors. Predicting network attack patterns in SDN using machine learning approach. 2016 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN); 2016 7-10 Nov. 2016.

[4] Meti N, Narayan DG, Baligar VP, editors. Detection of distributed denial of service attacks using machine learning algorithms in software defined networks. 2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI); 2017 13-16 Sept. 2017.

[5] Kokila RT, Selvi ST, Govindarajan K, editors. DDoS detection and analysis in SDN-based environment using support vector machine classifier. 2014 Sixth International Conference on Advanced Computing (ICoAC); 2014 17-19 Dec. 2014.

[6] Mohammed SS, Hussain R, Senko O, Bimaganbetov B, Lee J, Hussain F, et al., editors. A New Machine Learning-based Collaborative DDoS Mitigation Mechanism in Software-Defined Network. 2018 14th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob); 2018 15-17 Oct. 2018.

[7] Dey KS, Rahman MM. Effects of Machine Learning Approach in Flow-Based Anomaly Detection on Software-Defined Networking. Symmetry. 2019;12(1).

[8] Lima Filho FS de, Silveira FAF, de Medeiros Brito Junior A, Vargas-Solar G, Silveira LF. Smart Detection: An Online Approach for DoS/DDoS Attack Detection Using Machine Learning. Maglaras L, editor. Secur Commun Networks [Internet]. 2019;2019:1574749. Available from: https://doi.org/10.1155/2019/1574749

[9] Polat H, Polat O, Cetin A. Detecting DDoS Attacks in Software-Defined Networks Through Feature Selection Methods and Machine Learning Models. Sustainability. 2020;12(3).

[10] Dong S, Sarem M. DDoS Attack Detection Method Based on Improved KNN With the Degree of DDoS Attack in Software-Defined Networks. IEEE Access. 2020;8:5039-48.

[11] Ye J, Cheng X, Zhu J, Feng L, Song L. A DDoS Attack Detection Method Based on SVM in Software Defined Network. Cai Z, editor. Secur Commun Networks [Internet]. 2018;2018:9804061. Available from: https://doi.org/10.1155/2018/9804061

[12] Bakker JN, Ng B, Seah WKG, editors. Can Machine Learning Techniques Be Effectively Used in Real Networks against DDoS Attacks? 2018 27th International Conference on Computer Communication and Networks (ICCCN); 2018 30 July-2 Aug. 2018.

[13] Mininet. Online [Available from: Available: http://mininet.org/.

[14] vSwitch O. Online [Available from: http://openvswitch.org/.

[15] RYU. Online [Available from: https://osrg.github.io/ryu/.

[16] OpenFlow Switch Specification (Version 1.3.0), (2012).

[17] Housman, Oxicusa Gugi; Isnaini, Hafida; Sumadi, Fauzi Dwi Setiawan (2020), "SDN-DDOS (ICMP,TCP,UDP)", Mendeley Data, V2, doi: 10.17632/hkjbp67rsc.2