International Journal on Advanced Science Engineering Information Technology

Vol.13 (2023) No. 2
ISSN: 2088-5334

XGBoost Classifier for DDOS Attack Detection in Software Defined Network Using sFlow Protocol

Nadhir Fachrul Rozam [a], Mardhani Riasetiawan [a,*]

[a] *Department of Computer Science and Electronics, Faculty of Mathematics and Natural Sciences, Universitas Gadjah Mada, Yogyakarta, 55281, Indonesia*

Corresponding author: [*]mardhani@ugm.ac.id

***Abstract*— From a security perspective, Software Defined Network (SDN) separates security concerns into Control Plane and Data Plane. The Control Plane is responsible for managing the entire network centrally. Centralized SDN generates high vulnerability against the Distributed Denial of Service (DDOS). When the Software Defined Network overwhelms by DDOS, both Control Plane and Data Plane will lack resources. It can cause the SDN to fail to work if not detected early. Using the ability of sFlow Protocol to capture the flow traffic in real time, the data could be used to detect DDOS attacks. This sFlow sampling approach can reduce the workload of the network by lower down the processing in switches. This paper uses Extreme Gradient Boosting (XGBoost), Support Vector Machine (SVM), and Random Forest as detection methods. We use ONOS as SDN Controller and build the topology in GNS3. Prometheus retrieves data from the sFlow Collector as a time series database. The classifier then uses the data from Prometheus for DDOS detection. For the dataset, we use four different datasets. Datasets 1 and 2 consist of 6109 data, each divided into two classes and three classes. Datasets 3 and 4 consist of 400488 data divided into 2 and 3 classes, respectively. The evaluation results have proved the effectiveness of the proposed method. XGBoost has the highest accuracy of another algorithm. The best accuracy is 99.84% using Dataset 4 as the training set.**

***Keywords*— Software defined network; sFlow; distributed denial of service; extreme gradient boosting.**

Manuscript received 24 May 2022; revised 7 Nov. 2022; accepted 5 Jan. 2023. Date of publication 30 Apr. 2023.

I. INTRODUCTION

The main challenges in developing computer networks are to build a smart network architecture to simplify activation and adjustments to support increasingly diverse application needs. However, traditional networks are less relevant to supporting applications efficiently and massively [1]–[5]**.** SDN separates security concerns into Control Plane and Data Plane. This Control Plane is responsible for managing the entire network centralized on a controller that maintains the whole network so that it is easier to design and operate the network through a logical point. A centralized Control Plane in Software Defined Network (SDN) generates high vulnerability against the Distributed Denial of Service (DDOS) attack. In case of no early detection of this attack, the system immediately reduces the performances of the control plane, data plane, and secure channel between them. In the worst case, it can make SDN fail to work [6]–[8].

The research on DDOS attack detection on SDN is still ongoing to find an effective method. There are several techniques to detect DDOS in SDN, such as entropy-based [9]–[11], connection rate [12], [13], probabilistic [14], traffic pattern analysis [15], and machine learning based. According to Yan et al. [2], one of the effective methods is a machine learning based. The capability of centralized control of the entire node makes implementing big data analytics for attack detection effective. SDN will come to a new stage in proactive, intelligent systems with big data analytics and artificial intelligence.

Most machine learning in SDN uses packets sent from the switch to the controller for control, not traffic monitoring. Those control packets are only sent under certain conditions, and when used continuously, will make high utility of resources [16]–[18]. Furthermore, [19]–[21] creates a module to send control packets for certain intervals to all the switches to get statistical data from the switch. It can use the switch resources, which can be used for forwarding purposes. For the same reason, then [22]–[24] improves to reduce the resource

utility using the sFlow protocol, which can monitor packets on the switch with a sampling technique. In a previous study [22], the research proposes a new sampling technique, Adaptive Polling Sampling. The sFlow sampling protocol method, which has become the industry standard, got better results. sFlow protocol with sampling technique will reduce the load of the network. In addition, the sFlow makes the attack detection more modular, and the network scalability would be better. However, the classification method [22], SAE (Stacked Autoencoder), which belong to unsupervised learning, is not performing better than other paper with supervised learning methods [18], [20], [25]. Unsupervised learning provides convenience by not labeling the dataset, usually used to recognize the similarity of data patterns through clustering. Nevertheless, Supervised Learning would obtain more accurate results regarding data classification whose data class is known.

This paper focuses on classifying DDOS attacks using sFlow as a data source. The data is used from the sFlow Collector as it receives sFlow packets. The combination of DDOS detection in SDN using sFlow to reduce the load and apply the Supervised Learning method to improve classification performance. The supervised learning method applied is Extreme Gradient Boosting (XGBoost). XGBoost is an ensemble learning method. The method has more than one classifier to minimize false negatives (error prediction). For the comparison, we also use Random Forest, which is ensemble learning and Support Vector Machine.

In this paper, we first describe the architecture of SDN. The structural characteristics of the controller will make it vulnerable to DDoS attacks. Then, referring to the simulation tools [26], we use GNS3 for emulating network topology, ONOS [27] as a controller to build our system model, and use Hping3, SlowLoris, and GoldenEye for performing an attack, and Cisco TRex [28] for generating benign traffic. SFlow-RT collects the sampling flow traffic as the sFlow Collector, while Prometheus will get and save the data from sFlow-RT. Finally, the algorithms are used as the detection method. Compared with other classifiers, the DDoS attack detection results show that XGBoost performs more accurately than others. The rest of this paper is organized as follows. In Section II, Materials and Methods, we introduce the SDN's architecture as well as our system model and implementation. The Section III, Results and Discussion, we discuss the evaluation of our model as well as the data training. We have implemented our method in SDN environment and compare it with other classifiers also in this section. In Section IV, we conclude this paper

II. Material and Method

This section discusses the SDN architecture and the vulnerability against DDOS attacks. Our system model and implementation are also explained in this section.

A. SDN Architecture

Open Networking Foundation (ONF) [29] develops the SDN architecture, where network administrators can manage network services centrally. The SDN's programmatic features allow the network administrator to manipulate the network states and configuration with less effort.

SDN architecture separates into three layers: Application, Control, and Infrastructure. This architecture is shown in Fig 1. *Application Layer* is an interface used to manage or develop a SDN Network. Communicate with Control Layer using North Bound Interface (NBI) in this layer. The control Layer is a centralized controller and software-based used to control function and forwarding monitoring. *Infrastructure Layer has network elements used to run* the switching and forwarding packet function. The interface between Controller Plane and Data Plane is called SouthBound Interface (SBI) or Control-to-Data-Plane Interface (CDPI).

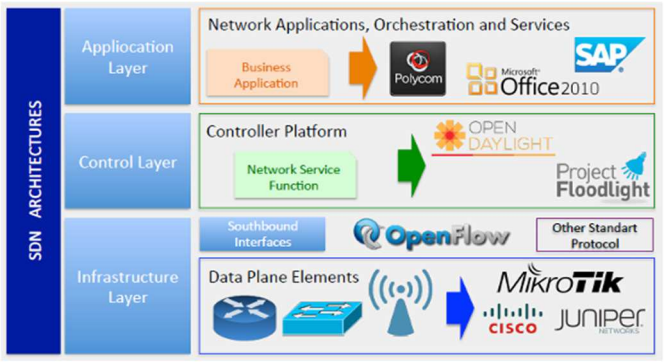

Fig. 1 Software Defined Network Architecture

B. DDOS Attack in SDN

The idea of SDN is separate between Control Plane and Data Plane. The controller is software in Control Plane, which acts as decision maker and is responsible for managing the entire network. It makes it easier to design and operate the network through a logical point. In Data Plane there are network devices such as a switch connected with Controller via OpenFlow protocol. It performs packet forwarding according to the flow table defined by the controller. As shown in Fig 2, when a new packet arrives at the switch, it will look up the flow table if there is a matching entry, as shown by arrow (1). If the flow matches, it will forward to the destination accordingly, as shown by (5). However, if the first packet to the destination is not found in the flow table, the switch will send the packet to SDN Controller as a packet, as shown by (2). The controller then will look up its flow table to see if the rule is matched. If it matches, the controller will send back a new rule to the switch, and the switch will be stored in its flow table for any subsequent packet with the same flow (shown as arrow (3) in the diagram).

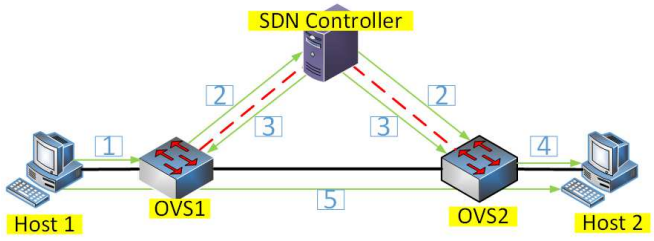

Fig. 2 Packet forwarding in SDN

The DDOS attacks could randomly generate a huge number of malicious packets and could not match the switch flow table, resulting in a large packet sent to the controller. This could overwhelm the control channel and exhaust controller resources [4]. The lack of resources will make the

controller unable to process the legitimate requirement, and this threat could make the controller fail to serve normal users. An attacker may impersonate the controller to steal user information or initiate a much more attack during this time. This DDOS attack can impact various places in SDN network. Three main possible places are the data plane, the control plane, and the secure channel between the control and data plane [30].

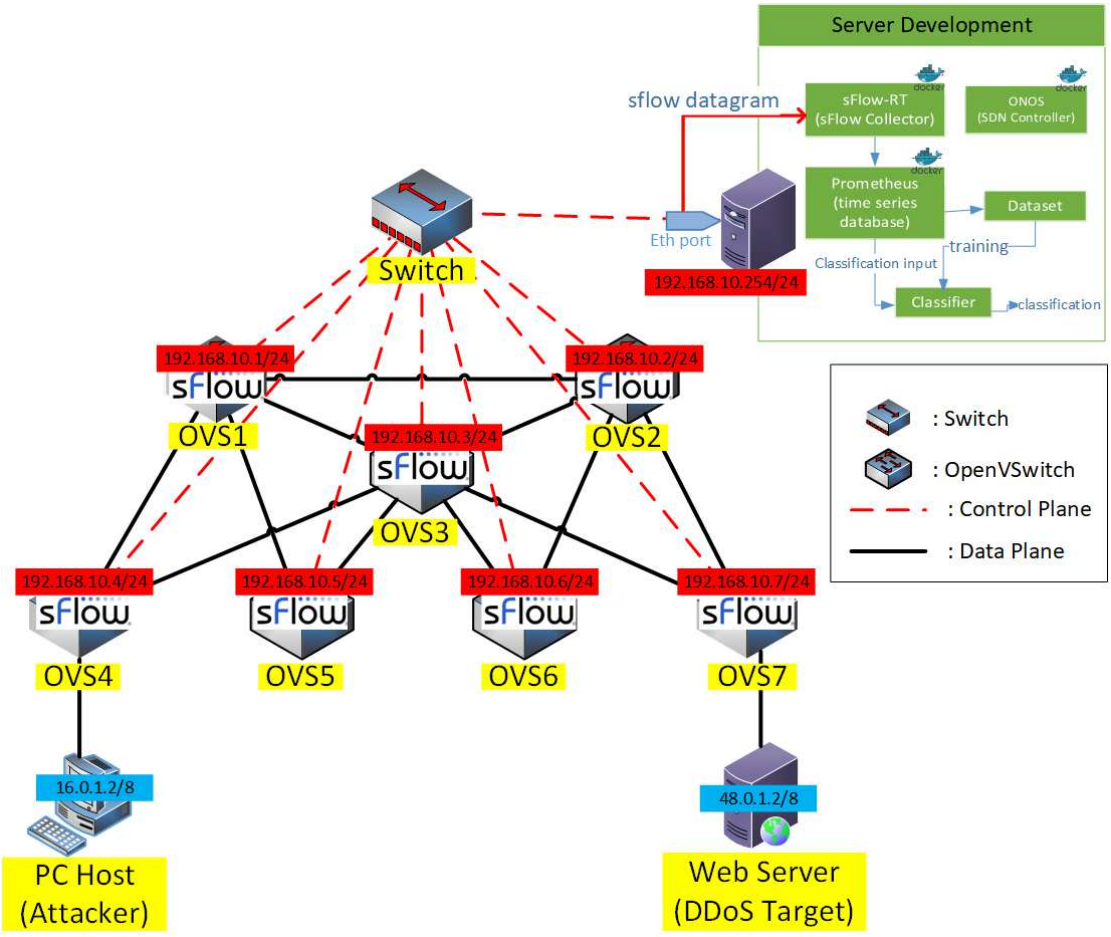

Fig. 3 Proposed system implementation

C. System Model and Implementation

In this paper, the system shown in Fig 3 runs on GNS3 emulator. The red line figured Control Plane path and the black line was Data Plane. We use ONOS as the SDN Controller as it is more modern, modular, and scalable [31]. This experiment will save the data directly from sFlow-RT (Collector) to Prometheus, so the saved data is the data from sFlow Collector. In addition to reducing computer utility, using sFlow-RT and Prometheus make the system more modular. Prometheus is an open-source time series database optimized for gathering large volumes of metrics [32]. We use sFlow-RT as the sFlow Collector, which has all the network traffic data from the sFlow Agent sampling results. The data can be accessed via REST API. Prometheus then gets the data from sFlow-RT every second to be stored in the database.

We use Cisco TRex tools with Advanced Stateful (ASTF) operating mode to generate normal traffic. TRex is an open-source, customizable packet generator tool that can be used to test firewalls and load balancing, as it can concurrently construct packets in multiple streams. In ASTF mode, it allows the tool to work stateful. It can perform TCP communication while monitoring the reply from the server if the TCP session has not ended. One of the advantages of using this advanced stateful mode is being able to emulate connections until the application layer, such as HTTP, DNS, DHCP, etc. This paper uses two hosts for the TRex tool to be configured as a server and a client. The tools for attack traffic run on the client side while there is an Apache2 web server using default configuration on the server. Use GoldenEye, SlowLoris (Application Layer Attack), and Hping3 (Protocol Layer Attack) for DDOS attack tools. The detail of the tools runs on the client and server host are shown in Fig 4.

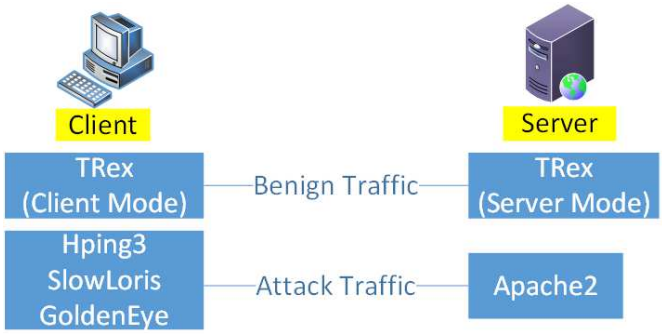

Fig. 4 Used tools in client and server host

The IP address on the attacker's computer and web server (DDOS target) using the /8 subnet accommodates the T-Rex tool to generate normal traffic. TRex will generate traffic with source IP network 16.0.0.0/8 and destination IP network 48.0.0.0/8. In ONOS as the SDN controller, routing is added to make the two networks can be connected to each other. Besides the routing, we need to activate the Reactive Routing application in ONOS. This reactive routing application will create a virtual gateway. Hosts in a legacy IP network use the gateway as the default router to access the Internet. However, an SDN network uses SDN switches to connect a network rather than routers, so the SDN network has no physical gateway router. Without a gateway, there is an issue for hosts inside SDN network. When hosts want to communicate with other hosts in different subnetworks, they do not know the next hop where the packets should be sent. Also, hosts do not know the MAC address of the next hop and cannot compose the entire packet and send it out.

TABLE I
THE DATASETS DETAIL

Dataset	Prometheus Pool Interval	Classes	Number of Data					
			Label 0	Label 1	Label 2	Total	Training Set	Test Set
Dataset 1	1 minute	2	3280	2829	-	6109	4887	1222
Dataset 2	1 minute	3	3280	1629	1200	6109		
Dataset 3	1 second	2	202533	197955	-	400488	320390	80098
Dataset 4	1 second	3	202533	127078	70877	400488		

D. Extreme Gradient Boosting (XGBoost)

Extreme Gradient Boosting (XGBoost) is the learning method to solve regression and classification problems based on the Gradient Boosting Decision Tree (GDBT). In, XGBoost is an improved GDBPT, generating boosted trees efficiently and operating in parallel [33]. XGBoost uses gradient descent to minimize errors when creating new models. Gradient Boosting starts with loading the model into the data specified in Eq (1).

$$F_1(x) = y \tag{1}$$

Furthermore, the results from the model are used to calculate the residuals in the previous process, which is given by Eq. (2). Then a new model is created as written in Eq. (3). The final model is obtained from a combination of *n* iteration models to produce the smallest error value from the residual. The final model can be defined in Eq. (4)

$$h_1(x) = y - F_1(x) \tag{2}$$

$$F_2(x) = F_1(x) + h_1(x) \tag{3}$$

$$\begin{aligned}(x) = F_1(x) \rightarrow\ & F_2(x) = F_1(x) + h_1(x) \ldots \\ & \rightarrow F_M(x) \\ & = F_{M-1}(x) + h_{M-1}(x)\end{aligned} \tag{4}$$

The final model from Boosting which can be written in Eq. (5)

$$f(x) = \gamma_0 + \sum_{m-1}^{M} \gamma_m\, h_m\,(x) \tag{5}$$

where $f_x = \gamma_0$ and $f_m\,(x) = \gamma_m\, h_m\,(x)$ for m = 1,2,3, … M with the value $h_m(x) \epsilon \{-1,1\}$

The final goal of this process is to obtain the closest function $\hat{F}(x)$to $f(x)$ by reduce the value of the loss function $L(y, f(x)$ which is defined in Eq. (6). Finally, the general gradient boosting algorithm formula which can be obtained as in Eq. (7).

$$\hat{F}(x) = argmin_f\, E_{x,y}\big[L\big(y, f(x)\big)\big] \tag{6}$$

$$\{\gamma_m\, h_m\} = argmin \sum_{m=1}^{M}\ L(y_i, f^{(m-1)}(x_i) + \gamma_m\, h_m\,(x_i)) \tag{7}$$

III. RESULT AND DISCUSSION

In this section, we discuss the evaluation of our model as well as the data training. How we get the data and the feature set used are explained.

A. Dataset Source

Tools used for the attack traffic are SlowLoris, GoldenEye, and Hping3. SlowLoris and GoldenEye are both Slow DDOS attacks. These types of attacks exploit the HTTP processing attribute of a web server by maintaining a connection until a HTTP request is completed, and the attacker uses a compromised host to establish a number of connections to the web server and makes the web server unavailable by continuing to send incomplete HTTP requests via each connection [34]. This Slow DDOS or commonly called Low-rate Distributed Denial-of-Service (LDDoS), is a recent evolution of DDoS attack that has emerged as one of the most serious vulnerabilities for the Internet, cloud computing platforms, the Internet of Things (IoT), and large data centers [35].

Meanwhile, Hping3 will be used to send SYN flood attack to exploit the TCP protocol 3-way handshake. For the benign traffic we use TRex with 3 different profiles. Data collection is done by generating packets alternately for all attack tools and profiles from TRex. Each session will generate traffic for 1 hour. Within 1 hour, Prometheus get the data from sFlow-RT with 1 minute and 1 second interval configuration. The process of how the data is collected can be seen in Fig 5. In the labeling for two classes label 0 is normal traffic while label 1 is DDOS attack trafic. For three clasees label 0 is normal traffic, label 1 is Slowloris and GoldenEye attack traffic, label 2 is Hping3 attack traffic. Slowloris and GoldenEye are included in the HTTP DDoS Attack (Application Layer Attack) labeled 1 while Hping3 is the TCP SYN Attack (Protocol Layer Attack) labeled 2. The amount of data for each class in the dataset present in Table I. Dataset will split into training and testing set with the proportion of sharing 80% and 20%, respectively.

TRex configuration for generating benign packets using 3 different profiles. This profile determines what types of traffic and how much traffic is sent. A PCAP file determines the type of traffic, TRex then read the pcap and generate with the rate of the traffic as in the Connection Per Second (CPS) value.

With this scheme TRex could be used for custom traffic generation, simply capture the packet to a PCAP file then define the profile. The first and second profile are called SFR Full and SFR Full 2K. TRex provides both of those profiles in package including the PCAP file. The profiles are defined to meet the IMIX (Internet MIX) [31] testing profile or a mixture of internet traffic. Network equipment vendors use IMIX profiles to simulate real-world traffic patterns and packet distribution. IMIX profiles are based on statistical sampling performed on Internet routers.

Meanwhile the third profile is custom profile. The SFR Full and SFR 2k profiles are more dominant for packet streaming. In the real world, the most packets are for streaming, both streaming for video access and video calls. Therefore, to balance the data in dataset, a custom profile is made from the modified SFR profile. Using the PCAP provided in the package, the thing needs to done is only customize the CPS. For the detail of each profile can be seen in Table II and Fig. 5.

TABLE II
TREX PROFILES

Traffic	Connection per Second (CPS)		
	SFR Full	SFR Full 2k	SFR Custom
HTTP GET	102	404.52	404.52
HTTP POST	102	404.52	404.52
HTTPS	33	130.87	130.87
HTTP Browsing	179	709.89	709.89
Mail Exchange	64	253.81	25.81
POP Mail	1.2	4.759	9.759
POP Mail (Port 111)	1.2	4.759	-
POP Mail (Port 112)	1.2	4.759	-
Oracle	20	79.31	79.31
RTP 160k	0.7	2.776	-
RTP 250k	0.5	1.982	-
SMTP	1.85	7.33	1.85
SMTP (Port 26)	1.85	7.33	1.85
SMTP (Port 27)	1.85	7.33	1.85
Video Call	3	24	1.7
Video Call RTP	7.4	29.347	-
Citrix	11	43.62	7
DNS	498	417	498.0
SIP	7.4	29.34	5.4
RSTP	1.2	4.8	2.2

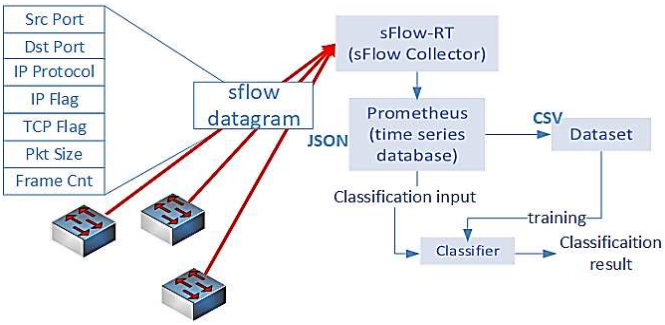

Fig. 5 Data gathering from sFlow

TABLE III
EVALUATION RESULTS

	Random Forest				Support Vector Machine				Extreme Gradient Boosting			
Train Dataset	1	2	3	4	1	2	3	4	1	2	3	4
Dataset Test Accuracy	99.59	99.67	99.81	99.81	99.59	98.45	99.47	99.48	99.75	99.75	99.84	99.84
Realtime Test Accuracy	98.60	99.65	99.80	99.84	99.43	95.22	99.82	99.88	99.55	99.72	99.90	99.78
Dataset Test Precision	99.82	99.80	99.81	99.83	99.46	98.02	99.47	99.52	100	99.85	99.84	99.86
Realtime Test Precision	98.27	99.77	99.94	99.90	99.28	95.39	99.95	99.91	99.76	99.82	100	99.82
Dataset Test Recall	99.28	99.60	99.81	99.85	99.64	97.96	99.47	99.56	99.46	99.70	99.84	99.87
Realtime Test Recall	99.18	99.61	99.67	99.83	99.53	91.07	99.68	99.85	99.33	99.71	99.78	99.76
Dataset Test F1	99.55	99.70	99.81	99.84	99.55	97.98	99.47	99.54	99.73	99.77	99.84	99.86
Realtime Test F1	98.72	99.69	99.80	99.86	99.40	92.57	99.81	99.88	99.55	99.76	99.89	99.79

To evaluate the model, along with test set from the dataset we evaluate from the real-time data. Real-time data is obtained by generating data with the tool. While the tools generating data, the sampling results from sFlow-RT is stored by Prometheus. At the same time data can be retrieved the data from Prometheus. The data from Prometheus then classified directly by the machine learning model. For real-time testing, the data is taken every second for 1 minutes, 5 minutes, and 10 minutes respectively for each tools. The classification output of the model is saved and evaluated. Because the data source is come from sFlow, the dataset features are closely related to sFlow capabilities to sampling the traffic packets. Different device version might has different capability of the sFlow Agent to sample the traffic flow. Here we determine the suitable features we can use to classify the traffic—the features used in this paper as shown in Table IV.

Common features such as IP Source, IP Destination, and Time to Live (TTL) could be use also as it is supported in sFlow. However, it is irrelevant for data modelling as it will change in different network topology and scenarios.

TABLE IV
THE DATASET FEATURES

Feature	Description
ipflags	IP Flags in binary (3-bit)
sourceport	Source port
destinationport	Destination port
ipbytes	IP packet size in bytes
frames	Frame per sampling
ipprotocol	Type of ip protocol. Eg. 6 for TCP, 17 for UDP.
tcpflags	TCP Flags in binary (8-bit)
payload	Payload size in bytes
bytes	Packet request size in bytes
frame	Number of frames from flow sampling

B. Detection Results and Analysis

In this section, we discuss the results of classification using XGBoost and compared it with two other machine learning algorithms, Random Forest and Support Vector Machine (SVM). The main comparison measures are classification

accuracy, precision, recall and F1. The evaluation also compares the real-time data with test set result.

TABLE V
BEST PARAMETERS FOR EACH ALGORITHM

Algorithm	Parameter	Best Parameter Value			
		D1	D2	D3	D4
SVM	Kernel	poly	poly	poly	poly
	C	5	15	10	10
RF	N Estimator	75	50	75	100
	Max Feature	sqrt	sqrt	log2	auto
XGB	N Estimator	100	25	75	100
	Learning Rate	0.1	1	0.5	0.75
	Tree Method	approx	approx	auto	auto

Each algorithm is trained with all four datasets. In each training, GridsearchCV will find the best parameters of the algorithm. Cross validation for GridsearchCV is set to 5-fold. There are different parameters for each algorithm.

For Random Forest, hyperparameter used to find the best combination of parameters are as follow:

- N Estimator: 25,50,75, and 100
- Max Feature: sqrt, log2, and auto

For Support Vector Machine, hyperparameter used are as follow:

- Kernel: Linear, RBF, and Polynomial
- C : 0.1, 0.5, 1, 5, and 10

For Xtreme Gradient Boosting, hyperparameter used are as follow:

- N Estimator: 25, 50, 75, and 100
- Learning Rate: 0.1, 0.3, 0.5, 0.75 , and 1
- Tree Method: auto, exact, approx, hist, and gpu_hist

To find the best parameters of each algorithm GridsearchCV try each parameter combination and do training and cross validation. The result in Table III is the score of the best parameter. For the best parameters of each dataset can be seen in Table V.

As shown in Fig. 6 and Fig. 7, each dataset for each algorithm has an accuracy value above 99%, except for the Support Vector Machine with Dataset 2 which has 98,45%. The Support Vector Machine has decreased with Dataset 2 with 3 classes, but in Dataset 4 which also has 3 classes SVM get accuracy score above 99%. It means SVM with a smaller number of sample data has an impact on classification performance. The unseen data could not be classified correctly by SVM. The unseen data exist because in the Dataset 2 Prometheus pooling interval are set for 1 minutes. The data in intervals of one minute may be different from the next minute. The SVM with the hyperplane made was not able to handle it. On the other hand, both ensemble learning algorithms with a decision tree basis i.e. Random Forest and XGB, can actually handle it. By increasing the amount of training data can also increase the accuracy of the Random Forest and XGB algorithms. XGBoost has the highest accuracy with Dataset 3 and 4 training set with 99,84% accuracy score followed by Random Forest with 99,81% for the same dataset.

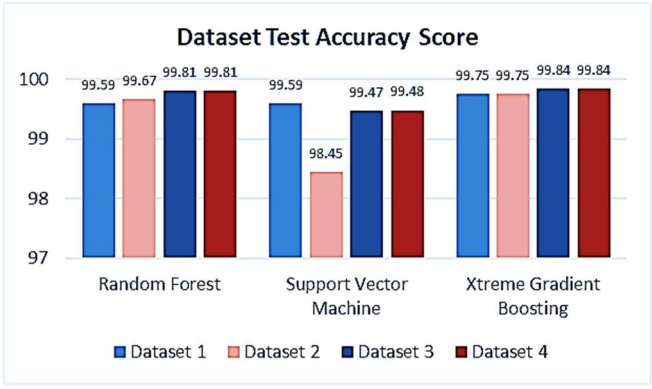

Fig. 6 Accuracy result from dataset test

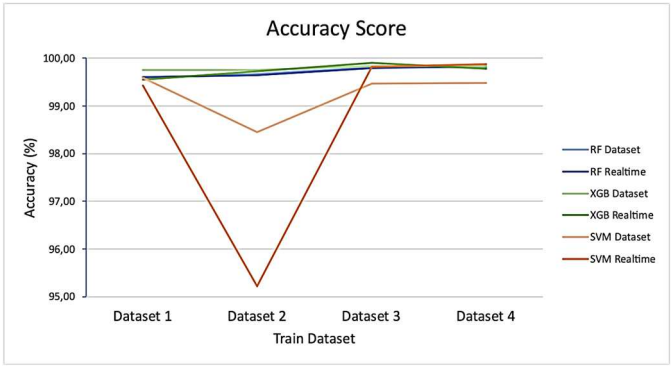

Fig. 7 Accuracy comparison from real-time test and dataset test results

Furthermore, the detail of evaluation shown in Table III. The values of precision, recall, and F1 displayed in the table are the average value of all classes. Overall, the Xtreme Gradient Boosting algorithm get the highest score among other algorithms for each dataset. Extreme Gradient Boosting Algorithm in terms of accuracy is also quite stable for each dataset. Even for classification with 3 classes this algorithm has better scores than classification with 2 classes. This can be interpreted that the data is indeed more suitable to be categorized into three categories. Hping3 attack traffic, a Protocol Layer Attack category, is better to be separated from SlowLoris and GoldenEye attacks, which are Application Layer Attack categories.

As in Table III, most algorithms have insignificant differences between tests using datasets and real-time data. This means that most classification models can be applied to real-time traffic classification. As shown in Fig 7, the biggest difference in the test result is the Support Vector Machine algorithm with Dataset 2. SVM has the worst results than other algorithms for classifying three classes with the low amount of data. The difference between real-time test and dataset test for SVM with Dataset 2 also the highest, indicating that the model is not robust when applied to traffic classification in real time. This high difference is because during the real-time test Prometheus interval pooling configuration is set at 1 second while the training data from Dataset 2 is data taken every 1 minute. SVM cannot perform with the Dataset 2 test, it is intended with real-time data will be worse as they will be more unseen data while classifying the traffic. Other than SVM, Random Forest and Xtreme Gradient boosting has similar results between dataset and real-time tests. Both algorithms have excellent performance

in multiple class classifications even with the less data training data.

IV. Conclusion

This research shows that sFlow can be used as a data source without any extract module. Sflow-RT and Prometheus are works together to provides real-time data. The data sampling result is consistent according to the real-time data test compared to dataset test. The best classification model for DDOS attack detection on SDN in this study is using the Xtreme Gradient Boosting algorithm with training set Dataset 4. From the test results, Dataset 4 with 3 classes has the most optimal results for each algorithm. The classification model using a combination of XGBoost – Dataset 4 has an accuracy of 99,84% for test with a test set from the dataset and 99,78% with real-time data. With Dataset 4 also, SVM and Random Forest has lower results with 99,48% and 99,81% accuracy, respectively. XGB and Random Forest are extremely stable for all dataset with two or three classes. Lower amount of the data in dataset still has a good result for XGB and Random Forest, but apparently it is not for SVM. The ensemble learning algorithm worked very well in this study. Both boosting and bagging method with the Decision Tree base are performing well for the classification. Binary classification with SVM has pretty good results, but for multiclass classification SVM requires more data to perform better.

For the further research, it is suggested to compare with other learning algorithm with the difference approach. The other suggestion is to make a mitigation action to counter the DDOS with the proposed method in this study. There is a potential to create a machine learning based framework for mitigating the DDOS attack as all the server applications run in a Docker Container. This will make the implementation modular and fast to deploy. For a large scale of the implementation a benchmark would be a great research. The dataset used in this study can also be used for similar research that use sFlow as a data source.

References

[1] Y. Zhao, Y. Li, X. Zhang, G. Geng, W. Zhang, and Y. Sun, "A Survey of Networking Applications Applying the Software Defined Networking Concept Based on Machine Learning," *IEEE Access*, vol. 7, pp. 95397–95417, 2019, doi: 10.1109/ACCESS.2019.2928564.

[2] Q. Yan, F. R. Yu, Q. Gong, and J. Li, "Software-defined networking (SDN) and distributed denial of service (DDOS) attacks in cloud computing environments: A survey, some research issues, and challenges," *IEEE Commun. Surv. Tutorials*, vol. 18, no. 1, pp. 602–622, 2016, doi: 10.1109/COMST.2015.2487361.

[3] S. Gupta and D. Grover, "A Comprehensive Review on Detection of DDoS Attacks using ML in SDN Environment," *Proc. - Int. Conf. Artif. Intell. Smart Syst. ICAIS 2021*, pp. 1158–1163, 2021, doi: 10.1109/ICAIS50930.2021.9395987.

[4] S. Kaur, K. Kumar, and N. Aggarwal, "Analysis of DDoS Attacks in Software Defined Networking," *2022 IEEE Delhi Sect. Conf. DELCON 2022*, 2022, doi: 10.1109/DELCON54057.2022.9753224.

[5] N. Sultana, N. Chilamkurti, W. Peng, and R. Alhadad, "Survey on SDN based network intrusion detection system using machine learning approaches," *Peer-to-Peer Netw. Appl.*, vol. 12, no. 2, pp. 493–501, 2019, doi: 10.1007/s12083-017-0630-0.

[6] Y. Cui *et al.*, "Towards DDoS detection mechanisms in Software-Defined Networking," *J. Netw. Comput. Appl.*, vol. 190, no. November 2020, p. 103156, 2021, doi: 10.1016/j.jnca.2021.103156.

[7] M. P. Singh and A. Bhandari, "New-flow based DDoS attacks in SDN: Taxonomy, rationales, and research challenges," *Comput. Commun.*, vol. 154, no. March, pp. 509–527, 2020, doi: 10.1016/j.comcom.2020.02.085.

[8] B. Mladenov, "Studying the DDoS Attack Effect over SDN Controller Southbound Channel," in *2019 X National Conference with International Participation (ELECTRONICA)*, May 2019, pp. 1–4, doi: 10.1109/ELECTRONICA.2019.8825601.

[9] Institute of Electrical and Electronics Engineers, "Detection of DDoS in SDN Environment Using Entropy-based Detection," in *2019 IEEE International Symposium on Technologies for Homeland Security (HST)*, Nov. 2019, pp. 1–4, doi: 10.1109/HST47167.2019.9032893.

[10] R. Neres Carvalho, J. Luiz Bordim, and E. Adilio Pelinson Alchieri, "Entropy-based DoS attack identification in SDN," in *Proceedings - 2019 IEEE 33rd International Parallel and Distributed Processing Symposium Workshops, IPDPSW 2019*, May 2019, pp. 627–634, doi: 10.1109/IPDPSW.2019.00108.

[11] U. Ahmed, J. C. W. Lin, and G. Srivastava, "Network-Aware SDN Load Balancer with Deep Active Learning based Intrusion Detection Model," *Proc. Int. Jt. Conf. Neural Networks*, vol. 2021-July, 2021, doi: 10.1109/IJCNN52387.2021.9534424.

[12] B. H. Lawal and A. T. Nuray, "Real-time detection and mitigation of distributed denial of service (DDoS) attacks in software defined networking (SDN)," *26th IEEE Signal Process. Commun. Appl. Conf. SIU 2018*, pp. 1–4, 2018, doi: 10.1109/SIU.2018.8404674.

[13] M. Imran, M. H. Durad, F. A. Khan, and H. Abbas, "DAISY: A Detection and Mitigation System against Denial-of-Service Attacks in Software-Defined Networks," *IEEE Syst. J.*, vol. 14, no. 2, pp. 1933–1944, 2020, doi: 10.1109/JSYST.2019.2927223.

[14] P. Maity, S. Saxena, S. Srivastava, K. S. Sahoo, A. K. Pradhan, and N. Kumar, "An Effective Probabilistic Technique for DDoS Detection in OpenFlow Controller," *IEEE Syst. J.*, vol. 16, no. 1, pp. 1345–1354, 2022, doi: 10.1109/JSYST.2021.3110948.

[15] Y. Wang, T. Hu, G. Tang, J. Xie, and J. Lu, "SGS: Safe-Guard Scheme for Protecting Control Plane Against DDoS Attacks in Software-Defined Networking," *IEEE Access*, vol. 7, pp. 34699–34710, 2019, doi: 10.1109/ACCESS.2019.2895092.

[16] F. Khashab, J. Moubarak, A. Feghali, and C. Bassil, "DDoS Attack Detection and Mitigation in SDN using Machine Learning," *Proc. 2021 IEEE Conf. Netw. Softwarization Accel. Netw. Softwarization Cogn. Age, NetSoft 2021*, pp. 395–401, 2021, doi: 10.1109/NetSoft51509.2021.9492558.

[17] C. B. Zerbini, L. F. Carvalho, T. Abrão, and M. L. Proença, "Wavelet against random forest for anomaly mitigation in software-defined networking," *Appl. Soft Comput. J.*, vol. 80, pp. 138–153, 2019, doi: 10.1016/j.asoc.2019.02.046.

[18] R. Santos, D. Souza, W. Santo, A. Ribeiro, and E. Moreno, "Machine learning algorithms to detect DDoS attacks in SDN," *Concurr. Comput. Pract. Exp.*, vol. 32, no. 16, pp. 1–14, 2020, doi: 10.1002/cpe.5402.

[19] M. Myint Oo, S. Kamolphiwong, T. Kamolphiwong, and S. Vasupongayya, "Advanced Support Vector Machine-(ASVM-) based detection for Distributed Denial of Service (DDoS) attack on Software Defined Networking (SDN)," *J. Comput. Networks Commun.*, vol. 2019, 2019, doi: 10.1155/2019/8012568.

[20] K. S. Sahoo *et al.*, "An Evolutionary SVM Model for DDOS Attack Detection in Software Defined Networks," *IEEE Access*, vol. 8, pp. 132502–132513, 2020, doi: 10.1109/ACCESS.2020.3009733.

[21] R. Fadaei, O. Ermiş, and E. Anarim, "A DDoS attack detection and countermeasure scheme based on DWT and auto-encoder neural network for SDN," *Comput. Networks*, vol. 214, no. March, p. 109140, 2022, doi: 10.1016/j.comnet.2022.109140.

[22] R. M. A. Ujjan, Z. Pervez, K. Dahal, A. K. Bashir, R. Mumtaz, and J. González, "Towards sFlow and adaptive polling sampling for deep learning based DDoS detection in SDN," *Futur. Gener. Comput. Syst.*, vol. 111, pp. 763–779, 2020, doi: 10.1016/j.future.2019.10.015.

[23] M. Wang, Y. Lu, and J. Qin, "Source-Based Defense Against DDoS Attacks in SDN Based on sFlow and SOM," *IEEE Access*, vol. 10, pp. 2097–2116, 2022, doi: 10.1109/ACCESS.2021.3139511.

[24] Z. A. El Houda, A. S. Hafid, and L. Khoukhi, "A Novel Machine Learning Framework for Advanced Attack Detection using SDN," *2021 IEEE Glob. Commun. Conf. GLOBECOM 2021 - Proc.*, 2021, doi: 10.1109/GLOBECOM46510.2021.9685643.

[25] H. A. Alamri and V. Thayananthan, "Bandwidth control mechanism and extreme gradient boosting algorithm for protecting software-defined networks against DDoS attacks," *IEEE Access*, vol. 8, pp. 194269–194288, 2020, doi: 10.1109/ACCESS.2020.3033942.

[26] S. Sirijaroensombat, C. P. Nangsue, and C. Aswakul, "Development of software-defined mesh network emulator testbed for DDoS defence study," *2019 IEEE 4th Int. Conf. Comput. Commun. Syst. ICCCS 2019*, pp. 468–472, 2019, doi: 10.1109/CCOMS.2019.8821667.

[27] P. Berde *et al.*, “ONOS: Towards an Open, Distributed SDN OS,” pp. 1–6, 2014, doi: 10.1145/2620728.2620744.

[28] TRex, “Cisco T-Rex: Realistic traffic generator.,” *https://trex-tgn.cisco.com/*. https://trex-tgn.cisco.com/ (accessed May 21, 2022).

[29] O. N. Foundation, “Software-Defined Networking: The New Norm for Networks [white paper],” *ONF White Pap.*, pp. 1–12, 2012, doi: citeulike-article-id:12475417.

[30] A. T. Kyaw, M. Zin Oo, and C. S. Khin, “Machine-Learning Based DDOS Attack Classifier in Software Defined Network,” in *2020 17th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON)*, Jun. 2020, pp. 431–434, doi: 10.1109/ECTI-CON49241.2020.9158230.

[31] K. Smida, H. Tounsi, M. Frikha, and Y. Q. Song, “Efficient SDN Controller for Safety Applications in SDN-Based Vehicular Networks: POX, Floodlight, ONOS or OpenDaylight?,” *2020 8th Int. Conf. Commun. Networking, ComNet2020 - Proc.*, pp. 1–6, 2020, doi: 10.1109/ComNet47917.2020.9306095.

[32] A. Bader, O. Kopp, and M. Falkenthal, “Survey and Comparison of Open Source Time Series Databases,” *Gesellschaft für Inform.*, vol. P-266, pp. 249–268, 2017.

[33] T. Chen and C. Guestrin, “XGBoost: A Scalable Tree Boosting System,” in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Aug. 2016, vol. 42, no. 8, pp. 785–794, doi: 10.1145/2939672.2939785.

[34] J. Singh and S. Behal, “Detection and mitigation of DDoS attacks in SDN: A comprehensive review, research challenges and future directions,” *Comput. Sci. Rev.*, vol. 37, p. 100279, 2020, doi: 10.1016/j.cosrev.2020.100279.

[35] A. A. Alashhab, M. S. M. Zahid, M. A. Azim, M. Y. Daha, B. Isyaku, and S. Ali, “A Survey of Low Rate DDoS Detection Techniques Based on Machine Learning in Software-Defined Networks,” *Symmetry (Basel).*, vol. 14, no. 8, p. 1563, Jul. 2022, doi: 10.3390/sym14081563.