*IOSR Journal of Computer Engineering (IOSR-JCE)*
*e-ISSN: 2278-0661,p-ISSN: 2278-8727, Volume 22, Issue 5, Ser. I (Sep. – Oct. 2020), PP 49-54*
*www.iosrjournals.org*

# Survey on Software Defined Networking based Network Intrusion Detection and Prevention System

Sunidhi[1], Abhishek Verma[2]
[1](Department of Computer Science, Manipal Institute of Technology, Manipal, India)
[2](Department of Computer Science, Siddaganga Institute of Technology, Tumkur, India)

***Abstract:****In recent years, with the rapid development of the current Internet and mobile communication technologies, the infrastructure, devices, and networking systems' resources are becoming more complex and heterogeneous. In order to efficiently organize, manage, maintain, and optimize networking systems, there is a need for the deployment of more intelligence. However, due to the inherently distributed feature of traditional networks, machine learning techniques are hard to be applied and deployed to control and operate networks. Software defined networking (SDN) brings us new chances to provide intelligence inside the networks. SDN capabilities (logically centralized control, global view of the network, software-based traffic analysis, and dynamic updating of forwarding rules) make it easier to apply machine learning techniques. This survey paper features different machine learning algorithms to secure software-defined networking from multiple malicious attacks.*

***Key Word****: Software defined networking, machine learning*

---

Date of Submission: 20-09-2020 Date of Acceptance: 05-10-2020

---

## I. Introduction

Software-defined networking (SDN) represents an emerging centralized network architecture in which a central unit manages the forwarding elements, called an SDN controller. SDN controller can obtain traffic statistics from each forwarding element to take the appropriate action required for preventing any malicious behavior or abusing the network. Simultaneously, the SDN controller uses a programmable network protocol, which is OpenFlow (OF) protocol, to communicate and forward its decisions to OF-enabled switches [1]. In SDN, we decouple the Data plane and control plane to help the network device work effectively. The Data plane or forwarding plane is concerned about moving the user traffic packets based on some rules, and these Control Plane define these rules. As pointed out, [2] control plane easily can be the victim of a Distributed Denial of Services (DDoS) attack, and all popular controllers suffer from security threats.

### 1. Software-DefinedNetworking

SDN is the network architecture where there is a separation of network control from the forwarding mechanism. There are three layers in SDN, such as the Infrastructure layer, the Control layer, and theApplication layer. The data plane refers to the Infrastructure layer and simple network devices- switches constitute this plane. These switches do not have any higher networking functions and only drop and forward the incoming packets according to the flow table configured from the control plane through the southbound protocol. The first standard in SDN uses the OpenFlow protocol. OpenFlow is defined in the OpenFlow switch specification published by Open Network Foundation (ONF) [3]. The upper layer refers to the Control plane, provides the interconnection of applications on the top and bottom of the architecture. The controller maintains a centralized view of the network. It allows applications to control the underlying network through open interfaces. The third layer is also called the management plane. The upper layer is composed of applications, managing, and securing the underlying network. The application could be running on the controller or communicating throughthenorthboundApplicationProgramming Interface (API) of theController.

### 2. Intrusion Detection and PreventionSystem

The Intrusion Detection System is a system that monitors network traffic to avoid anomaly in the network. Intrusion is any action attempted to compromise the availability, integrity, and security of any resource or service. Intrusion Detection System detects any threat trying to fiddle with these parameters of the network. Currently, an effective intrusion detection system can be classified as anomaly detection or misuse detection. Anomaly detection is a two-step approach to first train a system to identify a particular behavior that is normal to the system and raise an alert if any deviation beyond this behavior is detected.

## II. Literature Review

Abhilash G et al. trained OpenDaylight SDN Controller to distinguish between a regular or anomalous packet. For training the SDN controller against any anomalous packet [4], machine learning algorithms were applied. For testing and training controllers, the NSL KDD dataset was considered [5]. Attacks were detected using the classifier algorithms. After attack identification, for mitigation controller was notified.

In [6], an extensive survey of SDN-based DDoS attack detection techniques was implemented. DDoS attacks revolve around the fact that numerous sources are distributed across multiple locations to target a victim. For the smart city data center, the SDN-based proactive DDoS Defense Framework (ProDefense) was proposed. Low false-positive and high detection rates were desirable characteristics for an effective DDoS protection mechanism. Different filters were formulated to customize attackdetection.

In [7], the SDN framework was designed to identify and defend against DDoS attacks. This framework consisted of three parts: the traffic collection module, attack identification module, and flow table delivery module. The traffic collection module extracted traffic characteristics to prepare for traffic identification. As per the author, the accuracy of 0.998 was achieved for identifying DDoS attacks. The controller discarded packets according to the predefined rule if attack traffic was identified.

In [8], the author used seven machine learning techniques to accurately classify and predict different DDoS attacks like Smurf, UDP flood, and HTTP flood. Experimental results with proper analysis had been presented in this work. In order to detect attacks, Intrusion Detection Systems (IDS) follow two different approaches: signature-based or anomaly-based detection. Among seven classifier linear regression achieved high accuracy, precision, recall results, while Naive Bayes showed the worst result.

To mitigate DDoS attacks in SDN, the author combined the programmability power of SDN with the intelligence of machine learning [9]. NSL-KDD dataset was used to train the designed model for DDoS mitigation, and then the trained model is then tested on the real DDoS attack. The communication protocol between the SDN controllers was implemented to create a faster and more reliable DDoS mitigation method.

In paper [10], an anomaly detection method presented, called Saturation Attack-Detector, for dealing with a family of saturation attacks. SA-Detector was built upon the study of self-similarity of OpenFlow traffic, which showed that the normal and abnormal traffic patterned between the controller and the OpenFlow switches have different characteristics. The experimental results showed that the average accuracy was 96.54%, and the average precision was 92.06%. Hence, it indicated that SA-Detector was useful for detecting saturation attacks.

A hybrid machine learning model was used to protect the Controller from DDoS attacks [11]. The authors had proposed a combination of two machine learning-based models with Support Vector Machine (SVM) and Self Organized Map (SOM). Experimental results show that the hybrid machinelearningmodelprovidedmoreaccuracy,detectionrate, and less false alarm rate than simple machine learning models. The proposed hybrid model provided high accuracy of 96.77%, a high detection rate of 90.45%, and a low false alarm rate of0.032%.

DDoS attack detection system for SDN used two levels of security proposed [12]. First, signature-based attacks detected using Snort. Further, the author used machine learning algorithms to detect anomaly-based attacks. Two algorithms, namely, Support Vector Machine (SVM) classifier and the Deep Neural Network (DNN), were used to create a trained model based on the KDD Cup dataset. The system evaluated in the SDN environment created using a Mininet emulator with a Ryu controller. From the results, DNN had higher precision and accuracy value compared to that of the SVM classifier.

In paper [13], the author showed how DDoS attacks exhausted controller resources and provided a solution to detect such attacks based on the destination IP address's entropy variation. In the proposed solution, the randomness of the incoming packets was measured. A good measure of randomness is entropy. Based on the simulations done in this paper, an experimental threshold was chosen for entropy, and values lower than threshold values would be considered attacks.

The author used an advanced Support Vector Machine (ASVM) algorithm [3] to detect the DDoS attack. In the ASVM algorithm, the algorithm's input was traffic data, and the output resulted in the detection of a DDoS attack. The performance evaluation was measured using the training and testing time analysis to calculate the DDoS attack's false alarm rate and the detection rate on the packets flow. Performance attributes were true positive (TP), true negative (TN), false positive (FP), and false-negative (FN). ASVM ensured more detection rate and lowered false alarm rate as compared to SVM.

Software-defined networking (SDN) is a new paradigm that allows for developing more flexible network applications. In a study [14], a subset of features extracted from the NSL-KDD dataset based on the principal components analysis (PCA) approach and following supervised machine learning approaches were considered: decision tree (DT), extreme learning machine (ELM), Naive Bayes (NB), linear discriminant analysis (LDA), neural networks (NNs), support vector machines (SVM), random forest (RT), K-nearest-neighbor (KNN), AdaBoost, RUSBoost, LogitBoost, and Bagging Trees. DT approach showed the best performance in terms of accuracy, precision,F1-measure.

In work [15] author described a simple architecture deployed in an enterprise network that gathered traffic data using the Open Flow protocol. For obtaining the dataset studied in this work, a single OpenFlow switch was deployed in a non-SDNenterprise production network. The experimenter tested several ensemble learning classifiers; these classifiers used a set of regular classifiers and classified data by taking a weighted vote of each individual's prediction.

A policy-driven security architecture for securing end-to-end services across multiple autonomous domain-based SDN environments was proposed [16]. There was a language-based approach in designing a range of security policies relevant to SDN services and communications. Each controller maintained and updated a Topology Repository and a Policy Repository. The core component of PbSAwas a Policy Manager, who managed everyoperation of the security system. A Packet Handle Creator module created the necessary handles for AS domains, piggybacked with the Policy Manager's payload. These handles were used to check the packet's authenticity and the enforcement of policies at theswitches.

In paper [17], several attack scenarios were thoroughly addressed by the research community. Three scenarios had been discussed. Firstly, a network that contained malicious hosts, but where the network itself was not infected. Then a network that contained malicious switches but was operated by a benign controller. Lastly, a network in which the switches were benign, but the controller was compromised. This was the most unlikely and most critical scenario where compromised nodes in a software-defined network were demonstrated.

The work [18] focused on designing and developing an OpenFlow based firewall application. The implementation showed that most of the firewall functionalities could be built using software without dedicated hardware. POX controller was used based on python for experiments. VMWare virtualization solution and Mininet emulator was used for creating network topologies. Further, calculating values of latency and throughput with or without firewall rules. NA (Network Administrator) could modify or extend firewallcode.

A firewall is a program or device that looks as a determent to keep pernicious elements out of the network. Firewalls use many techniques to control traffic flowing in and out of networks. Firewalls work as filters for network traffic by blocking incoming or outgoing packets of information that are seen as unsafe according to the ruleset. Here [19] author had analyzed some software-defined firewall (SDF) techniques and challenges related to a software-defined network. Existing firewall technology was designed based on the OpenFlow prototype. The firewall controller controlled the traffic flow based on the rule set defined in the flow table and the switch that was enforced on the controller regulating traffic flow according to their flow table rules.

## III. Methodology Used

Attacks were detected by the author [4] using the classifier algorithms. Once this attack was identified, the controller was notified, and the mitigation was performed. Had the packets been classified as ordinary flow packet, they would have allowed in the network without further processing. Various parameters were associated with a network packet, but parameters having significant importance have been picked in simulating an attack. Some of those parameters were: duration of the connection, protocol, service type, connection status flag, bytes sent, bytes received, and in SDN influenced network, these parameters were Average of packets per flow (APf), Average of Bytes per flow (ABf), Average Duration per flow (ADf), Percentage of Pair-flows (PPf), Growth of Single-flows (GSf) and Growth of Different Ports (GDP). With the help of these parameters, classification can be done efficiently and effectively.

### 1. Algorithm Used

### 1.1. Support Vector Machine (SVM):

SVM is a supervised learning classifier that finds optimal separating hyperplane between points of two classes. SVM works well enough because it can handle nonlinear, sparse, high-dimensional data efficiently. An optimal hyperplane is onethat has the maximum possible distance from the nearest point of either class. However, in general, problems are not linearly separable. SVM tackles such issues using kernel functions, which convert nonlinear problems to linear problems. A central unit manages the forwarding elements, called an SDN controller, to obtain traffic statistics from each forwarding element to take the appropriate action required for preventing any malicious behavior or abusing the network. The author used Smile machine learning library's [20] Gaussian mercer kernel with different smoothing and soft margin penalties in the setup.

### 1.2. Decision Tree (DT) Algorithm:

Decision Tree Algorithm also belongs to the family of the Supervised learning algorithm, which, in the used case, served the purpose of creating a logical decision rule learning which predicts the nature of incoming packets. This logical decision rule tree is built based on information gain calculated for each class. For a completely logical decision tree, calculate information gain for every split, and from these calculated splits, the best one is chosen (for which the information gain is maximum). This process is used to build the tree

recursively. Since, at each split, maximum gain is chosen, i.e., greedy method, we can be assured that the resulted tree is the best solution that we can get from this method.

**2. Developing Plugins**

OSGi (Open Services Gateway initiative) framework is required with OpenDaylight Controller for services deployment. There is some difference between OSGi bundles and Jars. A useful OSGI bundle requires some data which could be used by the OSGi framework. This data is stored in its manifest and known as OSGi metadata. Without this metadata, the OSGi framework would not import bundles classes, i.e., these classes would be invisible to the framework. Three bundles are developed.

**2.1. First Bundle**

The first bundle provides java bindings and APIs of the services. These APIs are generated by the data modeldefined through yang. ODL ready APIs are classified into three parts, and they are as follows:

**2.1.1. API to train the model**

This API would be used by any SDN northbound application to train the controller. Every time a new intrusion signature is identified, it can be used to train the controller with the updated information.

**2.1.2.API to test this trained model**

This API would be used by the same application, which involved training the controller. This API checks the accuracy of the model training. The higher the accuracy attained, the better the training model.

**2.1.2API to predict the runtime traffic nature**

This API would be used to predict runtime traffic. The logic would consume this data and, in return, detects if the input is anomalous ornot.

**2.2. Second Bundle**

The second bundle contains the implementation of the java bindings and machine learning logic. This bundle provides two OpenDaylight modules. To train, test, and save the SVM classifier, the second one loads the classifier and predicts if the test dataset belongs to intrusion. These modules are capable of training, testing, and predicting with any dataset in the ARFF format. Datasets and models can be provided through URLs, making it a flexible service to train the controller. This functionality enables us to train the controller on the go whenever a new training set shows up.

**2.3. Third Bundle**

The third bundle contains dependencies for the first two bundles in OSGi ready format. Mostly dependency JAR files are not OSGi compatible. OSGi ready jar file is needed for OpenDaylight, i.e., KAR (Karaf Archive) is required. Eclipse IDE is used for thispurpose.

The basic flow for the creation of the above three bundles [4] can be seen pictorially from Figure1.

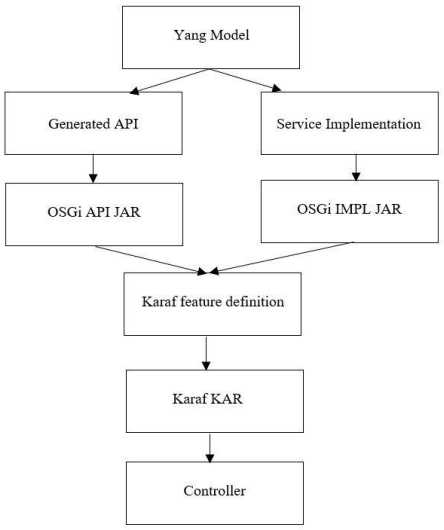


**Figure 1:** Basic flow for the creation of three bundles

## IV. Results

The controller was trained with various machine learning classification algorithms, such as Support Vector Machine (SVM), Naive- Bayes Algorithm, and Decision Tree algorithm. After training SDN controller classifiers were saved for future predictions, the experiment results performed [4] are presented in Table 1.

| Algorithm | TP Rate | FP Rate | Class | Accuracy |
|---|---|---|---|---|
| Support Vector Machine (SVM) | 0.925 | 0.375 | Normal | 75.395% |
| | 0.625 | 0.075 | Anomaly | 75.395% |
| Naïve-Bayes Algorithm | 0.997 | 0.567 | Normal | 67.167% |
| | 0.433 | 0.003 | Anomaly | 67.167% |
| Decision Tree (DT) | 0.973 | 0.304 | Normal | 81.534% |
| | 0.696 | 0.027 | Anomaly | 81.534% |

**Table1:**Result of the experiment after training SDN Controller

These results were calculated after training the OpenDaylight Controller. Network data packets can be collected for analysis in many ways, including [4]:

1. **Method 1:**Controller polling data from SDN enabled switch in an equal timeinterval.
2. **Method 2:** Writing a separate procedure which will collect the network packet in the required format and thenWriting a separate procedure which will collect the network packet in the required format. Then the collected data is passed through the exposed API.

A business rule can be defined for anomalous packets found, depending on the outcome. These business rules can be either blocking the source IP address, restricting its future communication, or redirecting this traffic through a separate independent channel which will be processed by third-party tools fordiagnosis.

## V. Conclusion

This paper provides an approach to intrusion detection and prevention in software-defined networking (SDN). We emphasized SDN technology as a platform, using machine learning approaches to detect vulnerabilities and monitor networks. The significant advantages of machine learning algorithms are feature learning and parameter optimization. The solution gives a generic answer which can work with any training dataset to classify the packets. After the controller training and testing, it is ready to predict the behavior of the packet. There is a vast prospect for research. Initially, it is intended towards the OpenDaylight SDN controller only. OSGI bundles developed are compatible with OpenDaylight Controller. Future potential includes building a controller agnostic solution, which can work for any controller. The study and implementation of more intelligent machine learning algorithms can be added for intrusion detection accuracy. Further, a detailed study of different types of DDoS attacks detection can be done along with the supervised and unsupervised machine learning algorithm execution.

## References


[1] Ha, T., Kim, S., An, N *et al*., "Suspicious traffic sampling for intrusion detection in software-defined networks", *Computer Networks*, 2016, 109, pp. 172– 182

[2] I. Ahmad, S. Namal, M. Ylianttila and A. Gurtov, "Security in Software Defined Networks: A Survey" , 2015 IEEE Communications Surveys & Tutorials, vol. 17, no. 4, pp. 2317-2346, Fourthquarter

[3] M.M. Oo, S. Kamolphiwong , T. Kamolphiwong ,"The Design of SDN based Detection for Distributed Denial of Service (DDoS) attack", 2017 21st International Computer Science and Engineering Conference.

[4] Abhilash G and Gupta G, "Intrusion Detection and Prevention in Software Defined Networking", 2018 IEEE International Conference on ANTS

[5] http://www.unb.ca/cic/datasets/nsl.html

[6] N.Z. Bawany, J.A. Shamsi *et al*., "DDoS Attack Detection and Mitigation Using SDN: Methods, Practices, and Solutions", 2 February 2017, Review Article Springer

[7] Y. Lingfeng, Z. Hui , "DDoS Attack Identification and Defense using SDN based on Machine Learning Method",2018 15th International Symposium on Pervasive Systems, Algorithm and Networks.

[8] K.S. Sahoo, A. Iqbal , "A Machine Learning Approach for Predicting DDoS Traffic in Software Defined Networks",2018 International Conference on Information Technology.

[9] S.S Mohammed *et al.*, "A New Machine Learning-based Collaborative DDoS Mitigation Mechanism in Software-Defined Network", 2018 14th International Conference on WiMob

[10] Zhiyuan Li *et al.*, "Detecting Saturation Attacks in Software-Defined Networks", 2018 IEEE International Conference on Intelligence and Security Informatics (ISI)

[11] V. Deepa , K.M. Sudar*et al.*, "Detection of DDoS Attack on SDN Control plane using Hybrid Machine Learning Techniques" ,2018 IEEE Conference on Smart Systems and Inventive Technology
[12] Karan B.V. *et al*., "Detection of DDoS Attacks in Software Defined Networks",2018 IEEE Conference on Computational Systems and Information Technology for Sustainable Solutions
[13] S.M. Mousavi, M. St-Hilaire ,"Early Detection of DDoS Attacks against SDN Controllers",2015 IEEE Conference on Computing, Networking and Communications, Communications and Info. Security Symposium
[14] MajdLatah,L. Toker,"Towards an efficient anomaly-based intrusion detection for software-defined networks", *IET Networks*, 2018, Vol. 7 Iss. 6, pp. 453-459
[15] P. Amaral *et al.*,"Machine Learning in Software Defined Networks: Data Collection and Traffic Classification" 2016 IEEE 24th International Conf. on Network Protocols Workshop on Machine Learning in Computer Networks
[16] K. K. Karmakar*et al.*,"On the Design and Implementation of a Security Architecture for End to End Services in Software Defined Networks", 2016 IEEE 41st Conference on Local Computer Networks
[17] A. S. Prasad *et al.*,"On the Security of Software-Defined Networks", 2015 IEEE Fourth European Workshop on Software Defined Networks
[18] K. Kaur, J. Singh,"Programmable Firewall Using Software Defined Networking" 2015 IEEE 2nd International Conference on Computing for Sustainable Global Development
[19] D. Satasiya *et al.*,"Analysis of Software Defined Network Firewall (SDF)", 2016 IEEE WiSPNET conference
[20] https://haifengl.github.io/smile/

Sunidhi, et. al. "Survey on Software Defined Networking based Network Intrusion Detection and Prevention System." *IOSR Journal of Computer Engineering (IOSR-JCE)*, 22(5), 2020, pp. 49-54.