A survey on security measures on SDN

Prateekshya Priyadarshini
Department of Computer Science and Engineering
Indian Institute of Technology
Guwahati, Assam, India
ppriyadarshini@iitg.ac.in

Vijay Purohit
Department of Computer Science and Engineering
Indian Institute of Technology
Guwahati, Assam, India
vijay.purohit@iitg.ac.in

***Abstract*—Software-Defined Networks (SDN) is a promising area in Computer Networks in recent years as it introduces programmability to the Network leading to efficient Network Management. In SDN, Network Intelligence is centralized by separating data plane and control plane. Hence, controllers are highly vulnerable to various kinds of attacks which can destroy the entire network. In recent years, significant research is being done on the security measures to be taken for the possible attacks. In this paper, we present a survey on the types of measures being taken for safety of SDN, specifically the control plane. We present the background of SDN, typical attacks possible on SDN, various security measures taken for the safety of SDN, conclusion and future directions.**

***Index Terms*—SDN, Security, Network Programmability, Attacks, Measures, Blockchain, Cloud, IoT, ML.**

I. Introduction

Software-Defined Networks (SDN) is a network technology that came from the idea of programmable networks. SDN separates the data plane and control plane logically and provides programmability, reliability and flexibility to the network in order to improve network performance and monitoring. It centralize network intelligence in one network that makes it more like cloud computing than traditional network management. In [1] we can see that SDN mostly has three layers i.e. control layer, infrastructure or data layer and application layer.

Control layer consists of the control logic that manages the forwarding table for each request of the switch based on the header of the packet. A network can have multiple controllers simultaneously to control a group of switches each. But, it might get more hectic to handle the control plane. So, one controller is chosen to be the main controller and the others would be the backup controllers. In case of the main controller failure, we use the other controllers which maintain the property of fault tolerance as well. SDN control plane can follow a centralized, hierarchical, or decentralized design. In case of a completely distributed control system where we have multiple controllers working simultaneously. There will be a central controller managing all the controllers. In this case, all the controllers won't be working as backup controllers.

Infrastructure layer consists of the switches which communicate with the controllers via a communication medium such as OpenFlow. It consists of three important parts i.e. flow table, secure channel and OpenFlow protocol. A flow table is used to get information about the packets and also do forwarding. A secure channel means a TLS or SSL channel between the switch and the controller. The protocol helps in the management of communication.

Application layer is the uppermost layer that deals with the end-user. It comprises of one or more applications, each of which can consume exclusive information about network provided by one or more SDN controllers. SDN Applications are programs that programmatically and directly communicate their network requirements to the SDN controller via Northbound Interfaces.

Northbound interface and *Southbound interface* are the interfaces used for communication and other purposes. The northbound interface is the interface between the SDN applications and SDN controllers and would be executed generally through REST APIs of SDN controllers. Whereas the Southbound interface enables the communication between SDN controllers and the nodes in the Infrastructure layer of networks elements and would be generalized through southbound protocols. In other words, the Northbound interface (or Controller Application Interaction) connects the controllers with the upper layer and provides abstract network views, enables direct expression of network behaviour while the Southbound interface (or Control to Data-Plane Interface) connects it with the lower layer and provides programmatic control of all forwarding operations, statistics reporting and event notifications. Unlike the Southbound interface, which has well-defined protocols like ForCES and OF (OpenFlow), the Northbound interface doesn't have any defined standard since that will be decided according to the logic in SDN Application.

Hybrid SDN: Deployment of SDN is costly and time-consuming. As per [2] the solution is to deploy a limited number of SDN-enabled devices. It will create a network that is a combination of SDN and general network devices. Eventually, we keep replacing the general network devices with SDN enabled devices. This type of network is called Hybrid SDN. Hybrid SDN can be beneficial in multiple ways i.e. deployment of the whole SDN is costly, hybrid SDN can still have some of the benefits of SDN paradigm without a full SDN deployment, easy management of scenarios where two SDNs are connected via a general network device, traditional routing protocols are also effective for some tasks, easy evaluation etc.

SDN most commonly southbound interface is associated with the *OpenFlow protocol* which is defined for Ethernet-

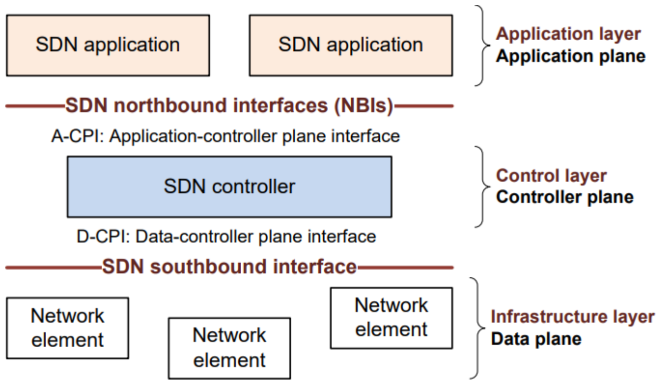

Fig. 1. Overview of Software-Defined Networking Architecture.

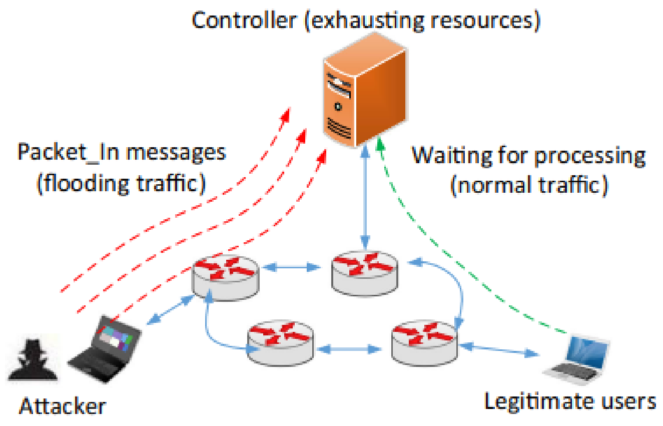

Fig. 2. Dos/DDoS attack on Controller.

based networks. It is a widely accepted standard protocol defined by the ONF organization. It intellectually accomplish forwarding actions as required. An OpenFlow switch carry multiple flow tables, within which a huge amount of flow entries is contained.

II. Related Work

Recent surveys on SDN security tell a lot about the types of attacks that are frequent. Many authors have classified them in different types and evaluated the performance of network.

The authors in [3] discuss eleven distinct works that are based on SDN security. They talk about the 4D Project developed from the original work "A Clean Slate 4D Approach to Network Control and Management", OpenFlow Security, Kandoo, AVANT-GUARD, FRESCO. They also talk about Traffic Anomaly Detection. OpenFlow security was their main focus.

Authors [4] have briefly explained the Security Challenges in SDN. They have classified them as hardware-based challenges and protocol-based challenges. Controllers might be the primary target for the attackers or the OpenFlow protocol.

Authors in [5] have mentioned security issues and attack objects at different layers of SDN i.e. Application Layer (Malicious Application, Authorized Authentication, Flow Rules Conflict), Northbound Interface (standardization problem), Control Layer (DoS/DDoS attacks, Hijacked/Rogue Controller, Poor Controller Deployment), Southbound Interface (OpenFlow protocol), Data Layer (Authorized Authentication, Legality and Consistency of flow rules, DoS/DDoS attacks, Side-Channel Attacks), Southbound interface (Commnuication security). Further, they have mentioned the causes of typical security problems in SDN and also explained the existing defence technology for all these layers.

The authors at [6] and [7] also mentioned various security threats to SDN, their causes and discussed typical countermeasures that can be taken against the threats.

In [2] author give the general overview of the security related to the hybrid SDN networks.

We shall talk about the recent trends in security measures of SDN that use various recent technologies like Machine Learning, Blockchain and Firewall based techniques. We shall also discuss the various measures taken to detect and prevent DDoS attacks on SDN.

III. Frequent Attacks

A. Types of Attacks

In [8] there is mentioning of various types of attacks. For example:

Application Layer: flow rules insertion/conflict, malicious code, authorized authentication;

Control Layer: DoS/DDoS attacks, attacks from applications, man-in-the-middle attacks, hijacked controller;

Data Layer: authorized authentication, consistency/legality of flow rules, side channel attacks. Some of the attacks are briefly explained in [9] and their division according to the layers of SDN. The attacks can be *Control Plane* specific attacks which are mostly against the SDN control and the application layer, *Control Channel* specific attacks which includes attacks against the protocols (e.g. OpenFlow) and *Data Plane* specific attacks which are done on network devices.

B. Types of DDoS Attacks

In a network, the availability of a network device is very important. Hence, *DoS and DDoS attacks* are the most important aspects to be considered. The main attack mode is to impose excessive load on the controller, which reduces the availability of SDN resources to legitimate users.

There are various types of DDoS Attacks.

1) Control Layer DDoS Attacks: In this attack, the attackers' main target is the control plane. There are many methods of launching this attack. Some of them are attacking the controller, attacking the northbound API, attacking the southbound API (OpenFlow). Many conflicting flow rules from many devices will create a lot of unnecessary traffic and the controllers will be busy servicing unnecessary requests occupying a high bandwidth which may cause DDoS attacks on the control plane. Countermeasure would be to analyze the characteristics of traffic flows stored in the OpenFlow Switches or use of framework FloodGuard.

2) Infrastructure Layer DDoS Attacks: Attackers may target two things i.e. switches or southbound interface. The packets with only header information require memory in the nodes. So, the attackers can release many such packets to flood the node memory. Node's memory is a bottleneck due to the cost factor. In order to save the fake packets, the required packets of the actual data might not get stored. Attacking the southbound interface i.e. the protocols like OpenFlow is a common practice of the attackers. It usually breaks the entire flow of the network. Countermeasures would be to use the tool FlowVisor which acts as an agent between switches and the controller and rewrites the rules. Other measure could be use of intrusion detection system or Virtual Source Address Validation Edge (VAVE)

3) Packet Flooding: This is a type of attack where the attackers generated a huge number of network flows and send them to the controller. As a result, the controller resources will be consumed by false data packets and the controller could move to an unpredictable state.

C. Control Layer Specific Attacks

It is the main part of SDN and weakest link in security chain. An attacker who comprises the controller will be able to control the entire network.

1) Control Message Manipulation: Common practices of the hackers through this mode are Switch Table Flooding, Switch Identification Spoofing and Malformed Control Message Attacks. The attackers try to manipulate control messages and when the controller tries to parse the message, it starts working unexpectedly.

2) Hijacked Rogue Controller: Unauthorized applications can use upper-layer applications to hijack the controller, thus causing legitimate request of users to be rejected [5].

3) Poor Controller Deployment: A multi-controller deployment scheme always reassigns failed controller loads to another controller and when the controller deploys improperly, it may easily burden the controller load beyond its capacity, and cause a failure.

4) Man in the middle Attack: The main philosophy of it to add an agent node between the source and the destination node, and is used to intercept communication data and modify them without being detected. Specific attack methods include session hijacking, DNS spoofing port mirroring, and so on [7]. Countermeasures to it would be to create a secure channel between the controller and switches (TLS) and using validation tool like FlowChecker.

D. Application Layer Specific Attacks

It consists of various types of applications. OpenFlow applications , security service applications or other third-party applications can formulate the flow rules.

1) Authorized Authentication: There is lack of enough trust assessment and management mechanism to verify the security of application. The defects in the application's role authentication and access control and vulnerabilities may be exploited by attackers to generate malicious flow rules.

2) Flow Rules Conflict: Due to large number of third-party open-source applications in SDN, produces higher degree of coupling which may appear to compete, cover and conflict.

3) Malicious application: A malicious application will bring unpredictable damage to SDN. It consists of viruses, Trojans, worms which can attack SDN through software agents in relatively less efforts.

E. Data Layer Specific Attacks

It consists of switches and other basic devices, mainly responsible for data processing, forwarding, discarding and status collecting.

1) Authorized Authentication: It lacks the effective authentication process between the basic devices and the controller. A malicious switch may manipulate or discard legal data packets. The switch can also be controlled by a malicious controller if established without authentication. Countermeasures would be to use of tool AuthFlow which uses a RADIUS server to authenticate the host identity.

2) Legality and consistency of flow rules: Legality of flow rules refers to malicious or incorrect flow rule injection. During the generation process, release process, and update process can cause conflicts flow, inconsistent rules and non-synchronized flow rules between switches. Countermeasures would be to use of the framework NeSMA and Flowvisor.

3) Side channel attacks: An attacker can infer network-related flow table information by testing the execution time of the specific type of data packet by using side channel attacks. It can trigger further attacks indirectly.

F. Security Issues in interfaces

Southbound interface attack is mainly caused by the leak of protocols like OpenFlow. It faces eavesdropping, controller spoofing, data leakage etc. Northbound interface faces the standardization problem as the relationship between the control layer and application layer is more fragile. There are no uniform methods of authorization and authentication due to diversity and constant update of SDN applications.

IV. Security Measures Taken

A. Against DDoS Attacks

The very first thought that comes when we talk about SDN security is securing the systems from DDoS attacks. Various techniques are proposed by authors for the same.

In [10], the authors propose an *optimized weight voting based ensemble model for detection and mitigation* of such attacks in an SDN environment. It implements six base classifiers i.e. two SVMs (Support Vector Machine), two random forests, two gradient booster machines that are differentiated by hyper-parameter values. SVM is used because it has high generalization ability and strong resistance to over-fitting. A hyper-plane is a collection of points $\overrightarrow{x}$ that satisfy the equation

$$\overrightarrow{w}\overrightarrow{x} + b = 0 \tag{1}$$

where $\overrightarrow{w}$ and b are the weight vector and bias respectively and $\overrightarrow{x}$ is the point. By establishing hyperplanes, SVM classifies

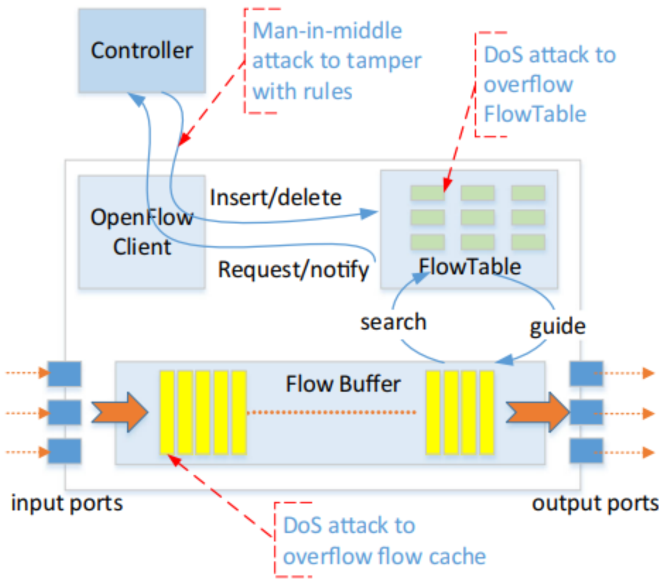
Controller
Man-in-middle attack to tamper with rules
DoS attack to overflow FlowTable
OpenFlow Client
Insert/delete
Request/notify
FlowTable
search
guide
Flow Buffer
input ports
output ports
DoS attack to overflow flow cache

Fig. 3. Security Threats of OpenFlow Switches.

data into different classes. The hyperplane with the greatest distance between the nearest points in the two groups is considered to be the best out of an excessive extent of hyperplanes. Random forests are also heavily used but they lack accuracy because of their low bias and high variance. They integrate bagging with random variable selection. Gradient boosting is a technique that combines the results of a large number of simple predictors to create an efficient committee with better performance than single members. It is used for both regression and classification task.
The authors propose an approach that has some phases i.e. learning phase, detection and mitigation phase. In the learning phase, they used DDoS Evaluation Dataset for training purposes. The optimal set of weights are identified by a novel hybrid metaheuristic optimization algorithm "BHO" whose dynamic fitness function is

$$1 - Accuracy = \frac{\alpha FN + \beta FP}{TP + TN + \alpha FN + \beta FP} \tag{2}$$

where TP is True Positive, TN is True Negative, FP is False Positive and FP is False Negative. α and β are to assign penalty weights to FN and FP.
The algorithm is a five-step algorithm. BHO starts by taking population size, the number of groups, problem dimension, lower and upper bounds of solutions, maximum iterations and other parameters as its input. In the end, the population will be passed through the worst agent handling module in which N_w worst solutions are selected and replaced with randomly generated solutions using the greedy selection strategy. This step helps to escape the local optima trap. In the detection and mitigation phase, the results from each base learner are obtained. A weighted majority vote of the results obtained from each base learner is used as a final prediction to classify a new flow as either malicious or normal. The final prediction is

$$\begin{aligned} majority[W_1.SVM_1, W_2.SVM_2, W_3.RF_1, W_4.RF_2, \\ W_5.GBM_1, W_6.GBM_2] \end{aligned} \tag{3}$$

In [11] authors propose an entropy-based technique followed by Machine Learning. Detection is done using an initial module based on information entropy and then Machine Learning is used with a stacked sparse autoencoder. The information entropy associated with a random variable increase as the dispersion of its distribution increases. Mathematically, entropy is

$$H(x) = -\sum_{i=1}^{N} (\frac{n_i}{S}) \log (\frac{n_i}{S}) \tag{4}$$

where $S = \sum_{i=1}^{N} n_i$ is the total number of samples in x and n_i/S represents the probability of outcome x_i. Auto Encoder uses the output of a neural network to reconstruct the input data. It adopts the back-propagation gradient descent algorithm to correct the model parameters.

In [12] a mitigation approach using *Deep Neural Networks* is suggested. Usually, rate-limiting logic is implemented as a defence mechanism to protect a system from excessive service.

There are several techniques for measuring and limiting rates i.e. *Token Bucket* (maintains a rolling and accumulating budget of usage as a balance of tokens and decreases the count when a service request is made. If there are no tokens in the bucket that means the limit is reached and no more requests can be serviced.)

Leaky Bucket (similar to the token bucket but the rate is limited by the amount that can leak out of the bucket. It assumes that the system has a limited capacity and any request beyond that is considered as a leak and is discarded.)

Fixed Window (limits the number of requests per period of time) and *Sliding Window* (has all the benefits of the fixed window but the sliding window of time smooths out the bursts).

Deep Neural Networks are formed by stacking layers of neural networks along with the depth and width of smaller architectures. They focused on minimizing the loss function by dynamically adjusting the allowable number of new flow requests in each host where they use a sliding window-based rate-limiting mechanism. A weight is used to specify the allowable number of new flow requests that a host can send to its edge switch during a cycle. The weights are adjusted at the end of each cycle using two discounts (one is computed through a training model which predicts the number of received PacketIn messages at a controller then compares it with the available queue capacity using an MSE-based loss function and the other is obtained through evaluation of the disparity between the currently available memory of the flow-table of a switch and the number of PacketIn messages to the controller for processing). When the weights are adjusted, the controller declares a new allowable number of flow requests to the host using the OpenFlow messages. Then the controller checks whether the changes after the new policies remain within a particular range or not. If it is within the range then the host is considered to be normal otherwise it should be blocked. The controller blocks the host by installing a new default rule within the flow table by which all new flow requests which are not matched will immediately be dropped.

B. Machine Learning Techniques

After discussing the detection and mitigation techniques for DDoS attacks, which also included ML and Deep Learning techniques, we shall discuss more ML techniques and how it is used in SDN security.

In [13] the authors have proposed an attack pattern prediction using ML techniques. ML algorithms are trained using traditional network threat intelligence data to identify potentially malicious linkages and probable attack targets. Based on *Conventional Network Data (DT)*, *Bayesian Network (BayesNet)*, *Naive-Bayes*, *C4.5* and *Decision Table* are used to predict the largest host that will be attacked. Bayesian Network computes probabilistic interactions between various interest variables. The random variable and its directional edge for each node are reflected in the resultant graph. Native-Bayes utilizes Bayesian Theory which employs the training samples to predict the sort of unidentified samples based on the previous results. Native Bayes divides the instruction set as a choice vector and attribute vector. C4.5 utilizes the decision making tree in common. It can handle extremely noisy data. In Decision Table, the primary learning phases were:

1) It selects an attribute modelled on which to evaluate a logical check.
2) A part of training data for the child node is extracted from the experimental results.
3) All the child nodes run the methods iteratively.
4) A node is represented by a leaf depending on specific concluding principles.

The experiments show that the overall accuracy is often more than 90%. In this study SDN and NFV (Network Function Virtualization) are combined, and the proposed network was named Software Defined Network Function Virtualization (SDNFV). The proposed stateful firewall was configured to effectively detect and prevent Denial of Service(DoS) Attacks. The core features of the proposed methodology were

1) The use of traditional data to train ML-based applications
2) The use of the trained design to detect possibly susceptible hosts and establish the safety rules in the VNF (Virtualized Network Functions) module according to the predictive performance of the Machine Learning algorithm.

Basically, the design is trained using traditional data and then is used to determine susceptible hosts. Here,

$$Accuracy = \frac{p}{t} * 100 \tag{5}$$

where p is the number of attacks correctly predicted and t is the number of attacks in total. The choice of dataset affects the accuracy of the prediction. The greater the data variability, the greater the likelihood of incorrect prediction is. Using the prediction results of machine learning models, it is possible to define the security flow rules for stateful firewall VNF to avoid unauthorized clients from entering the network.

The authors in [14] have given an evaluation of the ML techniques discussed above (e.g. Naive Bayes, SVM etc) for SDN security. They have evaluated techniques like SVM, Naive-Bayes, Decision Tree, Logistic Regression. The experiment was a simulation-based experiment that was run on a Core i7 machine with a 1.80 GHz processor, 16 GB of RAM and 512 GB of SSD on ubuntu 18.04. A remote controller with a tree-based topology having a depth of two is used to create the testbed with Mininet. POX controller and Open vSwitches with OpenFlow are used. According to the results, SVM performs better than other algorithms with an accuracy of 97.5% and a precision of 97% with a very low error rate of 2.5%. All other algorithms have an accuracy of 96% except Logistic Regression.

C. Firewall Techniques

According to [15] Firewall is a simple security mechanism that is placed between different trust zones and controls the traffic. The goal is to make all the networks pass through the firewall, to allow only authorized traffic based on security policy to pass, to maintain logs since it is immune to penetration. The following techniques are proposed using Firewall.

1) Stage focus on the control plane to manage the entire network is more extensive and will bring a bunch of new features to the table. These changes streamline and networking operations must accelerate.
2) An important consideration is to balance the load across the number of firewalls.
3) Also tries to reduce overhead on SDN firewall
4) Provide good for flexible and easy configuration of firewall object on users need and demand.

The OpenFlow protocol implements an intermediate firewall. Every host has an individual firewall. The hosts get connected to the network with a firewall object and then OpenFlow transfers control to the host and allows traffic passing. Firewall module implements firewall like functionality for POX. This helps in securing SDN from attacks.

Authors in [13] discuss the methods of predicting attack pattern by exploiting stateful firewall as Virtual Network Function via ML as discussed in ML techniques.

D. Blockchain Techniques

Authors in [16] introduce B-DAC, a blockchain-based framework for decentralized authentication and fine-grained access control for the northbound interface to assist administrators in managing and protecting critical resources. Blockchain has the power of cryptography such as hash, asymmetric cryptography, digital signatures to ensure that data are kept securely and confidential. It has a public ledger that contains all records of digital transactions transmitted among the involved parties. This ledger is replicated and stored in the whole peer-to-peer nodes. Thus, it makes transaction records accessible even if several nodes are unavailable or disrupted. The process of consensus keeps the database of the blockchain under the control of multiple nodes. A new transaction data must be validated by all nodes before being added to the blockchain. So, it is infeasible for the hackers to manipulate records of the ledger since they must have controls over multiple nodes to overcome the consensus scheme. Hyperledger Fabric is an open-source implementation of permissioned enterprise-grade blockchain for running smart contracts, named chain code. It contains three groups of nodes which are peer, ordered and client-owned by different organizations. The task of identifying all nodes is carried out by a Membership Service Provider (MSP).

Current issues in SDN are deficiencies of authentication-authorizing-accounting, lack of tamper-proof and integrity & Single point of failure (SPoF), malicious flow rules injection, exhausting controller resources, information disclosure. According to B-DAC, a participant must have a digital identity that is stored in a wallet to connect to a Hyperledger Fabric. Given a key pair of the private key and the public key, it uses a random number to improve the security of the signature. A random number is created before encryption.

Here, the main goal is to protect the communication between the controller and the OpenFlow protocol. To enable trust establishment, the B-DAC framework evaluates the trustworthiness of applications in a specific network, based on their behaviours. Here, the authors introduce an additional field called *trust index* along with the other necessary information. The value of the trust index decreases with over-privileged attempts or conflicts in flow rules installation from OpenFlow applications. To support the authentication, only registered applications are allowed to interact with the system, for which a token is used. These tokens are created and managed by the Token Management Module. In addition, there is a Permission Module designed to check whether the action of the application is over-privileged and needs to be noticed. Logging is the basic function of accounting where log entries are obtained and saved in a specific format. The design has a Log Module that manages this job. All types of actions that tend to change the value of assets in the distributed ledger (blockchain) are performed by transactions. To enhance the flexibility and independence of the system, the AAA scheme is implemented outside the controller.

B-DAC provides RESTful APIs for controllers as well as OpenFlow applications to interact with it. AAA scheme includes Authentication to B-DAC's REST API, Authentication to OpenFlow application, Authorization and Accounting. After getting a request, multiple fields are extracted to reform the flow rule. These flow rules can be checked against the predefined conflict policy by verifying the validity of the flow rule (all the parameters and their values are extracted from the flow rule and then it is checked whether they are supported as well as their value does not violate restrictions like range, maximum value etc) followed by detecting and classifying flow rule conflicts (by comparing a target flow rule to a saved conflict-free flow rules database to figure out whether any conflict exists). The workflow of B-DAC goes into nine steps starting from request sending followed by authentication (validation), token assignment, request dispatch, permission parsing, conflict detection and request servicing. The security characteristics like Immutability, Decentralization, Authentication, Authorization, Accounting, Flow rules conflicts prevention, Fine-grained control, Trust Level management are ensured by the B-DAC system. This system enhances the security of the northbound interface in SDN by managing OpenFlow Applications. Overall this system aims to secure the interaction between SDN controllers and network applications i.e. the northbound interface by utilizing the underlying characteristics of the blockchain.

V. Conclusion and Future Work

In this paper, we concisely reviewed the architecture of SDN. We explained the types of attacks that are very frequent on SDN as well as various types of security issues. We discussed the countermeasures to prevent those attacks. Security measures using ML techniques, blockchain and firewall etc have been generally used in SDN. Some specific systems have also been implemented to detect and prevent specific types of attacks. As one of the most promising technologies, existing solutions are not sufficient to completely alleviate security issues in SDN. Future work could include exploration of Machine Learning, blockchain techniques and implementation of the ideas provided by the authors. Also, the security aspects of hybrid SDN networks has vast potential for future research as only a very limited number of studies have addressed it.

References

[1] S. Rowshanrad, S. Namvarasl, V. Abdi, M. Hajizadeh, and M. Keshtgary, "A survey on SDN, the future of networking," *Journal of Advanced Computer Science & Technology*, vol. 3, pp. 232–248, Nov. 2014. DOI: 10.14419/jacst.v3i2.3754.

[2] R. Amin, M. Reisslein, and N. Shah, "Hybrid sdn networks: A survey of existing approaches," *IEEE Communications Surveys Tutorials*, vol. 20, no. 4, pp. 3259–3306, 2018. DOI: 10.1109/COMST.2018.2837161.

[3] M. Coughlin, *A Survey of SDN Security Research, University of Colorado Boulder*, 2014.

[4] H. Zhang, Z. Cai, Q. Liu, Q. Xiao, Y. Li, and C. Cheang, "A survey on security-aware measurement in sdn," *Security and Communication Networks*, vol. 2018, Apr. 2018. DOI: 10.1155/2018/2459154.

[5] Y. Liu, B. Zhao, P. Zhao, P. Fan, and H. Liu, "A survey: Typical security issues of software-defined networking," *China Communications*, vol. 16, no. 7, pp. 13–31, 2019. DOI: 10.23919/JCC.2019.07.002.

[6] S. Scott-Hayward, G. O'Callaghan, and S. Sezer, "Sdn security: A survey," in *2013 IEEE SDN for Future Networks and Services (SDN4FNS)*, 2013, pp. 1–7. DOI: 10.1109/SDN4FNS.2013.6702553.

[7] Z. Shu, J. Wan, D. Li, J. Lin, A. Vasilakos, and M. Imran, “Security in software-defined networking: Threats and countermeasures,” *Mobile Networks and Applications*, vol. 21, Oct. 2016. DOI: 10.1007/s11036-016-0676-x.

[8] M. Iqbal, F. Iqbal, F. Mohsin, M. Rizwan, and F. Ahamd, “Security issues in software defined networking (sdn): Risks, challenges and potential solutions,” Jan. 2019.

[9] T. Jose and J. Kurian, “Article: Survey on sdn security mechanisms,” *International Journal of Computer Applications*, vol. 132, no. 14, pp. 32–35, Dec. 2015, Published by Foundation of Computer Science (FCS), NY, USA.

[10] A. Maheshwari, B. Mehraj, M. Khan, and M. Idrisi, “An optimized weighted voting based ensemble model for ddos attack detection and mitigation in sdn environment,” *Microprocessors and Microsystems*, vol. 89, p. 104 412, Mar. 2022. DOI: 10.1016/j.micpro.2021.104412.

[11] Z. Long and W. Jinsong, “A hybrid method of entropy and ssae-svm based ddos detection and mitigation mechanism in sdn,” *Computer and Security*, 2022.

[12] A. E. Kamel, H. Eltaief, and H. Youssef, “On-the-fly (d)dos attack mitigation in sdn using deep neural network-based rate limiting,” *Computer Communications*, pp. 153–169, 2022.

[13] S. Prabakaran, R. Ramar, I. Hussain, *et al.*, “Predicting attack pattern via machine learning by exploiting stateful firewall as virtual network function in an sdn network,” *Sensors*, vol. 22, no. 3, 2022, ISSN: 1424-8220. DOI: 10.3390/s22030709. [Online]. Available: https://www.mdpi.com/1424-8220/22/3/709.

[14] A. Ahmad, E. Harjula, M. Ylianttila, and I. Ahmad, “Evaluation of machine learning techniques for security in sdn,” in *2020 IEEE Globecom Workshops (GC Wkshps*, 2020, pp. 1–6. DOI: 10.1109/GCWkshps50303.2020.9367477.

[15] D. Satasiya, R. Raviya, and H. Kumar, “Enhanced sdn security using firewall in a distributed scenario,” in *2016 International Conference on Advanced Communication Control and Computing Technologies (ICACCCT)*, 2016, pp. 588–592. DOI: 10.1109/ICACCCT.2016.7831708.

[16] D. Phan, H. Hoang, D. Hien, A. Nguyen, and V.-H. Pham, *B-dac: A decentralized access control framework on northbound interface for securing sdn using blockchain*, Nov. 2021.