Towards Securing OpenFlow Controllers for SDNs using ARMA Models

WAEL HOSNY FOUAD ALY
College of Engineering and Technology
American University of the Middle East
KUWAIT

HASSAN KANJ
College of Engineering and Technology
American University of the Middle East
KUWAIT

NOUR MOSTAFA
College of Engineering and Technology
American University of the Middle East
KUWAIT

SAMER ALABED
College of Engineering and Technology
American University of the Middle East
KUWAIT

Abstract: Control layers are moved away from the forwarding switching layers in Software Defined Networks. SDNs allow more programmability and flexibility to the controllers. OpenFlow is a protocol that gives access to the forwarding plane of a network switch or router over the SDN network. OpenFlow uses a centralized control of network switches and routers in SDN environment. Security is of a major importance for SDN deployment. Transport Layer Security (TLS) is be used to implement security for OpenFlow. This paper proposes a new technique to improve the security of the OpenFlow controller through modifying the TLS implementation. The proposed model is referred to as *Secured Feedback model using Autoregressive Moving Average (ARMA) for SDN* networks ($\text{SFBARMA}_{\text{SDN}}$). $\text{SFBARMA}_{\text{SDN}}$ depends on computing the feedback for incoming packets based on ARMA models. Filtering techniques based on ARMA techniques are used to filter the packets and detect malicious packets to be dropped. $\text{SFBARMA}_{\text{SDN}}$ is compared to two reference models. One reference model is based on Bayesian and the other reference model is the standard OpenFlow. Results are very promising. $\text{SFBARMA}_{\text{SDN}}$ has outperformed both the secured standard using Bayesian network for SDN (SSBN_{SDN}) and the standard OpenFlow in different scenarios by an average improvement of 7% and 80% respectively. The processing time overhead for the $\text{SFBARMA}_{\text{SDN}}$ increases by only a percentage of 3% and 5% when compared to the SSBN_{SDN} and the standard OpenFlow respectively.

Key–Words: Software defined network; network security; OpenFlow controller; ARMA; empirical technique; Bayesian network

Received: May 15, 2021. Revised: April 11, 2022. Accepted: May 10, 2022. Published: June 25, 2022.

1 Introduction

Internet softwarization is the future of the Internet which has an impact on the network innovation towards network improvement and enhancement. Software-Defined Networking (SDN) is known for the near to optimal solutions regarding network design and implementation. SDN is considered as an appropriate environment to become immune to malicious attacks. SDN separates the control functionalities and the forwarding functionalities. Controller modules are responsible for the control decision functionalities, while the switch modules are responsible for the forwarding functionalities. A typical SDN architecture is composed of north-bound interfaces, management layer, control layer, data layer, east-west Bound Interfaces. Moreover, the management layer consists of a set of network applications that manage the control logic of SDN. The control layer contains a set of controllers that forward different rules and policies to the data layer through the southbound interface. The data layer represents the forwarding network elements on the network [1].

SDN architecture has different types of interfaces such as north-bound interface, east-west bound, and south-bound interfaces. The north-bound interfaces is considered as the interface that controls the connection between the controller module and the application. It is responsible for the communication between control layer and the management layer. Moreover,

the east-west bound interface allows communication between multiple controllers through message passing. South-bound interfaces facilitate allow the interaction between the control layer and the data layer [2], [3].

OpenFlow is one of the popular protocols that pushes policies to the forwarding plane. Security of SDN has been under investigation since the OpenFlow is considered as the first standard communications interface designed by the Open Network Foundation (ONF). OpenFlow security relies on the optional implementation of the transport layer security (TLS) that is considered to be vulnerable to malicious attacks. Increasing the SDN reliability has been studied in various research work such as [1, 4–8].

This paper proposes a novel model to add secure capabilities to SDN networks using feedback control techniques based on ARMA models. The proposed approach is based on feedback control techniques using autoregressive moving average (ARMA) models. The proposed model is referred to as *Secured Feedback model using Autoregressive Moving Average (ARMA) for SDN networks* ($\text{SFBARMA}_{\text{SDN}}$). Extensive comparisons were conducted to compare the proposed model $\text{SFBARMA}_{\text{SDN}}$ to two reference models. The first reference model is based on Bayesian networks to provide security capabilities and is referred to as secured standard using Bayesian network for SDN (SSBN_{SDN}). The second reference model is referred to as the standard OpenFlow [9].

The reason behind using the ARMA model in the proposed model is that in classical engineering environments, the relationships between outputs and inputs are studied through physical laws that are referred to as the first-principles techniques. The main barrier for using first-principle modeling in the computing system domain is that some unrealistic assumptions are made. For that reason we use empirical approaches for developing transfer functions through autoregressive moving average (ARMA) approach [10]. The proposed feedback control system model relies on having a tuning parameter that is easy to control. This tuning parameter has an influence on a controlled output parameter. The system desires to ultimately achieve a certain target value for the controlled output parameter. The controlled output parameter is a parameter that is needed to be controlled but cannot be adjusted directly and hence the tuning parameter comes into place. The tuning parameter could be directly tuned and has an impact on the controlled output parameter. In this work the tuning parameter used is the current value of the security level of the system. The controlled output parameter is the enhanced security level of the system. Feedback controllers use the ratio between outputs and inputs. This is defined mathematically through transfer functions [11].

The main reference model used in this work is based on Bayesian network (BNs). BNs are considered as probabilistic models that use graphical representations for knowledge. A BN is a domain that deals with uncertain domains. The main strength of the BN is that it applies probability theory to control model complexities. BN nodes are represented as random variables while edges are represented as conditional probability for the appropriate random variables. A major weakness of the BNs is the need to fully specify the probability distributions for the network, and this number is high [12].

The paper is organized as follows Section 2 has the related work. Section 3 has a description of the SSBN_{SDN} reference model. Section 4 has the $\text{SFBARMA}_{\text{SDN}}$ proposed model. Section 5 has the experimental results. Finally, section 6 has the conclusion and the future work.

2 RELATED WORK

This section has a brief discussion about the OpenFlow protocol. OpenFlow protocol is considered as the de facto SDN protocol for SDN. OpenFlow is implemented at both the SDN interface and the SDN controller. It is responsible for forwarding packets in the forwarding plane for all SDN network elements [13]. OpenFlow is categorized into three categories (1) switch category, (2) channel category, and (3) controller category [14].

The switch has flow tables with flow entries lists. For each arriving packet, the switch matches packets based on the flow table. Selection is performed based on the highest available priority that matches the packet header. In case of similar priorities, the selected flow is not defined. If a packet does not match any row table and there is no table miss flow entry, then the packet is to be discarded. Otherwise, the packet should be processed according to the available miss policies. OpenFlow channel is considered as an interface to connect OpenFlow switches to the appropriate OpenFlow controllers. This interface guarantees that the controller makes the following operations (1) configures the switch, (2) receives various events from the switch, and (3) sends packets to the switch.

Types of messages that are allowed in these channels are controller to switch messages, switch to controller messages to update the controller about network conditions, and messages by either switches or controllers in emergency cases. In literature controllers are considered to be either centralized or distributed controllers. OpenFlow controller is a centralized controller that is responsible for maintaining poli-

cies and instructions among network elements. Policies determine the way to manage packets without matching flow entries and also manage the switch flow table by updating flow entries securely. TLS is used as the default security mechanism for the OpenFlow controller since the standard OpenFlow does not provide security [12, 13, 15–18].

The OpenFlow switch is able to establish connection and communication to various controllers. The reliability of the SDN network could be improved through utilizing multiple controllers by resisting failures. OpenFlow switches start by connecting all existing controllers although the messages to be sent should only be sent to the appropriate switch. Several research teams investigated about various security issues for OpenFlow [19–22].

Agborubere et al. [22] proposed a model to improve the OpenFlow and TLS communications security. The model summarized the TLS security issues by recommending techniques to improve the TLS. Authors in [22] focused on securing the OpenFlow controller through classifying different types of attacks using BN classifier for TLS. SDN controllers dynamically add or remove policies and rules. Administrators configure the network and deploy new protocols through the controllers. Therefore, the management of SDN greatly increases the programmability and flexibility of the network.

Meng et al. [12] proposed a management model based on Bayesian-models for insider attackers in healthcare environment. Tseng et al. [15] proposed a new model called ControllerSEPA. ControllerSEPA is considered as a light¬weight plugin project to protect the network against intruders.

Uchupala et al. [6] proposed an application-aware network based on spanning tree technique in addition to the shortest path routing network for various scenarios. Craiget et al. [21] proposed a technique for using bloom filter-based multicasting in SDN that achieves the substantial forwarding state reduction while eliminating false positive packet delivery. Song et al. [16] proposed a technique for SDN control plane to assign a group of the event processing module to switches. Authors in this work suggested to use switches for dealing with the OpenFlow events rather than the controllers.

Qiu et al. [20] suggested using global flow tables for global flow collection, computation of all the paths for network and also for global flow storing. Xiong et al. [17] proposed a queuing model for analyzing OpenFlow-based SDNs.

Silva et al. [19] proposed an extension to the OpenFlow protocol to include flexible time-triggered real-time communication services. The proposed model presented an extension to the SDN OpenFlow protocol that complements its functionality supporting real-time reservations.

3 $SSBN_{SDN}$ REFERENCE MODEL

This section has the reference model used in this paper. The reference model is referred to as *Secured Standard using Bayesian network for SDN (SSBN$_{SDN}$)* [23]. $SSBN_{SDN}$ has security features implemented into the OpenFlow controller. $SSBN_{SDN}$ uses Bayesian network model for packet filtering decisions in addition to filtering rules matching techniques. $SSBN_{SDN}$ implements packet filters through utilizing security features of OpenFlow controllers. $SSBN_{SDN}$ inspects packets based on its individual characteristic.

$SSBN_{SDN}$ depends on the process of packet inspection using BN classifiers to determine whether the packet is a malicious packet or not. $SSBN_{SDN}$ performs the filtering process by following the occurrence of the repeated attacks to a specific target with a certain probability. This is used as an indication that the host is being attacked. In this scenario, the policy is to store the attacking packet information in the flow table to be discarded. $SSBN_{SDN}$ uses Ryu framework for handling OpenFlow since it has conformance with OpenFlow specifications [13].

Each OpenFlow switch receives a packet on its input port, and hence a matching process is performed to match the packet to an entry in the flow table. In case that the packet does not exist in the flow table, the packet is sent to the controller for more inspection and processing. $SSBN_{SDN}$ uses packet filtering rules to be able to discard malicious packets. This includes the packet header information, an encapsulated protocol being used, a transport layer source and destination ports, an ICMP message type, and incoming and outgoing interfaces for the packet. In case of a packet hit scenario (i.e. matching occurs), the packet is considered as a *green* packet. If it is a packet miss scenario (i.e. no matching), the packet is considered as a *red* packet and hence is discarded.

$SSBN_{SDN}$ defines the problem as a set R=(r_1, r_2, ..., r_n) of orthogonal instances of the random variables (R_1, R_2, ..., R_n). As a result, the network N that best matches R could be found. One of the common approaches used for scoring is the Bayesian score approach. The score works by computing the probability of the data given the directed acyclic graph. Maximizing the Bayesian score using M is an NP-hard problem that could be solved using heuristic techniques. $SSBN_{SDN}$ computes the scores as shown in equation

1. It represents the probability of the graph.

$$S(N:R) = logP(N|R) = logP(R|N)+logP(N)+M \quad (1)$$

R||N denotes the average data over parametric assignments to N. $SSBN_{SDN}$ consists of a directed acyclic graph (DAG). DAG is composed of a set of nodes, one node is dedicated for each random variable. DAG contains a set of directed edges in additional to a set of conditional probability distributions. A conditional probability distribution for each node in terms of its parents could be defined as shown in equation 2.

$$P(X_i|Parents(X_i)) \quad (2)$$

A conditional distribution is represented as a conditional probability table (CPT) that gives a distribution over X_i for each combination of the parents' values. The conditional probability for the boolean X_i with k boolean parents has up to 2^k different possibility rows for all the combinations of the parent values. Each possibility requires probability of P for X_i to be true and a probability of $1-P$ for X_i to be false. The joint probability distribution could be represented as shown in equation 3.

$$P(X_1,\ldots,X_n) = \prod_{i=1}^{n} P(X_i|Parents(X_i)) \quad (3)$$

$SSBN_{SDN}$ utilizes a centralized OpenFlow controller model that establishes a pattern of trust among nodes and also detects untrusted devices for dynamic thresholds. Threshold values are adjusted by the administrator. $SSBN_{SDN}$ classifies the input packets into two classes; safe packets class and unsafe packets through filters. $SSBN_{SDN}$ assumes that if the probability of IP destination address is greater than a certain threshold it is considered as an unsafe packet otherwise it is a safe packet.

4 $SFBARMA_{SDN}$ PROPOSED MODEL

In this section, the proposed model is discussed. The proposed model is referred to as *secured feedback control using auto regressive moving average for SDN (SFBARMA$_{SDN}$)*. $SFBARMA_{SDN}$ is based on feedback control theoretic techniques. $SFBARMA_{SDN}$ works in two phases. (1) The system identification phase which is responsible for mathematically modeling the system, and (2) the control law phase that is responsible for detecting the safe and unsafe packet classes based filtering schemes.

Figure 1 has the block diagram for the feedback control system utilized by the proposed model $SFBARMA_{SDN}$. $SFBARMA_{SDN}$ adds security levels to the SDN through filtering based on ARMA approaches. The output is a higher secure system. After a time slot, the system becomes vulnerable to attacks and might get back again to the unsecured state. The observer module detects the level of vulnerability. and compares the security level to a threshold value. If the value exceeds the threshold value, then controller's control law takes the appropriate action to rectify this malicious behavior.

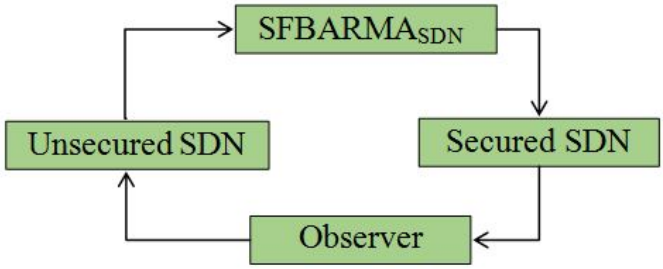

Figure 1: Block diagram for Feedback Control System for the $SFBARMA_{SDN}$ model

Figure 2 has the feedback control system for the $SFBARMA_{SDN}$ model. The *controller module* has the control function that modifies the tuning parameter. The *actuator module* is responsible of executing the appropriate actions based on ARMA approaches. The vulnerable system is referred to as the *controlled system module*. The *observer module* is used to detect the current status of the security level of the SDN. The output is fedback to the comparator to be compared to a certain reference value. The reference value is assumed to be 90% in this model.

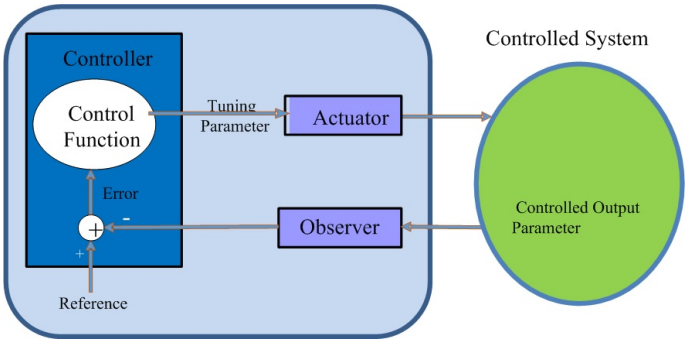

Figure 2: Feedback Control System for the DFBCP model

System identification phase focuses on using linear regression to model SDN elements. The generic time domain of the ARMA model is expressed in equation 4 in terms of the output of the module, $y(t)$,

as a function in the input, $x(t)$. The output is expressed as a series of the inputs. ARMA model has n series of the weighted values of the previous output and m weighted values of input values as shown in equation 4. The values of i and j are the indices values for the previous output and the input values respectively.

$$y(t) = \sum_{i=1}^{n} a_i \times y(t-i) + \sum_{i=0}^{m} b_j \times x(t-j) \quad (4)$$

In order to ease the mathematical modeling and derivation, a frequency Z-transform version of the ARMA is derived as shown in equation 5:

$$H(z) = \frac{Y(z)}{X(z)} = \frac{\sum_{j=0}^{m} b_j \times z^{n-j}}{z^n - (\sum_{i=1}^{n} a_i \times z^{n-i})} \quad (5)$$

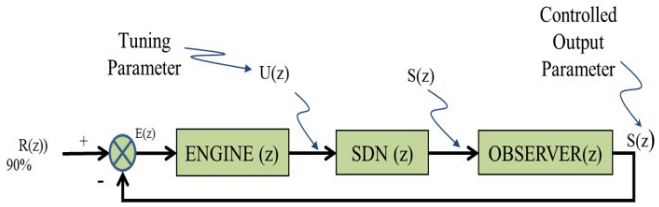

Figure 3: Modeling the SFBARMA$_{SDN}$ using feedback system

SFBARMA$_{SDN}$ models the SDN using a feedback control system as shown in figure 3. The SFBARMA$_{SDN}$ model is composed of the security engine that is responsible for maintaining the required security level. The SDN module represents the SDN network that needs to be secured including the overflow protocol. The observer module represents the sensing element that is responsible for reading the controlled output parameter (COP). SFBARMA$_{SDN}$ assumes that the COP is the enhanced security level of the system denoted by $s_i(t)$. The security engine module computes the tuning parameter of the model which is the current value of the security level of the system denoted by $u_i(t)$.

The relationship between the $s_i(t)$ and the $u_i(t)$ is given by equation 6 by applying into the ARMA mathematical model explained in equation 4. For simplicity, the values of n and m were chosen to be 1 and 0 respectively.

$$s_i(t) = a_1 s_i(t-1) + b_0 u_i(t) \quad (6)$$

4.1 SFBARMA$_{SDN}$ Control Law and Gain Design

SFBARMA$_{SDN}$ engine module uses proportional integral (PI) controller due to its simplicity. SFBARMA$_{SDN}$ uses the control law to update the tuning parameter to continuously work on minimizing the current error value in a feedback fashion. The SFBARMA$_{SDN}$ control law is given in equation 7.

$$u_i(t) = u_i(t-1) + K_i e_i(t-1) \quad (7)$$

The error value which is fed as an input to the SFBARMA$_{SDN}$ controller is calculated as a result of the difference between the current reference value and the current level of security level as shown in equation 8.

$$e_i(t) = ref_i(t) - s_i(t) \quad (8)$$

In order to be able to compute a stable gain K, root locus is used for the appropriate gain selection. Root locus plot is shown in figure 4. The gain K is chosen to be 0.5 to obtain the closed loop root of 0.47.

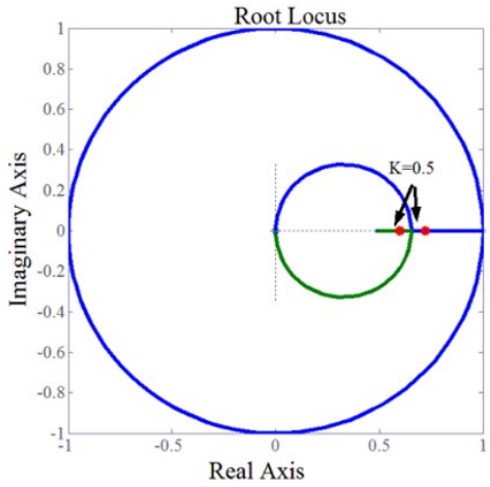

Figure 4: Root Locus for the SFBARMA$_{SDN}$ Model

4.2 SFBARMA$_{SDN}$ Transfer Functions

Transfer functions are mathematical models that express the relationship between the outputs and the inputs. Transfer functions are the frequency domain equivalent for the time domain is described in equation 9 through equation 11. Equation 9 has the Z-transform equivalence for equation 6.

$$T(z) = \frac{S_i(z)}{U_i(z)} = \frac{b_0 z}{z - a_i} \quad (9)$$

The controller engine module's transfer function is shown in equation 10. The transfer function carries the control law that relates the tuning parameter current security level of the system to the error value tuning parameter.

$$G_i(z) = \frac{U_i(z)}{E_i(z)} = \frac{K_i}{z-1} \tag{10}$$

To compute the aggregated transfer function of the overall system, the result is obtained by multiplying the transfer functions for each module resulting in equation 11.

$$W(z) = \frac{S(z)}{R(z)} = \frac{K \times z \times (z \times b_0)}{(z-1) \times (z-a_1) + K \times z \times (z-b_0)} \tag{11}$$

$\text{SFBARMA}_{\text{SDN}}$ uses least squares regression to estimate the values of the parameters of the ARMA model $a1$ and $b0$. $a1$ is estimated to be 0.4364, and $b0$ is estimated to be 0.2897. The goodness of the model is measured using R^2. R^2 is measured to be 87.5% as an indication of the linearity of the model.

The $\text{SFBARMA}_{\text{SDN}}$ filtering algorithm starts by observing of the sequence of the packets to make sure that the data is a reliable data. If the data is stationary, $\text{SFBARMA}_{\text{SDN}}$ calculates the *Akaike's information criterion (AIC)*. Packets are filtered, if packets pass the selection criteria, then packet are considered as *green* packets, otherwise they are considered *red* packets and discarded. The flowchart for $\text{SFBARMA}_{\text{SDN}}$ is shown in figure 5.

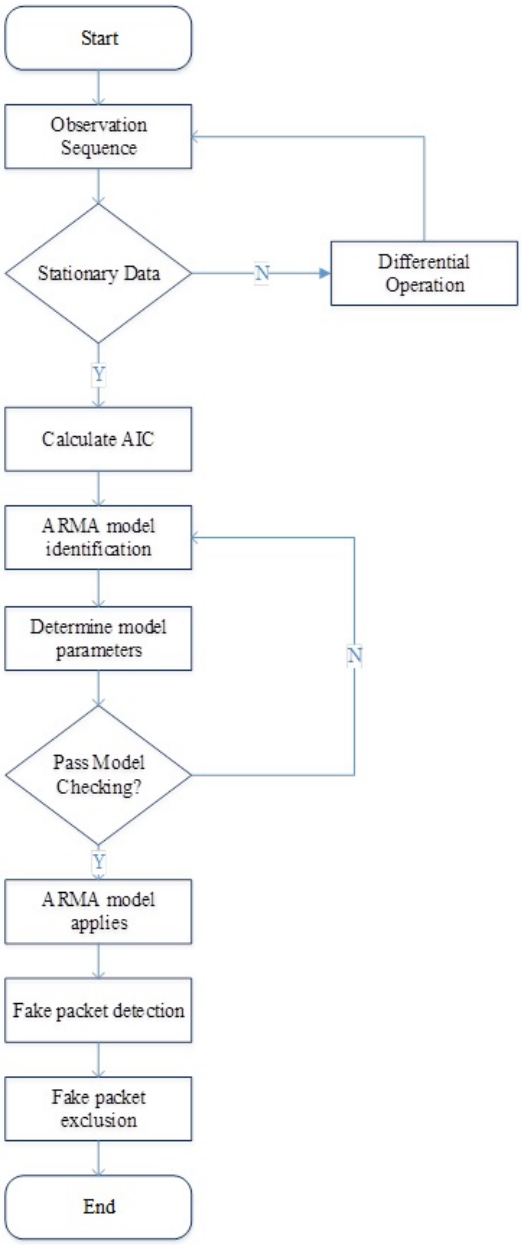

Figure 5: $\text{SFBARMA}_{\text{SDN}}$ Filtering

5 EXPERIMENTAL TESTBED AND RESULTS

This section provides the experimental testbed followed by the experimental results.

5.1 Experimental Testbed

$\text{SFBARMA}_{\text{SDN}}$ adds a feedback module based on ARMA regression model that is responsible for adding a security capability to the OpenFlow controller. The added module is responsible for maintaining the required reference value of the security level by computing the overall security in addition to the computation overhead. In order to evaluate the security of the proposed system, a prototype is implemented by adding security capabilities to the existing OpenFlow controller using Python scripts. To simulate attacks, *freedom of information (FOI)* data sets are used. *Mininet* software emulator tool is installed on Linux machine. *Wireshark* is a protocol analyzer that is responsible for observing the inbound and outbound packets. The network emulation runs on an HP machine with Intel Core i5-2520M, CPU 2.50 GHz and RAM 8 GB running Ubuntu (version 20.04) 64 bits.

In the experimentation setting, attacks were varying to be from 10 to 100 with a step of 10 while the number of packets generated by each attack is set to be 1000. Extensive topologies were used in the experimentation phase. A case study topology that was used for experimentation purposes is shown in figure 6. The experimental topology is composed of Ryu SDN controller that is used as OpenFlow framework.

Ryu has high conformance with OpenFlow specifications [13]. The topology consists of six hosts that are connected to the OpenFlow router. The OpenFlow router is substituted by three types of routers in the experimentation phase. The basic reference OpenFlow, the Bayesian based router $SSBN_{SDN}$ and the secured feedback model based on ARMA. Added capabilities to secure the OpenFlow controller is performed. Malicious intruders controllers represent the various malicious attacks.

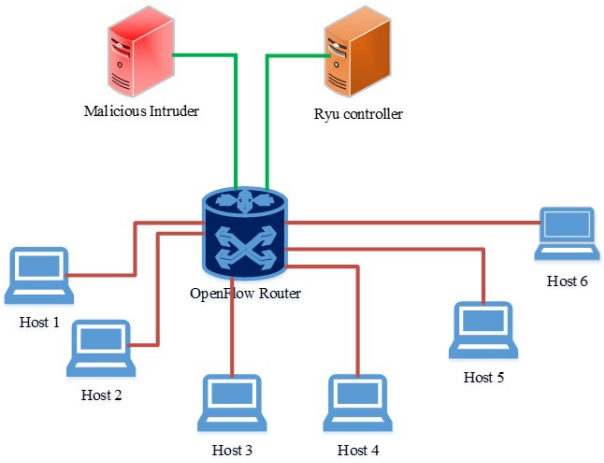

Figure 6: Experimental Topology

5.2 Experimental Results

In order to evaluate the detection accuracy of the malicious intrusion, $SFBARMA_{SDN}$ adds security capabilities to the standard OpenFlow controller. $SFBARMA_{SDN}$ is compared to both the standard OpenFlow controller and also to the $SSBN_{SDN}$.

$SFBARMA_{SDN}$ assumes that the success rate to filter fake packets to is set to 90%. Figure 7 compares the number of fake packets for the $SFBARMA_{SDN}$ and the two reference models; the standard OpenFlow controller and the $SSBN_{SDN}$.

Results show that $SFBARMA_{SDN}$ can detect fake packets with a percentage of 91% as opposed to 88% for the $SSBN_{SDN}$. Both algorithms outperformed the standard OpenFlow controller as depicted in Figure 7.

Figure 8 shows the comparison of the processing times for the standard OpenFlow, $SSBN_{SDN}$ and the proposed $SFBARMA_{SDN}$. Experiments show that the percentage of overhead introduced by the $SFBARMA_{SDN}$ when compared to the $SSBN_{SDN}$ and the standard OpenFlow was limited to be 3% and 5% respectively. This indicates that the proposed $SFBARMA_{SDN}$ does not produce extra overhead when compared to the standard OpenFlow and the $SSBN_{SDN}$.

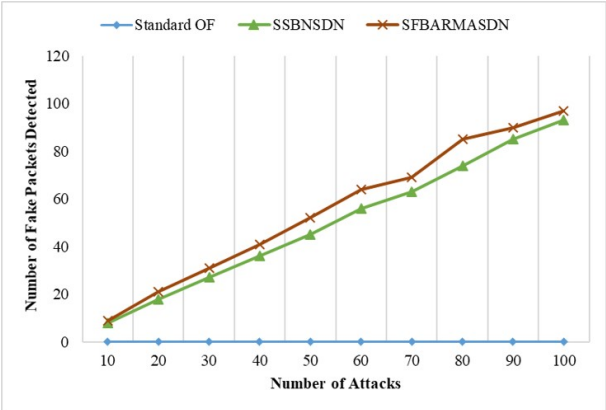

Figure 7: Comparison of the fake packets detected for the standard OpenFlow, $SSBN_{SDN}$ and $SFBARMA_{SDN}$

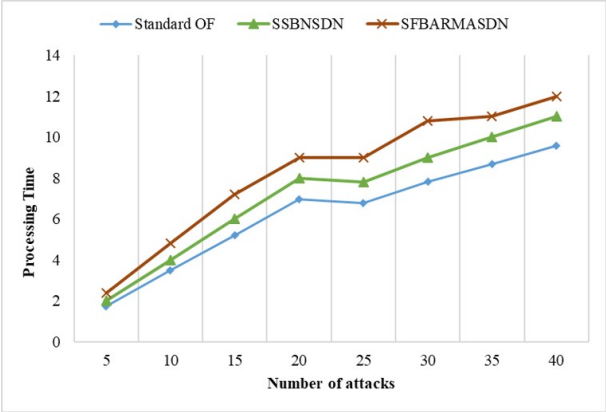

Figure 8: Comparison of the processing time among the Standard OpenFlow, $SSBN_{SDN}$ and $SFBARMA_{SDN}$

6 CONCLUSION AND FUTURE WORK

This work proposes a novel model to add extra security capability to the standard OpenFlow controllers in SDN networks. The proposed model is referred to as Secured Feedback model using Autoregressive Moving Average (ARMA) for SDN networks ($SFBARMA_{SDN}$). $SFBARMA_{SDN}$ adds a security capability to the OpenFlow controller through classifying various attacks using packet filters. The filter inspects the properties of the fake packets through feedback control theoretic techniques. $SFBARMA_{SDN}$ uses ARMA models to filter fake packets and hence improve the security of the OpenFlow. $FBARMA_{SDN}$ calculates the *Akaike's information criterion (AIC)*. Packets are filtered, if packets pass the selection criteria, then packet are considered as *green* packets, oth-

erwise they are considered *red* packets and discarded. In order to measure the added value for the proposed model. The $SFBARMA_{SDN}$ is compared to two reference models. The first reference model is based on Bayesian networking and the second reference model is the standard OpenFlow model. The standard OpenFlow controller optionally implements the Transport Layer Security (TLS).

Extensive experiments were conducted to test the performance of the proposed model. The number of fake packets were measured and compared to two reference models. $SFBARMA_{SDN}$ outperformed both $SSBN_{SDN}$ and the standard OpenFlow in different scenarios by an average value of 7%. The processing time overhead was computed. The processing overhead was around 3% for the $SFBARMA_{SDN}$ compared to the $SSBN_{SDN}$. A virtual network was established using an improved version of the Ryu controller to add the security and filtering capabilities. Results are very promising, $SFBARMA_{SDN}$ has outperformed reference models with minimum overhead.

As a future work to this work is to use more complicated topologies to test more complex scenarios for the $SFBARMA_{SDN}$. Also artificial intelligent based techniques could also be utilized for filtering. Another research direction for improvement could be to use the ARMA based models to improve NOX, POX OpenFlow controllers.

References:

[1] M. Sjoholmsierchio, B. Hale, D. Lukaszewski, and G. Xie, "Strengthening sdn security: protocol dialecting and downgrade attacks," in *2021 IEEE 7th International Conference on Network Softwarization (NetSoft)*, pp. 321–329, IEEE, 2021.

[2] W. H. F. Aly, "Lbftfb fault tolerance mechanism for software defined networking," in *2017 International Conference on Electrical and Computing Technologies and Applications (ICECTA)*, pp. 1–5, IEEE, 2017.

[3] W. H. F. Aly, "A novel fault tolerance mechanism for software defined networking," in *2017 European Modelling Symposium (EMS)*, pp. 233–239, IEEE, 2017.

[4] W. H. F. Aly and Y. Kotb, "Towards sdn fault tolerance using petri-nets," in *2018 28th International Telecommunication Networks and Applications Conference (ITNAC)*, pp. 1–3, IEEE, 2018.

[5] W. H. F. Aly and A. M. A. Al-anazi, "Enhanced controller fault tolerant (ecft) model for software defined networking," in *2018 Fifth International Conference on Software Defined Systems (SDS)*, pp. 217–222, IEEE, 2018.

[6] I. Z. Bholebawa and U. D. Dalal, "Performance analysis of sdn/openflow controllers: Pox versus floodlight," *Wireless Personal Communications*, vol. 98, no. 2, pp. 1679–1699, 2018.

[7] W. H. F. Aly, "Generic controller adaptive load balancing (gcalb) for sdn networks," *Journal of Computer Networks and Communications*, vol. 2019, 2019.

[8] W. H. F. Aly, "Controller adaptive load balancing for sdn networks," in *2019 Eleventh International Conference on Ubiquitous and Future Networks (ICUFN)*, pp. 514–519, IEEE, 2019.

[9] Z. Cai, C. Hu, K. Zheng, Y. Xu, and Q. Fu, "Network security and management in sdn," 2018.

[10] Y. Wang, S. Liu, S. Zhang, Y. Huang, and K. Fan, "A filter algorithm based on arma model to suppress the influence of atmospheric disturbance in laser straightness measurement," in *Tenth International Symposium on Precision Engineering Measurements and Instrumentation*, vol. 11053, pp. 667–675, SPIE, 2019.

[11] J. C. C. Chica, J. C. Imbachi, and J. F. B. Vega, "Security in sdn: A comprehensive survey," *Journal of Network and Computer Applications*, vol. 159, p. 102595, 2020.

[12] W. Meng, K.-K. R. Choo, S. Furnell, A. V. Vasilakos, and C. W. Probst, "Towards bayesian-based trust management for insider attacks in healthcare software-defined networks," *IEEE Transactions on Network and Service Management*, vol. 15, no. 2, pp. 761–773, 2018.

[13] C. Fernandez and J. Muñoz, "Software defined networking (sdn) with open flow 1.3," *Open vSwitch and Ryu,(June 2010)*, vol. 183, 2016.

[14] W. Li, W. Meng, and L. F. Kwok, "A survey on openflow-based software defined networks: Security challenges and countermeasures," *Journal of Network and Computer Applications*, vol. 68, pp. 126–139, 2016.

[15] Y. Tseng, Z. Zhang, and F. Naït-Abdesselam, "Controllersepa: a security-enhancing sdn controller plug-in for openflow applications," in

2016 17th International Conference on Parallel and Distributed Computing, Applications and Technologies (PDCAT), pp. 268–273, IEEE, 2016.

[16] P. Song, Y. Liu, T. Liu, and D. Qian, "Controller-proxy: Scaling network management for large-scale sdn networks," *Computer Communications*, vol. 108, pp. 52–63, 2017.

[17] B. Xiong, K. Yang, J. Zhao, W. Li, and K. Li, "Performance evaluation of openflow-based software-defined networks based on queueing model," *Computer Networks*, vol. 102, pp. 172–185, 2016.

[18] L. Silva, P. Gonçalves, R. Marau, P. Pedreiras, and L. Almeida, "Extending openflow with flexible time-triggered real-time communication services," in *2017 22nd IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, pp. 1–8, IEEE, 2017.

[19] Y. Watashiba, K. Ichikawa, H. Iida, *et al.*, "Application-aware network: Network route management using sdn based on application characteristics," *CSI Transactions on ICT*, vol. 5, no. 4, pp. 375–385, 2017.

[20] X. Qiu, K. Zhang, and Q. Ren, "Global flow table: A convincing mechanism for security operations in sdn," *Computer Networks*, vol. 120, pp. 56–70, 2017.

[21] A. Craig, B. Nandy, I. Lambadaris, and P. Koutsakis, "Bloomflow: Openflow extensions for memory efficient, scalable multicast with multi-stage bloom filters," *Computer Communications*, vol. 110, pp. 83–102, 2017.

[22] B. Agborubere and E. Sanchez-Velazquez, "Openflow communications and tls security in software-defined networks," in *2017 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*, pp. 560–566, IEEE, 2017.

[23] N. Sophakan and C. Sathitwiriyawong, "Securing openflow controller of software-defined networks using bayesian network," in *2018 22nd International Computer Science and Engineering Conference (ICSEC)*, pp. 1–4, 2018.