International Journal of Computer Networks and Applications (IJCNA)
DOI: 10.22247/ijcna/2021/209704 Volume 8, Issue 4, July – August (2021)

RESEARCH ARTICLE

Modified Deep Learning Methodology Based Malicious Intrusion Detection System in Software Defined Networking

Thangaraj Ethilu
Department of Computer Science and Engineering, Annamalai University, Chidambaram, Tamil Nadu, India.
ethilthangaraj@yahoo.co.in

Abirami Sathappan
Department of Computer Science and Engineering, Annamalai University, Chidambaram, Tamil Nadu, India.
reachabisv@gmail.com

Paul Rodrigues
Department of Computer Science and Engineering, DMI College of Engineering, Chennai, Tamil Nadu, India.
drpaulprof@gmail.com

Received: 29 June 2021 / Revised: 29 July 2021 / Accepted: 05 August 2021 / Published: 28 August 2021

Abstract – Software Defined Networking (SDN) has increased a high-level attention in recent years, mainly because of its ability to address the cyber security challenges. Machine learning architectures were developed as the SDN system to detect the security threads; however, present techniques are limited with (i) higher computation time during malicious switch detection, (ii) reduced malicious switch detection rate (MSDR). This paper presents modified deep learning architecture based SDN system consist of two stages: (i) training stage, computes the external feature maps from both trusted and malicious network switches connected to the SDN controller, (ii) testing stage, classifying the trust and malicious switches connected with SDN controller. The feature maps are trained and classified with Modified LeNET Convolutional Neural Networks (CNN) architecture. The proposed methodology is simulated via network simulator under environmental constraint conditions. The results shows that the proposed methodology reduced the malicious switch detection computational time about a half as well as it increased the MSDR to about 6% compared to the conventional methodologies.

Index Terms – SDN, Switch, Malicious, CNN, Feature Maps.

1. INTRODUCTION

Software Defined Networking (SDN) is a networking technology which separates the control and user plane in the networking model to enhance the functional features. The network administrator in control plane controls the whole networking system using the developed application programs [1-4]. The control plane in SDN networking is SDN controller and the user plane in SDN networking is networking switches. Many enterprises require SDN networking system to reduce their functional operating cost. The SDN system is constructed with the two individual components as SDN controller and SDN switches. The SDN controller must be unique for the entire SDN system and it may be located in remote cloud environment for faster application. The SDN switches are the user plane devices which are act as the data centers to SDN controllers [5-7]. The management plane controls the networking administration functionalities of SDN controller in SDN system as illustrated in Figure 1.

The Application Programming Interfaces (API) is developed in SDN system (management plane) which is used to interface between SDN controller and SDN switches [8-9]. These switches in SDN system may be either trusted or malicious switch. The malicious switch in SDN system transfers irrelevant or dummy packets to the SDN controller to degrade the energy networking performance by increasing the energy consumption rate. In order to overcome such limitation in conventional SDN system, this article proposes a methodology to detect the malicious switch to improve the network performance. The conventional methods for the detection of malicious switches in SDN used machine learning algorithm. These conventional methods also used external features only to detect the malicious switches in SDN. These methods consumed more computational time and less PDR due to its complex architecture. Hence, there is a requirement for the proposed system for the detection of malicious switch in SDN using simplified architecture. This paper develops the malicious and trust switch detection system in SDN using modified deep learning architecture.

RESEARCH ARTICLE

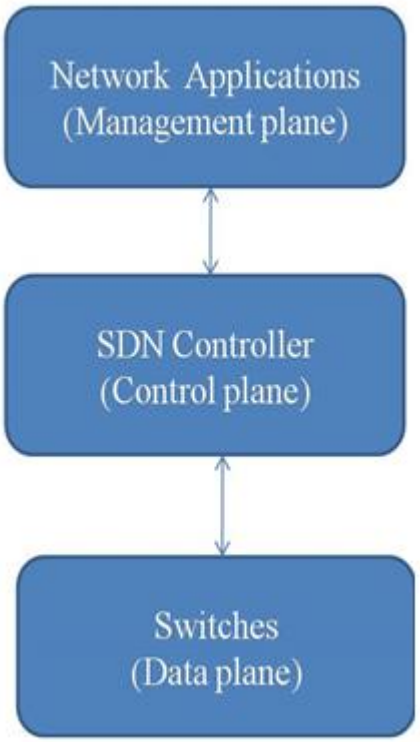


Figure 1 SDN System

The objectives of the research work are stated as follows.

- To detect the malicious switches in Software Defined Networking using modified deep learning architecture.
- To improve the malicious switch detection rate by modifying the conventional LeNET CNN architecture through the alterations in internal layers.
- To improve the performance analysis parameters MSDR, PDR and computational time of the proposed malicious switch detection system.

1.1. Organization of Paper

The organization of the succeeding manuscript are as follows: Section 2 discusses the literature survey of this paper and conventional security enhancement methods in SDN system, section 3 depicts the problem statement, section 4 presents the proposed methodology to detect malicious switch in the SDN, section 5 discusses the simulation results and section 6 concludes the paper.

2. LITERATURE SURVEY

Chasaki et al. (2021) detected and classified malicious host nodes and switches in SDN using system call learning method. The unusual activity or transfer of data from the switches was detected using the system call learning patterns. The main limitation of this method was that it consumed more computational time for classifying the trust and malicious switch [10]. Marek Amanowicz et al. (2021) used different data mining methodologies for detecting the malicious switches in SDN. The authors mainly structured the data mining rules for the classification of the features in order to detect and identify the malicious switches from the trust switches. The data transferring rate was low due to its unstructured data pattern as the main limitation of this work [11]. Derhab et al. (2021) designed microcontroller architecture for detecting the malicious switches in SDN. The authors used block chain methodology for classifying each switch in SDN into either malicious or trusted. The energy consumption of this methodology was high due to its hardware based architecture [12].

Nife et al. (2020) proposed an application-aware based firewall mechanism in SDN networking controller to enhance the security features of the network. This proposed work consisted using four different modules as Main Module, Filtering Module, Application Identification Module (AIM), and the Security-Enforcement Module (SEM). The main module was developed at the SDN controller level and the filtering module filters the incoming and outgoing packets, the application mechanism was handled by AIM with the security improvement by SEM [13]. The limitation of this work was its less PDR. Sebbar et al. (2020) detected Man in the Middle (MitM) attack which was the severe and complex attack in SDN system which degraded the energy efficiency of the system. This attack captured the data flow between the network switch and the SDN controller and duplicated the captured packets. These duplicated packets were spoofed into SDN controller to degrade its performance efficiency. In order to detect and mitigate these attacks in SDN system, context-based node acceptance based on the random forest model (CBNA-RF) was developed in this work using multi-level security policies. The limitation of this work was its high computational time period for detecting the malicious switches in large scale SDN environment [14]. Chi et al. (2020) used SanboxNet to detect the malicious functional activities in SDN controller. The application developed at SDN controller (controller plane) was used to detect the malicious attacks which were generated by network devices (user plane). The authors analyzed the performance of the developed SanboxNet model with respect to energy consumption and packer delivery rate. This developed architecture is not suitable for large scale SDN environment [15].

Neu et al. (2018) developed Lightweight Intrusion Prevention System (LIPS) for the performance improvement in SDN networks. The proposed LIPS methodology stated in this work detected and mitigated port scan attacks which was often occurred on SDN network devices. The stages in the proposed LIPS methodology were Collection Module (CM), Detection Module (DM) and Prevention Module (PM). The

RESEARCH ARTICLE

function of CM was to collect the data resources from various switch devices in network and these data were analyzed to detect the port scan attack using the predefined rules. The PM module installed set of rules on SDN controller to mitigate them by the detected port scan attacks. The main limitation of this work is that it does not support large number of switches in SDN [16]. Chang et al. (2018) developed cloud-based clustering firewall in SDN network system to enhance their security features. The authors constructed and applied rule-placement algorithm for the better improvement in the life cycle of Ternary Content-Addressable Memory (TCAM) in SDN controller. The application layer of this proposed algorithm was installed in cloud networking to speed up the proposed methodology for the detection of intruders in SDN system. Low detection rate is the limitation of this work [17].

From the literature survey, the following limitations are observed.

- High computational time for detecting the malicious switches.
- Less PDR and MSDR.

In order to overcome such limitations in the conventional methods for detecting the malicious and trust switches in SDN, the modified LeNET CNN architecture is proposed in this paper.

3. PROBLEM STATEMENT

Intrusion Detection System (IDS) is important for detecting the unauthorized activities in SDN system. Many researchers developed methodologies to detect various attacks in Application layer of the SDN system (SDN controller) to improve the performance efficiency. There is very limited research focused to detect the unauthorized activities in SDN system (control plane), which is also important to improve the networking performance. Hence, the malicious switches which are located in data plane of the SDN system, are detected using deep learning architecture in this article.

4. PROPOSED METHODOLOGY

Using deep learning architecture, the performance efficiency of the SDN network is increased in this article by recognizing and classifying network switches as either trusted or malicious switch devices. The proposed methodology is divided into two stages: training and testing, which are depicted in Figures 2(a) and 2(b), respectively. The trusted network switch, malicious network switch, features computations and deep learning algorithm are all part of the training stage. It computes the external feature maps from both trusted and malicious network switches which are connected with SDN controller. The externally computed feature maps are trained with Modified LeNET CNN architecture which produces the trained patterns. The testing stage includes unknown network switch, feature computations and deep learning algorithm. External feature maps are computed from switches connected to the SDN controller (to check for malicious actions). The externally computed feature maps from this network switch are classified with Modified LeNET CNN architecture which produces the classified responses as either trusted or malicious network switch.

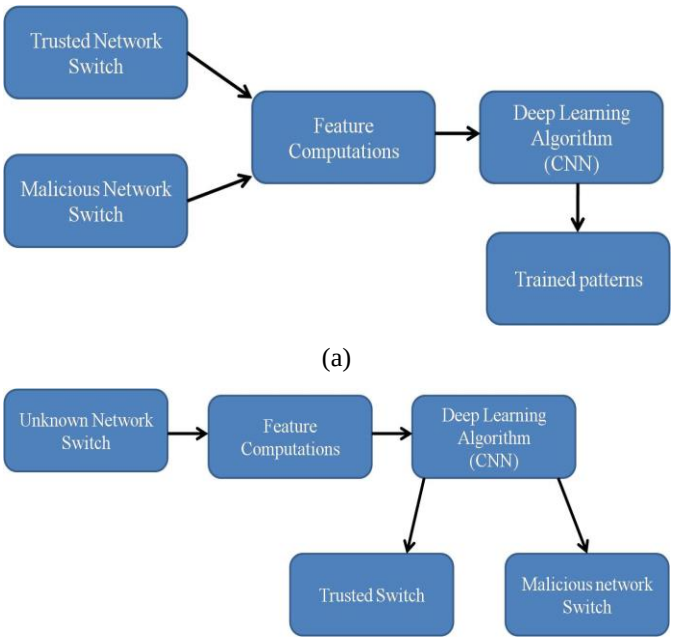


Figure 2 Malicious switch detection systems using deep learning algorithm (a) Training (b) Testing

4.1. External Feature Maps Construction

The CNN classification architecture for the classification of unknown network switch into either trust switch or malicious switch requires both external and internal feature maps. The internal feature maps will be generated by the CNN architecture itself and the external feature maps are computed externally from the switch device which is to be tested.

The weight feature map is generated using the following equation (1).

$$WFM = \frac{\alpha_i * \beta_i}{w_i} \quad (1)$$

Where, α_i is the alpha index factor and β_i is the beta index factor with the computational weightw_i.

The alpha index factor (α_i) is computed based on the number of packet flow between switch and the SDN controller and it is given in the following equation (2).

$$\alpha_i = \frac{P_{sc} * P_r}{N} \quad (2)$$

RESEARCH ARTICLE

Where, P_{sc} is the number of packets successfully transmitted from switch (s) to SDN controller (c) over the time 't', P_r is the total number of packets successfully received at SDN controller from the switch (s) and N is the total number of packets actually transmitted from s to c.

The beta index factor (β_i) is computed based on the number packets dropped out between switch and SDN controller and it is computed using the following equation (3).

$$\beta_i = \frac{d_{sc}}{N} * E_s \quad (3)$$

Whereas, d_{sc} is the total dropout packets from s to c and E_s is the energy consumption of the switch (s).

The weight index (w_i) of the SDN controller is computed using the following equation (4).

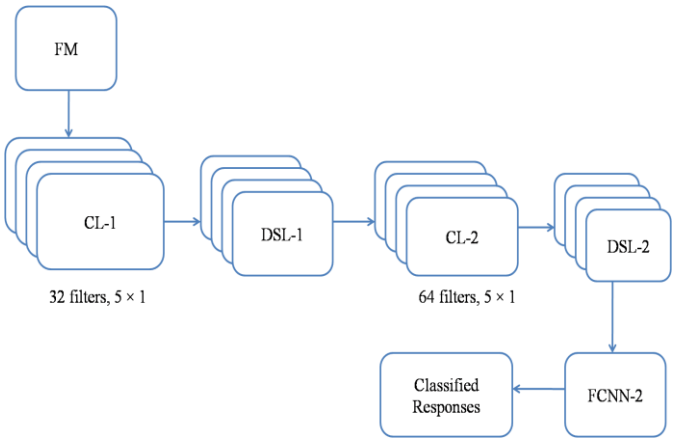
$$w_i = \frac{\sum_{i=1}^{N1} P_{ic}}{N1} \quad (4)$$

Whereas, P_{ic} is the number of packets received by SDN controller from its surrounding switches and N1 is the total number of surrounding switches over SDN controller.

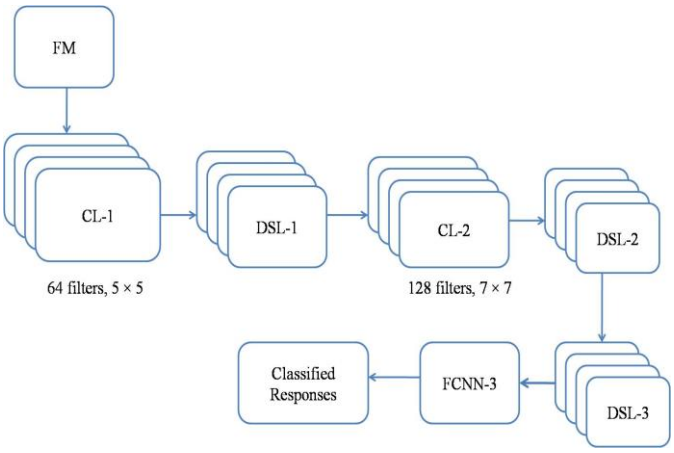
The computed feature is stored in a matrix which is called as Feature Map (FM) and it is fed into the proposed CNN architecture for identifying the malicious switches in SDN network system.

4.2. Proposed CNN Architecture

The conventional and modified LeNET CNN architectures are depicted in Figure 3 (a) and Figure 3 (b) respectively. The proposed CNN architecture used in this article for the detection of malicious switch from the trust switch is derived from the conventional CNN architecture. The CNN architecture is constructed with Convolutional Layers (CL) and Down Sampling Layers (DSL) with Fully Connected Neural Networks (FCNN). The conventional and modified CNN architecture are differ from the number of CL, DSL and FCNN layers and the number of parameters.



(a)



(b)

Figure 3 LeNET (a) Conventional CNN Architecture (Rinki Gupta et al. 2020) (b) Conventional CNN Architecture

In conventional CNN architecture, the first CL is constructed with 32 filters with the kernel size of 5*1. The size of the output from the first CL is reduced by passing the CL1 output through DSL-1 with the kernel size of 5*1. The second CL is constructed with 64 filters with the kernel size of 5*1. The size of the output from the second CL is reduced by passing the CL2 output through DSL-2 with the kernel size of 5*1. The output features from DSL-2 is fed into 2 FCNN layers which produces the classified responses.

In modified CNN architecture, the first CL is constructed with 64 filters with the kernel size of 5*5. The function of this CL is to convolve the extracted external feature maps (FM) with the kernel of each filter which is located in CL, as illustrated in the following equation (5).

$$CL_{outputi} = FM * Fi; \text{ i=1,2….N;} \quad (5)$$

Whereas, FM is the externally computed feature map and Fi is the kernel of each filter in CL, N is the number of filters in each CL.

The output of each filter is stored in a new feature map and this map is passed to the next DSL.

The output feature maps from this CL-1 may have the negative impact values which affect the further processing of these computed feature maps. Hence, Linear Rectification Unit (ReLU) is placed after every CL in this CNN architecture which transforms the negative feature maps into positive feature maps as illustrated in the following equation (6).

$$b = \begin{cases} b; if\ b\ is\ positive \\ -b; if\ b\ is\ negative \end{cases} \quad (6)$$

Whereas, b is the feature map output which is computed from CL-1.

RESEARCH ARTICLE

The size of the output from the first CL is reduced by passing the CL1 output through DSL-1 with the kernel size of 5*5. The second CL is constructed with 128 filters with the kernel size of 7*7. The size of the output from the second CL is reduced by passing the CL2 output through DSL-2 and DSL-3 with the kernel size of 7*7. The output features from DSL-3is fed into 3-FCNN layers which produces the classified responses as either malicious switch or trust switch.

The first FCNN is constituted with 1024 number of neurons, second FCNN is constituted with 512 numbers of neurons and the third FCNN is constituted with 2 numbers of neurons. The number of neurons is fixed after several iterations with the proposed system to produce maximum output.

Modules in CNN Architecture	Conventional CNN Architecture	Proposed CNN Architecture
Number of CLs	2	2
Number of DSL	2	3
Number of FCNN	2	3
Neurons in FCNN-1	1024	1024
Neurons in FCNN-2	2	512
Neurons in FCNN-3	-	2

Table 1 Difference between Conventional and Proposed CNN Architectures

Table 1 shows the difference between conventional and proposed CNN architectures with respect to number of CL, DSL and FCNN layers. The performance of the proposed system is improved by increasing the number of DSL and FCNN layers in CNN architecture.

The proposed algorithm 1 for malicious switch detection system is given below.

Inputs: Switches in SDN;

Output: Trust or malicious switch;

Start;

Step 1: Feature maps are constructed from well-known trust and malicious switch during training mode.

Step 2: The computed feature maps are trained using modified LeNET CNN architecture in order to produce the training pattern.

Step 3: Compute the feature map from the unknown switch (which is to be tested) and they are classified using modified LeNET CNN architecture with training pattern, which is obtained from step 2.

Step 4: The final response from modified LeNET CNN architecture is either malicious or trust.

Step 5: Repeat step 1 to 4 until no more switches in simulation environmental area.

End;

Algorithm 1Malicious Switch Detection System in SDN

Table 2 shows the specifications of the modified LeNET CNN architecture.

Layers	Modified LeNET architecture (Proposed in this work)			Conventional LeNET architecture (Rinki Gupta et al. 2020)		
	Number of filters	Kernel size	Number of neurons	Number of filters	Kernel size	Number of neurons
CL-1	64	5*5	-	32	5*1	-
DSL-1	64	5*5	-	32	5*1	-
CL-2	128	7*7	-	64	5*1	-
DSL-2	128	7*7	-	64	5*1	-
DSL-3	128	7*7	-	-	-	-
FCNN-1	-	-	1024	-	-	1024
FCNN-2	-	-	512	-	-	2
FCNN-3	-	-	2	-	-	-

Table 2 Specifications of the Modified and Conventional LeNET CNN Architectures

RESEARCH ARTICLE

5. RESULTS AND DISCUSSIONS

In this article, the proposed SDN system is implemented in SDN controller which is used to detect the malicious switch in the defined network system. This proposed methodology is simulated in Network Simulator version 2 environment under environmental constraint conditions. In this simulation environment 100 numbers of switches and one SDN controller is used in the simulation area of 1000m*1000m. The baud rate between each switch and SDN controller is 250 Mb/s. Among these 100 switches, 20 switches are converted into malicious switch and the remaining 80 switches are act as trust switches. The simulation environmental setup is illustrated in Table 3.

Simulation parameters	Setting values
Numbers of total switches	100
Number of SDN controller	1
Simulation area	1000m*1000m
Baud rate	250 Mb/s.
Number of malicious switches	20

Table 3 Simulation Setup

Malicious Switch Detection Rate (MSDR) is a parameter which is used to analyze the performance of the proposed SDN system as described in the following equation (7).

$$MSDR = \frac{Number\ of\ correctly\ detected\ malicious\ switches}{Total\ number\ of\ malicious\ switches} * 100\% \quad (7)$$

Methodology	Total malicious switches	Correctly detected malicious switches	MSDR (%)
Modified LeNET	20	19	95
Conventional LeNET	20	17	85

Table 4 Performance Comparisons of Proposed Malicious Switch Detection system in SDN with Respect to CNN Architectures

In this article, 19 malicious switches are detected over 20 malicious switches using the proposed methodology (modified LeNET CNN architecture) stated in this work. Hence, the MSDR of the proposed system is about 95%. The performance comparisons of the proposed methodology with respect to conventional and modified LeNET CNN architectures are depicted in Table 4. From Table 4, the conventional LeNET CNN architecture correctly detects 17 malicious switches over 20 malicious switches and achieves 85% of MSDR and the proposed LeNET CNN architecture correctly detects 19 malicious switches over 20 malicious switches and achieves 95% of MSDR. It is observed from Table 4, the proposed malicious switch detection system using modified CNN architecture provides superior MSDR performance when compared with conventional CNN architecture.

PDR is the performance analysis parameter which is defined as the ratio between the number of packets correctly received by SDN controller and the switches in SDN network. It is measured in percentage and has the value between 0 and 100. The performance of the proposed SDN is high if the average PDR is having high value. In this article, the performance of the PDR is computed by varying the number of malicious switches count in the proposed SDN system with respect to conventional LeNET and proposed LeNET CNN architectures, as illustrated in Table 5.

	PDR (%)	
Number of Malicious Switches	Modified LeNET	Conventional LeNET
2	98.2	93.1
4	96.1	91.9
6	95.2	88.6
8	93.7	83.1
10	92.1	80.4
12	90.5	79.4
14	88.6	77.9
16	86.1	75.3
18	84.9	72.9
20	81.9	70.7

Table 5 PDR Performance of Proposed SDN System using Conventional and Modified CNN Architecture

The proposed SDN system using modified CNN architecture obtains 93.1% of PDR for 2 numbers of malicious switches in the simulation environment. The proposed SDN system using conventional CNN architecture obtains 70.7% of PDR for 20 numbers of malicious switches in the simulation environment. The proposed SDN system using modified CNN architecture obtains 98.2% of PDR for 2 numbers of malicious switches in the simulation environment. The proposed SDN system using modified CNN architecture obtains 81.9% of PDR for 20

RESEARCH ARTICLE

numbers of malicious switches in the simulation environment. From Table 5, it is observed that the proposed malicious switch detection SDN system using modified CNN architecture provides high performance when compared with the conventional CNN architecture.

Figure 4 is the graphical illustrations of the proposed system in terms of PDR.

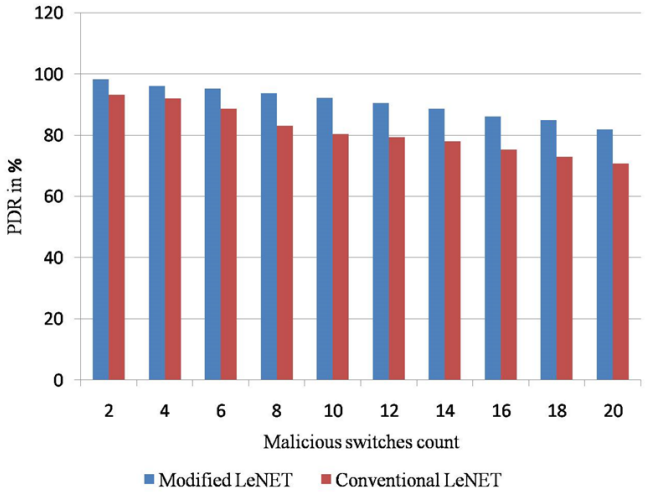


Figure 4 Graphical Illustrations of the Proposed System in Terms of PDR

Computational time is another performance evaluation parameter in SDN system, defined as the detection time of the malicious switch in SDN system using the proposed CNN classification methodology. It is computed in terms of ms and its computational value is based on the presence of number of malicious switches in SDN system.

Table 6 is the performance of proposed SDN system using conventional and modified CNN architecture in terms of computational time.

	Computational Time (ms)	
Number of Malicious Switches	Conventional LeNET	Modified LeNET
2	0.12	0.56
4	0.38	0.67
6	0.76	0.98
8	0.92	1.51
10	1.20	1.82
12	1.31	1.98
14	1.42	2.01
16	1.56	2.14
18	1.78	2.37
20	1.92	2.56

Table 6 Performance of Proposed SDN System using Modified CNN Architecture in Terms of Computational Time

The proposed SDN system using modified CNN architecture consumes 0.12 ms for detecting 2 numbers of malicious switches in the simulation environment. The proposed SDN system using conventional CNN architecture consumes 1.92 ms for 20 numbers of malicious switches in the simulation environment. The proposed SDN system using conventional CNN architecture consumes 0.56 ms for detecting 2 numbers of malicious switches in the simulation environment. The proposed SDN system using conventional CNN architecture consumes 2.56 ms for 20 numbers of malicious switches in the simulation environment.

Figure 5 is the graphical illustrations of the proposed system in terms of computational time.

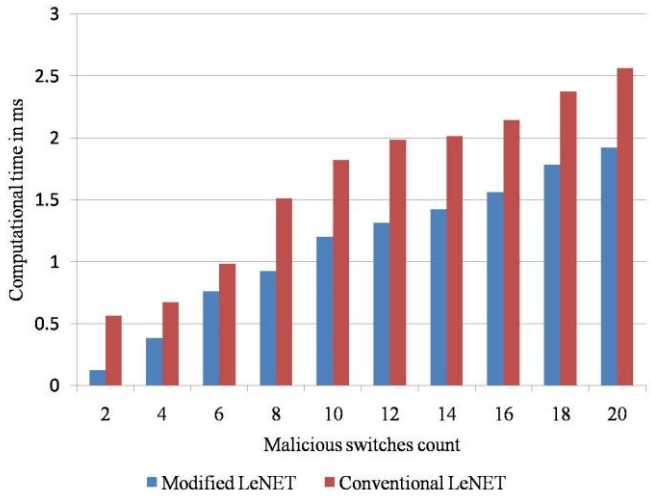


Figure 5 Graphical Illustrations of the Proposed System in Terms of Computational Time

Methodology	Performance Analysis Parameters		
	MSDR (%)	PDR (%)	Computational Time (ms)
Proposed SDN system using modified LeNET CNN architecture	95	81.9	1.92
Sebbar et al. (2020)	90	75.1	3.78
Chi et al. (2020)	88	73.3	3.62

Table 7 Comparisons of Proposed SDN System

Table 7 shows the comparisons of proposed SDN system using deep learning methodology and conventional methodologies. The proposed SDN system using modified LeNET CNN architecture achieves 95% of MSDR, 81.9% of PDR and consumes 1.92 ms of computational time. Sebbar et

RESEARCH ARTICLE

al. (2020) achieved 90% of MSDR, 75.1% of PDR and consumes 3.78 ms of computational time. Chi et al. (2020) achieved 88% of MSDR, 73.3% of PDR and consumes 3.62 ms of computational time.

From Table 7, it is clearly observed that the proposed SDN system using modified deep learning architecture stated in this article provides superior performance results when compared with other conventional methods. Figure 6 shows the graphical comparisons of proposed SDN system with conventional methodologies.

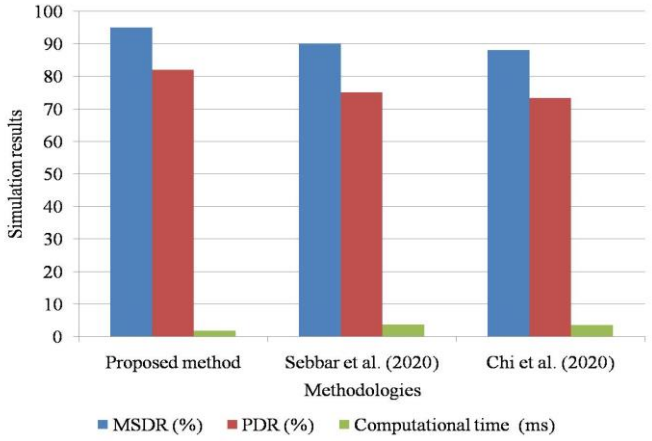


Figure 6 Graphical Comparisons of Proposed SDN System with Conventional Methodologies

6. CONCLUSIONS

This paper presented the improved performance of SDN system in view of detection of malicious switches using modified deep learning architecture. The external feature maps are computed from the networking switch device and these external feature maps are trained and classified using CNN architecture. The proposed SDN system using modified CNN architecture obtains 93.1% of PDR for 2 numbers and 70.7% of PDR for 20 numbers of malicious switches, respectively, in the simulation environment. In addition, the proposed SDN system consumes 0.12 ms for detecting 2 numbers of malicious switches and consumes 1.92 ms for 20 numbers, respectively. It is observed that the proposed methodology improves the MSDR and PDR rate, as well as reduced the computational time. This paper will be extended to detect the link failure in SDN system using hybrid classification methodology such as integration of machine and deep learning architectures.

REFERENCES

[1] X.-F. Chen, and S.Z.Yu, "CIPA: Collaborative intrusion prevention architecture for programmable network and SDN,"Comput. Secur. Vol. 58, No.1, 2016, pp. 1-19.
[2] T. Das, V. Sridharan, and M. Gurusamy, "A survey on controller placement in sdn. ieee communications surveys and tutorials," Vol. 22, No. 1, 2020, pp. 472-503.
[3] E. Vasilomanolakis, S. Karuppayah, M. Muhlhauser, and Mathias Fischer, "Taxonomy and survey of collaborative intrusion detection," ACM Comput. Surv. Vol. 47, No. 4, 2015, pp. 1-10.
[4] C.J. Fung and R. Boutaba, "Design and management of collaborative intrusion detection networks," IFIP/IEEE International Symposium on Integrated Network Management (IM), Vol.1, No.1, 2013, pp. 955-961.
[5] S. Hameed, and H.A. Khan, "SDN based collaborative scheme for mitigation of DDOS attacks," Future Internet, Vol. 10, No. 3, 2018, pp. 281-288.
[6] Z. Ma, L. Liu, and W. Meng, "Towards multiple-mix-attack detection via consensus-based trust management in IOT networks," Comput. Secur., Vol.1, N0.1, 2020, pp. 12-17.
[7] Y. Meng, "The practice on using machine learning for network anomaly intrusion detection," IEEE International Conference on Machine Learning and Cybernetics, Vol.1, No.1, 2011, pp. 576-581.
[8] W. Meng, W. Li, Y. Xiang and K.-K.R. Choo., "A bayesian inference-based detection mechanism to defend medical smartphone networks against insider attacks," Journal of Network and Computer Applications, Vol. 78, No.3, 2017, pp. 162-169.
[9] R. Gupta, and S. Rajan, "Comparative analysis of convolution neural network models for continuous Indian sign language classification", Procedia Computer Science, Vol. 171, No.2, 2020, 1542–1550.
[10] D. Chasaki and C. Mansour, "Detecting malicious hosts in SDN through system call learning," IEEE Conference on Computer Communications Workshops, 2021, pp. 1-6.
[11] M. Amanowicz and D. Jankowski, "Detection and classification of malicious flows in software-defined networks using data mining techniques," Sensors, Vol.21, No.1, 2021, pp.1-24.
[12] A. Derhab, M. Guerroumi, M. Belaoued, and O. Cheikhrouhou, "BMC-SDN: blockchain-based multi controller architecture for secure software-defined networks," Wireless Communications and Mobile Computing, Vol. 2021, No. 9984666, 2021, pp.1-15.
[13] F.N. Nife, and Z. Kotulski, "Application-aware firewall mechanism for software defined networks," J. Network Syst Manage, Vol. 28, No.1,2020, pp. 605–626.
[14] A. Sebbar, K. ZKIK, and Y. Baddi, "MitM detection and defense mechanism CBNA-RF based on machine learning for large-scale SDN context," Journal of Ambient Intell Human Comput, Vol.11, No.7, 2020, pp. 5875–5894.
[15] P. W. Chi, M. H. Wang, and Y. Zheng, "Sandbox Net: An online malicious SDN application detection framework for SDN networking," International Computer Symposium (ICS), Vol.1, No.1, 2020, pp. 397-402.
[16] C.V. Neu, C. Tatsch, R.C. Lunardi, R.A. Michelin, A.M. Orozco, and A.F. Zorzo, "Lightweight IPS for port scan in open flow SDN networks," IEEE/IFIP Network Operations and Manag. Symposium, Taipei, Taiwan, 2018, pp. 1–6.
[17] Y. Chang, and T. Lin, "Cloud-clustered firewall with distributed SDN devices," IEEE Wireless Communications and Networking Conference (WCNC), Barcelona, Vol.1, No.1, 2018, pp. 1–5, 2018.

Authors

Mr. Thangaraj Ethilu, received the M. Tech in Computer Science and Engineering at Dr. M.G.R Educational and Research Institute University in Chennai and B. E degree in Computer Science and Engineering at Madurai Kamaraj University in Madurai. Presently he is pursuing the Ph.D. in Department of Computer Science and Engineering, Annamalai University Chidambaram, Tamil Nadu India. His area of interest is networking and cloud computing.

RESEARCH ARTICLE

Dr. Abirami Sathappan, received the M. E in Computer Science and Engineering at Annamalai University in Chidambaram and Ph.D. in Computer Science and Engineering at Annamalai University in Chidambaram. Presently she is working as an Assistant Professor in Computer Science and Engineering, Annamalai University Chidambaram, Tamil Nadu India. Her area of interest is Image Processing.

Dr. Paul Rodrigues, received the M. E in Computer Science and Engineering at Mothilal Nehru Regional Engineering College in Allahabad and Ph.D. in Computer Science and Engineering at Pondicherry University in Pondicherry. Presently he is working as a Professor in Computer Science and Engineering, DMI College of Engineering, Chennai, Tamil Nadu India. His area of interest is networking.

How to cite this article:

Thangaraj Ethilu, Abirami Sathappan, Paul Rodrigues, "Modified Deep Learning Methodology Based Malicious Intrusion Detection System in Software Defined Networking", International Journal of Computer Networks and Applications (IJCNA), 8(4), PP: 381-389, 2021, DOI: 10.22247/ijcna/2021/209704.