Study the Effectiveness of Sequential Probability Ratio Test in detection DDoS Attacks against SDN

Basheer Husham Ali
Department of Computer Engineering, AL-Iraqia University, Baghdad, Iraq
Al-mafrachi.2@wright.edu

Abstract

In traditional networks, switches and routers are very expensive, complex, and inflexible because forwarding and handling of packets are in the same device. However, Software Defined Networking (SDN) makes networks design more flexible, cheaper, and programmable because it separates the control plane from the data plane. SDN gives administrators of networks more flexibility to handle the whole network by using one device which is the controller. Unfortunately, SDN faces a lot of security problems that may severely affect the network operations if not properly addressed. Controllers of SDN and their communications may be subjected to different types of attacks. DDoS attacks on the SDN controller can bring the network down. In this research, we studied effectiveness of sequential probability ratio method in identifying the compromised switched interface and detecting Distributed Denial of services (DDoS) attacks that are targeted the controller of Software Defined Network (SDN). We implemented the detection method and evaluated the performance of the method using publicly available DARPA datasets. Finally, we found that SPRT has the highest accuracy and F score and detect almost all DDoS attacks without producing false positive and false negative.

Keywords: Sequential Probability Ratio, detection DDoS Attacks, SDN**.**

1. Introduction

In traditional networks, traffic flows are transferring through networking devices such as routers and switches that are distributed around the world. Networking devices are responsible to control and forward traffics. Although these traditional networks are widespread and popular, they have several drawbacks. First, they do not provide flexibility to researchers to do their experiments and add new features or protocols [1], [2]. Second, traditional networks are not programmable, so they cannot accept new commands to improve their functionality. Third, the cost of networking devices is very high because each device contains both the control and data plane [3].

However, Software Defined Networking (SDN) fixes the problems of traditional network. SDN is a programmable and virtualized network that helps researches to insert their new ideas. SDN separates the control plane from the data plane. The control plane is responsible for handling information whereas the data plane is responsible for forwarding data. By using SDN, researchers can do their own experiment in network without disturbing other people who depend on it. Multiple network devices can be managed and configured by using single device which is the control plane [4]. This may lead to reduce the time of recovery when errors happened. Finally, SDN is cheaper than traditional networks [3], [5].

Because the SDN infrastructure is more flexible, programmable, and simpler than the traditional networks, it can be deployed in many different types of networks such as private networks, enterprise networks, and wide area networks [6]. Unfortunately, SDN has many challenges that need to be addressed. Scalability, performance, and security are some of the challenges that face SDN.

There are many kinds of threat vectors that have been determined in SDN [8]. Some of these threats target main components of SDN such as the control plane, the data plane, or application. Other threats target communication among these components. The most dangerous threat attacks the control plane component and the communication between this component and others. These threats would be done by exploiting the

vulnerabilities or bugs that exist in the controller or communication protocols. Attackers would be able to control the whole network if they can successfully attack the control plane. Controllers of SDN and their communications are subjected to different types of attacks. The most dangerous one is DDoS attacks because research shows that the controller is a vulnerable target of DDoS attacks such as [1], [9], [6], [10], and [11]. If the controller is brought down, the whole network will be stopped. The paper organizes as the following: SDN architecture explained in section 2. The methodology mentioned in section 3. Finally, section 4 and 5 specified for results and conclusion.

2. SDN Architecture

SDN consists of three main components which are application (application layer), the control plane (control layer), and the data plane (infrastructure layer). Application locates in the upper side, and it contains multiple application logic and Northbound Interfaces (NBIs). The control plane exists in the middle, and it contains NBIs, the control logic, and Control-Data-Plane-Interfaces (CDPIs). Finally, the data plane locates in the bottom of this design, and it contains multiple CDPIs and forwarding engines as illustrated in figure 1.

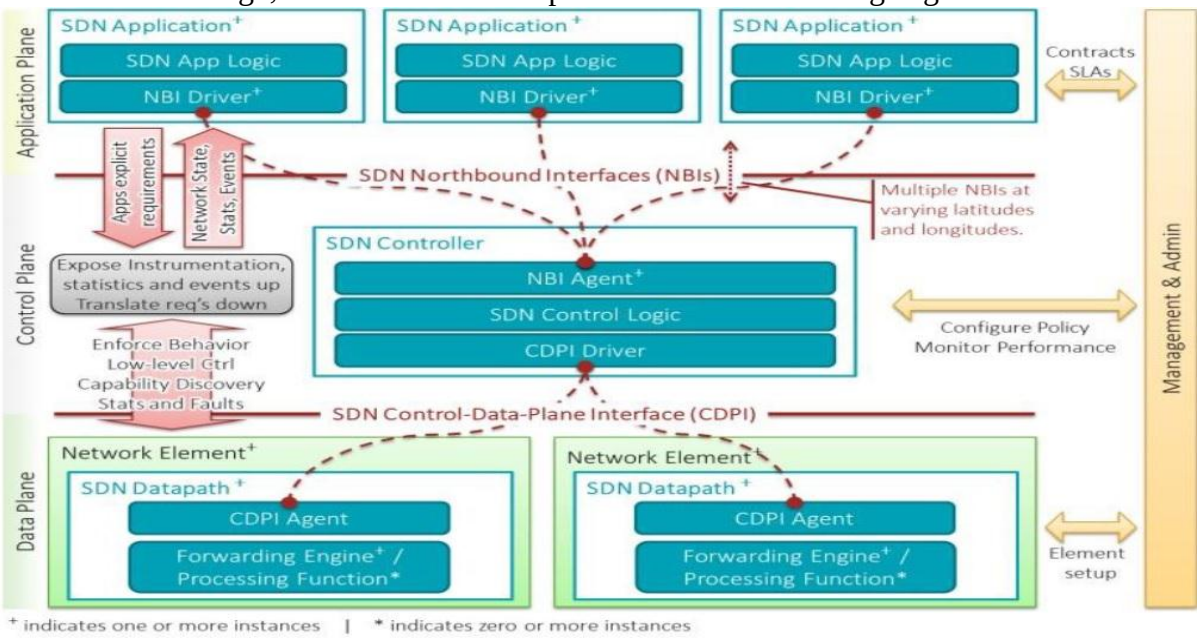

Figure 1: SDN components with management [12]

The NBIs help application plane to communicate with the control plane. Application send down their network requirements to the controller while the control plane send up its desired network behavior, statistics, and events to provide application with abstract view of the whole networks. However, the southbound interfaces or (CDPIs) help network elements that exist in the infrastructure plane to communicate with the control plane. The data plane transfers its statistics, reports, events, and notifications up to the control plane. The control plane sends down its network requirements to the network elements that exist in the data plane, and the data plane obeys rules of control plane [12].

In the right side of the design as shown in figure 1, management and admin component are responsible for providing static tasks to all planes that include the control plane, the data plane, and application. The services agreement and contracts (SLAs) will be configured in the last component which is application plane [12]. Finally, this design also has several agents and coordinators that are spread in the data plane and control plane. These agents and coordinators are responsible to set up the isolation and sharing configuration between the data plane and control plane [7].

The SDN also faces the DoS and/or DDoS attacks. The DDoS attacks happen when compromised host targets single system by sending flood of unnecessary traffics (large number of new low-rate packets). The main goal of this attack is to decrease system availability and prevent legitimate users from accessing available services. If

attackers use many hosts instead of only one to target single system which is the controller, this is called DDoS attacks. DDoS has harmful consequences on the controller of the SDN. For example, businessmen who are offering online services can loss large amount of money if attackers can carry out DDoS successfully against these services [13]. Portsweep, neptune, and smurf are some kind of the DDoS attacks that are used to implement the method.

3. Methodology

This section explains flow classifications in the first part. In the second part, it depicts algorithms that are used to identify the compromised switch interfaces and detect DDoS attacks against the controller in the SDN.

3.1 Flow Classification

Flows are sequence of packets that share same characteristics. These characteristics could be (source IP address, destination IP address, source port number, destination port number, and/or protocol type). All of these information can be extracted from header of each packet. Flows of TCP and UDP based protocols might be these five tuples. However, flows of ICMP protocol could be grouping all packets that have same source IP address, destination IP address, and protocol type because ICMP packets do not have port numbers in their header.

The main aim of classification is to identify DDoS attacks by classifying these flows to either low-traffic flows (malicious flows) or normal flows. Let consider (F_o^i), where (o) is a sequence observations of different flows (F) that injected an interface (i) of the SDN switch. (F_o^i) is low flow if total number of packets within this flow is lower than or equal to certain threshold. However, (F_o^i) is normal flow if total number of packets within this flow is larger than that threshold. The (F_o^i) can be defined as follow [6]:

$$(F_o^i)=\begin{cases}1, & \text{if number of packets} \le \text{Threshold} \\ 0, & \text{if number of packets} > \text{Threshold}\end{cases} \quad (1)$$

3.2 SPRT Detection

SPRT is the first algorithm that was developed by Wald, and it is a specific sequential hypothesis test based on mathematical calculation [14]. It uses two hypothesizes which are H0 and H1. H0 means that interface is normal whereas H1 means that interface of switch is compromised. The compromised interface (H1) is injected by large number of low-traffic flows whereas normal interface is injected by large number of normal flows.

In reality, detection process produces two types of error which are false positive and false negative. False positive error is benign interfaces (H0) that are falsely identified as compromised interfaces (H1). False negative error is the compromised interfaces (H1) that are falsely identified as benign interfaces (H0). To avoid these two types of errors, value of false positive error should not exceed a specified value of (α), and value of false negative error should not exceed a specified value of (β).

In [6], SPRT was used to decide whether the interface (i) is compromised or not by considering a sequence of (n) which is observation of normal and compromise flows (F_o^i) where (o) is the series of observation (o=1,2,3,…,n). These sequences of flows observation are obtained from the first stage which is flow classification. According to SPRT method, (D_n^i) is a detection function that can be defined as a log-likelihood ratio of (n) flows observation, whether they are normal flow or low-traffic flow, for certain interface (i).

Now, the value of D_n^i compares each time with the upper threshold (A) and lower threshold (B). If value of D_n^i is smaller or equal to (B), then the interface (i) is H0 and terminate the test. If value of D_n^i is larger or equal to (A), then the interface (i) is H1 and terminate test. Otherwise, monitor will continue with additional observation. The value of (A) and (B) can be calculated as shown in equation (2) [6], [14]:

$$\begin{cases} A=\ln\frac{\beta}{(1-\alpha)} \\ B=\ln\frac{(1-\beta)}{\alpha} \end{cases} \quad (2)$$

4. Results

We used (07/03/1998) dataset that is available in [15] to evaluate this method. It has almost one million packets as shown in table 1.

4.1 Flow Classification Result:

These packets can be grouped to (256055) flows. The datasets that were captured during 1998 has one router that has one interface which is "00:00:0C:04:41:BC". Therefore, we need to get only these flows that are injected to this interface, which is considered as SDN switch in our case. We got (250551) flows that have this MAC address as destination in their first packet as shown in figure 2.

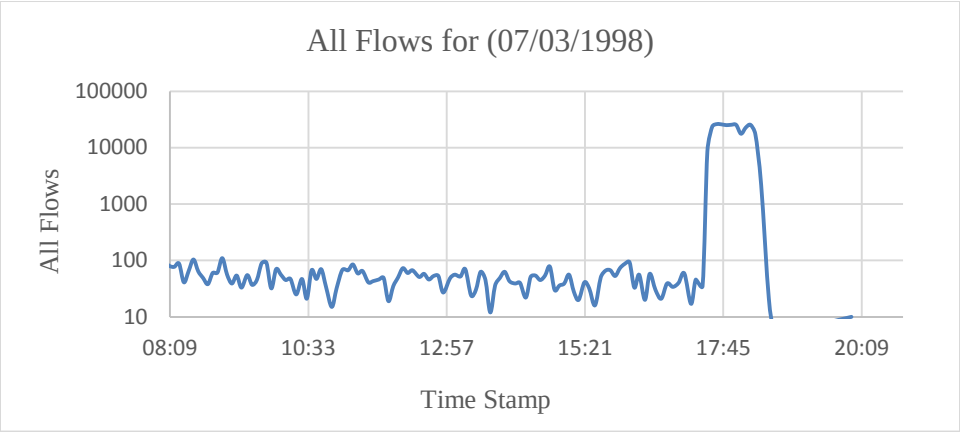

Figure 2: All Flows that have 00:00:0C:04:41:BC as Destination Address in the First Packet of each Flow for (07/03/1998) Dataset

Table 1: Statistics of Classification Flows phase for 1998 Dataset

Dataset name	Number of all packets in each dataset	Number of all Flows	Number of all flows that have 00:00:0C:04:41:BC as destination in the first packet of each flow	Number of low flows that have 00:00:0C:04:41:BC as destination in the first packet of each flow	Number of normal flows that have 00:10:7B:38:46:32 as destination in the first packet of each flow
07/03/1998	1194920	256055	250551	243730	6821

There are many new low-traffic flows starting to occur at different time stamp for this dataset as mentioned in DARPA website and shown in figure 3. First, portsweep attack generated low flow starting at 11:46:39. Another attack which is neptune also produced many new flows at 17:27:07. Finally, at 18:00:15, smurf attack has started to occur as shown in figure below.

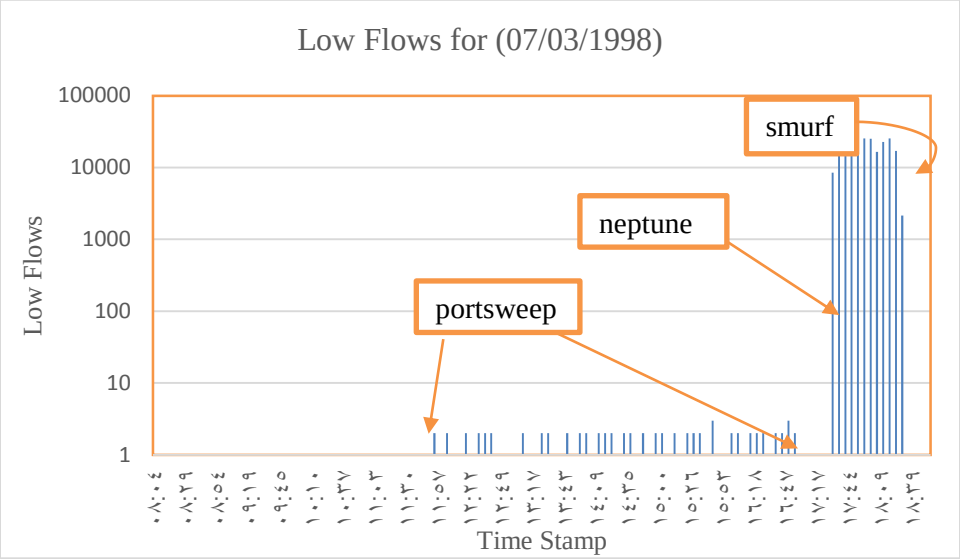

Figure 3: Low-Traffic Flows that have 00:00:0C:04:41:BC as Destination Address in the First Packet of each Flow for (07/03/1998) Dataset

4.2 Results of Detection Algorithm

For this dataset, there are three attacks which are “portsweep”, “neptune”, and “smurf” that were started to occur at “11:46:39”, “17:27:07”, and “18:00:15” respectively as shown in figure 3. SPRT detected “neptune” and “smurf” attack because they were generated many new low-traffic flows. SPRT can decide that interface is compromised after observing 6 continuous low-traffic flows which are considered to be as a minimum number of observations whereas the maximum number of observations was 62. However, the minimum number of normal flows that are required to decide that switch interface is normal in this dataset is 10 whereas maximum number is 40.

On other hands, SPRT method was not able to detect “portsweep” attack although this attack produced low-traffic flows. This attack produces one low-traffic flow that was coming from a port which is 1234 toward a port from a group of 1-100 port every three minutes. This is like a DoS attack when an attacker attacks multiple machines from single machine. There are many flows were generated every three minutes, and most of them were normal flows. SPRT was taking its decision based on these normal flows. Thus, SPRT produces false negative and fails to detect attacks when low-traffic flows are distributed over long time periods.

4.3 Evaluation of Detection Method by using Confusion Matrix

We used confusion matrix that is mentioned in [16] to evaluate and compare all detection method. This matrix depends on four main components which are True Negative (TN), False Positive (FP), False Negative (FN), and True positive (TP). From these elements, many metrics can be computed as shown in Table 2. First of all, we calculated TPR, FPR, TNR, FNR, PPV, FDR, FOR, and NPV. These values were between 0 and 1. The values of TPR, TNR, PPV, and NPV for SPRT method were very closed to 1 and results of FPR, FNR, FOR, and FDR for this detection method were very closed to 0 as shown in Table below. This means this detection method is good. SPRT had also 99% of accuracy and F1 score. Finally, the value of prevalence metric was closed to 1, and this is we would like to get.

		True condition			
	Total population = 250551	Condition positive= 243730	Condition negative= 6821	Prevalence= 0.97277	Accuracy = 0.99746
Predicted condition	Predicted condition positive= 243901	True positive= 243498	False positive= 403	Positive predictive value(PPV), Precision = 0.9983	False discovery rate= 0.00165
	Predicted condition negative = 6650	False negative= 232	True negative= 6418	False omission rate = 0.03488	Negative predictive value= 0.9651
		True positive rate(TPR), Recall Sensitivity= 0.99904	False positive rate (FPR), Fall-out= 0.05908	Positive likelihood ratio (LR+)= 16.909	Diagnostic odds ratio= 16714
		False negative rate (FNR), Miss rate= 9.51872E-4	True negative rate (TNR), Specificity (SPC) = 0.94091	Negative likelihood ratio (LR−)= 0.00101	F1 score = 0.9986

Table 2: Confusion Matrix Results

5. Conclusion

The SDN makes networks design more flexible, cheaper, and programmable because it separates the control plane from the data plane. The SDN gives administrators of networks more flexibility to handle the whole network by using one device which is the controller. Unfortunately, the controller of the SDN faces the dangers of DDoS attacks. Attackers trigger their switches to generate large number of new low-rate packets toward controller. DDoS attacks can reduce system availability and bring the network down.

We conducted a study to discover the effectiveness of SPRT method in detection DDoS attacks against controller of SDN and identifying compromised switch interfaces. Because attackers generated new and low-traffic flows, flows were classified to either low-traffic flows or normal flows. Results of classification were as an input for SPRT method. DARPA dataset were used to evaluate the method. We used confusion matrix to evaluate the method. Finally, we found that this method had 99% of accuracy and F1 score.

REFERENCES

[1] B. Raghavan et al., "Software-defined internet architecture: Decoupling architecture from infrastructure," Proc. 11th ACM Workshop Hot Topics Netw., p. 43–48, 2012.
[2] D. Kreutz et al., "Software-Defined Networking: A Comprehensive Survey," Proceedings of the IEEE, vol. 103, no. 1, pp. 14-76, 2015.
[3] S. Sakir et al., "Are we ready for SDN? Implementation challenges for software-defined networks," IEEE Communications Magazine, pp. 36-43, 2013.

[4] H. Kim and N. Feamster, "Improving network management with software defined networking," IEEE Communications Magazine, vol. 51, no. 2, pp. 114-119, 2013.
[5] N. Zhang, H. Hämmäinen and H. Flinck, "Cost efficiency of SDN-enabled service function chaining," info, vol. 18, no. 5, pp. 45-55, 2016.
[6] D. Ping et al. , "A detection method for a novel DDoS attack against SDN controllers by vast new low-traffic flows," IEEE International Conference on Communications (ICC), 2016.
[7] "SDN architecture issue 1," Open Networking Foundation, pp. 1-68, 2014.
[8] D. Kreutz, F. Ramos and P. Veríssimo, "Towards Secure and Dependable Software Defined Networks," Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking. ACM, pp. 55-60, 2013.
[9] C. Kuan-yin et al, "SDNShield: Towards More Comprehensive Defense against DDoS Attacks on SDN Control Plane," 2016 IEEE Conference on Communications and Network Security (CNS), pp. 28-36, 2016.
[10] D. Kotani and Y. Okabe, "A Packet-In Message Filtering Mechanism for Protection of Control Plane in OpenFlow Networks," 2014 ACM/IEEE Symposium on Architectures for Networking and Communications Systems (ANCS) Architectures for Networking and Communications Systems (ANCS), 2014 ACM/IEEE Symposium on, pp. 29-40, 2014.
[11] S. M. Mousavi and M. St-Hilaire, "Early Detection of DDoS Attacks against SDN Controllers," International Conference on Computing, Networking and Communications, Communications and Information Security Symposium, pp. 77-81, 2015.
[12] "SDN Architecture Overview Version 1.0," Open Networking Foundation, pp. 1-5, 2013.
[13] A. Doyal, J. Zhan and H. A. Yu, "Towards Defeating DDoS Attacks," 2012 International Conference on Cyber Security cybersecurity Cyber Security, pp. 209-212, 2012.
[14] A. Wald, Sequential Analysis, New York: John Wiley and Sons, Inc., 1947.
[15] "MIT Lincoln Laboratory," Intrusion detection atttacks database, [Online]. Available: https://ll.mit.edu/ideval/index.html.
[16] “Confussion Matrix,” 7 August 2017. [Online]. Available: https://en.wikipedia.org/wiki/Confusion_matrix. [Accessed 11 November 2017].