International Journal on Advanced Science Engineering Information Technology

Vol.8 (2018) No. 2
ISSN: 2088-5334

Improving DDoS Detection Accuracy Using Six-Sigma in SDN Environment

Achmad Khalif Hakim[#], Maman Abdurohman[*], Fazmah Arif Yulianto[*]

[#]*Telkom Indonesia Corp. Jln. Japati No. 1, Bandung, Indonesia*
E-mail: kholif@telkom.co.id

[*]*School of Computing, Jln. Telekomunikasi No. 1, Bandung, 40257, Indonesia*
E-mail: {abdurohman, fazmaharif}@telkomuniversity.ac.id

***Abstract*— This paper proposes the new method for improving the accuracy of detection of DDoS attacks on the SDN by utilizing control plane using Six-Sigma method. Software-Defined Networking (SDN) is a centralized network control system. This system offers flexibility on receiving, processing and forwarding packets between subnetworks. The centralized system of SDN, which separates control plane and data plan, has an immense number of advantages, but it also has the risk of becoming a single point of network failure. Distributed Denial of Service (DDoS) attack is the major issues faced in the security aspect of SDN. This attack can make network resources unreachable by the real packets. The widely known method has been implemented on SDN for avoiding a DDoS attack is Three-Sigma method. Three-Sigma method uses a threshold value to determine the existence of a DDoS attack. However, this method has drawbacks regarding accuracy in determining the DDoS attack. The main contribution of this paper is utilizing central control plane of SDN for improving accuracy on detecting the DDoS attack. Several experiments performed for proving the concept. The result shows the new method can improve the accuracy of detection of a DDoS attack, either in constant or fluctuating traffic, by reducing the false positive. The performance is about 50% more accurate than the previous method.**

***Keywords*— Software-Defined Networking (SDN); Distributed Denial of Service (DDoS); three-sigma; six-sigma**

I. Introduction

There are several methods for avoiding DDoS attacks on SDN environment [1], [2], [3]. There are two level attacks, network, and application level attacks. Network attacks are usually made by flooding the network by useless packets. The transaction-level attack occurred by activating of applications that consume resources of a computer system [4]. In the traditional / existing Network, to innovate and manage the network is hard due to each device has its control logic and has vendor dependency. The challenge is how to implement new ideas in real traditional networks because new ideas often include nonstandard aspects. This shortage makes it difficult on implementing DDoS detection mechanism

SDN is an approach to computer network system that uses software for control and manages network devices that can be programmed. OpenFlow is secured communication protocol between planes in SDN [5]. This research has explored the implementation of Software-Defined Networking (SDN) with OpenFlow protocol for detecting network-level attacks.

Based on the theoretical and conceptual of the SDN and the statistical approach several points can be enhanced by the accuracy of the DDoS detection in SDN network. In a traditional network, there is a limitation of DDoS detection, such as adding many of third-party tools in the network, depending on the topology. The previous method of detecting DDoS in SDN is using three-sigma threshold [1], that is too low for defined as a threshold, so while suddenly there is an increase of legitimate traffic it will be detected as a DDoS.

The sigma or deviation can be used as detection threshold. The sigma is a standard deviation of the list of value. It can be used as a reference standard or limit, how much traffic may be distorted. Because sigma is the standard deviation value, it is not the value of traffic that will be compared with a standard deviation of itself, but the difference between the current traffic by an average of previous traffic history. The use of sigma for threshold can use the general equation for finding deviation.

In other research, there is a method using Six-Sigma. The three-sigma normal distribution output falls between 99.7%. However, the normal distribution output of Six-Sigma is 99.97%, this normal distribution output can be used for achieving extremely low false positive or negative. By identifying Six-Sigma as threshold values [1], we will use only the Upper Control Limit (UCL) value as an upper threshold because the attack traffic is always expanding the traffic.

The previous method just uses three of standard deviation threshold mechanism from 60 pool of packet flow from switches, which if there is a sudden increase of traffic then it will be detected as a DDoS attack although it is regular traffic. To improve the detection, we propose the modification of the previous method by changing the three-sigma value to Six-Sigma value, Gupta identified that Six-Sigma value has better accuracy and low false positive [1] in a DDoS attack in traditional / existing networks. This paper works in SDN environment which has different characteristic compared to traditional / existing networks. In this research to improve the previous method, we change the threshold of detection to Six-Sigma to lowering the false positive rate.

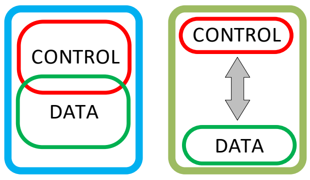
II. Material and Method

A. *Software-Defined Networking (SDN)*

There are various types of attack, and it can be classified base on the characteristics of the effect on the victim. There are software attack, protocol attack and bandwidth attack [6]. This paper focuses on traffic attack solution.

SDN is a new concept in computer network system, it simplifies control of the network and enables innovation through the programmable network, it has a concept of separating the control from forwarding planes. On SDN, control-plane is separated on another machine called controller [7]. This process provides innovative network architecture, programmable, cost-efficient and vendor-agnostic [8].

Fig. 1, (a) and (b) shows how the existing traditional network running specification, which is the control and the data planes are on the same machine. Fig. 2 shows the running specification of SDN OpenFlow where planes are separated in a different machine.



(a) (b)
Fig. 1 (a) or (b) is an existing traditional network [5]

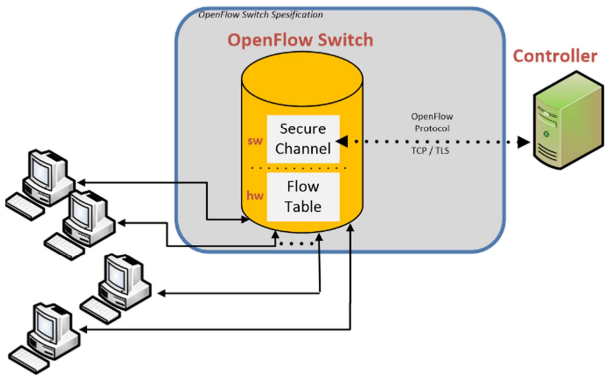


Fig. 2 The specification of SDN OpenFlow [9]

B. *SDN OpenFlow*

OpenFlow is a new protocol that used by the controller to communicate with the switches. This protocol is an essential part of SDN. It controls the switches and external entity to arrange the flow of packets on the network.

Every switch has ingress and egress paths data on the table. OpenFlow makes the controller can access these tables.

Fig. 3 and Fig. 4 show the SDN architecture with OpenFlow protocol [10]. Fig. 5 shows OpenFlow table that stores 12-tuple with fields and every flow has a statistic, a priority, and action [4].

C. *SDN Controller*

SDN controller is an essential device that is responsible for maintaining all of the network rules. It distributes proper instructions for the network devices. The OpenFlow controller is entirely responsible for handling packets [9].

In the traditional / existing Network, to innovate and manage the network is hard due to each device has its control logic and has vendor dependency. It is not easy to experiment new ideas in real traditional networks because new ideas often include nonstandard aspects. To break this limitation, SDN was introduced. SDN enabled controllers can control devices through a secure channel.

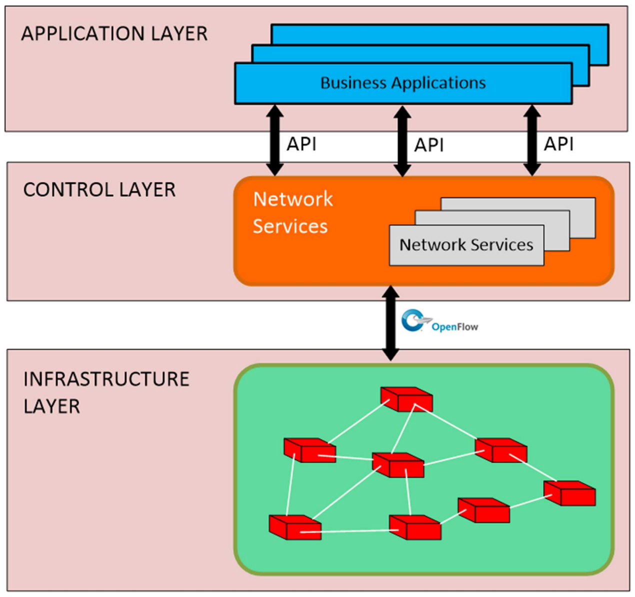


Fig. 3 SDN architecture with OpenFlow protocol [10]

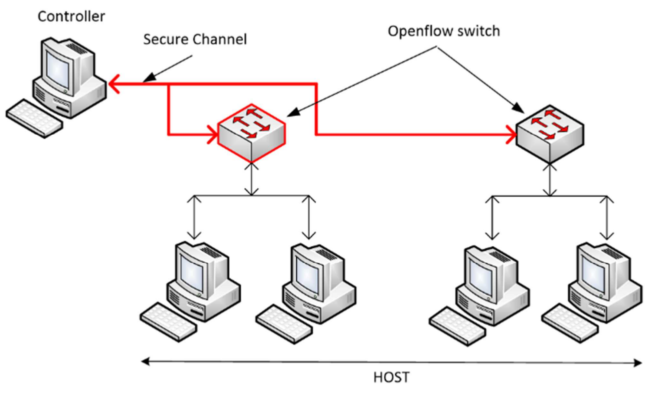


Fig. 4 Simple network using OpenFlow protocol

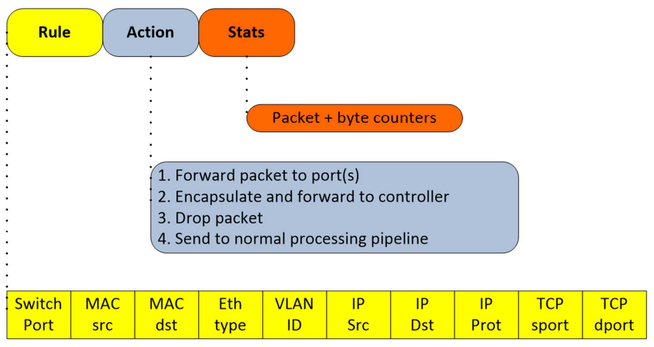


Fig. 5 OpenFlow table entry [4]

Controllers instruct devices to forward packets based on the instructions or flow entries. Users can deploy their new idea on the SDN network with centralized control logic [11]. The controller will communicate to the data plane device using OpenFlow protocol. With this benefit, it is possible to organize OpenFlow Switches [2].

D. Available Methods for Detecting DDoS

There are several methods related to DDoS detection that is based on traffic anomaly. The DDoS detection on traffic anomaly that commonly used is based on statistical approach likewise the use of standard deviation and entropy.

The standard deviation could be used for divining a threshold for DDoS detection; there are several researches of using standard deviation as a DDoS detection threshold. The standard deviation itself sometimes called sigma, referring to the standard deviation symbol (σ).

1. Three-sigma on SDN network: There are some methods on using OpenFlow for detecting malicious activity [12]. It functions to inspect certain traffic classes [13]. The previous method using three-sigma deviation from 60 pool of packet flow from switches, which suddenly there is an increase of traffic that the range is still reasonable from normal behavior, it will be detected as a DDoS, whether it is normal traffic. After 60 pooling T_{60}port stats from the switch, the value of the 3sigma and the mean (μ) are calculated, the three-sigma (3σ) will be defined as a threshold, and the mean value will be kept as for next process. When the 61st (T_N) traffic pooled, it will subtract with the previous mean (μ) value, the result of subtraction will be compared to the 3σ, if it is higher than the 3σ, is known as an attack. The equation can represent this concept.

$$\text{Attack} \Rightarrow (T_N\text{-}\mu(T_{60})) > (3\sigma(T_{60})) \qquad (1)$$

2. Six-Sigma: Six-Sigma, is six standard deviations from the mean, the plan is proposed by Motorola to address quality issue what's more, business change. Six-Sigma signifies an efficient, creative movement to factually gauge and investigate reasons for deformities that happen in all parts of the administration, and at that point evacuate those causes by distinguishing proof of limits of the critical measurements which are measured. Six-Sigma asserts that concentrating on the diminishment of variety will settle process. By utilizing an arrangement of factual instruments to comprehend the variance of a procedure, the administration can start to foresee the normal result of that procedure. [1]. In other research, there is a method using Six-Sigma for DDoS detection in the traditional network. The standard deviation obtained from traffic flow will be multiplied by six and then add or subtract the result for calculating the mean value.

$$\text{UCL} = \mu + 6\sigma \qquad (2)$$

In equations formula 3, UCL is a method that uses the upper limit control mechanism as the threshold. However, by using the mean plus Six-Sigma, there is a potential that the threshold is too high. Because the sigma is a deviation, so the difference of the mean should compare it. Also, there is no literature showing the use of Six-Sigma on SDN network. Table 1 shows the difference between two methods.

TABLE I
DIFFERENTIATION BETWEEN TWO METHODS

Method	Used in SDN	Normally Distributed Output
Three-sigma	Yes	99.7%
Six-Sigma	No	99.97%

The three-sigma, have a normally distributed output of 99.7%. Otherwise, Six-Sigma have the normally distributed output is 99.97% [1], this normally distributed output can be used for achieving extremely low false positive, this method used in the traditional network. On the other hand, there is the use of standard deviation as DDoS detecting threshold in SDN, but it just used the three-sigma.

E. Method for the Proposed System

We have proposed the new scheme using Six-Sigma because the three-sigma normally distributed output fall between 99.7%. Otherwise, Six-Sigma have the normally distributed output is 99.97% [1], this normally distributed output can be used for achieving extremely low false positive, by identifying Six-Sigma as threshold values [1] we will use only the UCL value as an upper threshold because the attack traffic always up warding the traffic.

In comparing mechanism, we choose the previous method comparing subtracted the "now traffic" with the mean of 60 pool of traffic, and compare it with the standard deviation that multiplies by 6 (Six-Sigma) for defining a threshold. The Six-Sigma is a deviation, so it should be comparing it with the difference value between the "now traffic" and the mean. If the value of "now traffic" minus the mean of 60 pools is higher than the Six-Sigma, it considered as DDoS. Base on the benefit of that method. The standard deviation we get from the traffic flow will be multiplied by 6.

$$\text{Attack} \Rightarrow (\text{TN-}\mu(\text{T60})) > (6\sigma(\text{T60})) \qquad (3)$$

After 60 pooling T_{60}port stats from the switch, the value of the 6sigma and the mean (μ) are calculated, the 6sigma (6σ) will be defined as a threshold, and the mean value will be kept as for next process. When the 61st (T_N) traffic pooled, it will subtract with the previous mean (μ) value, the result of subtraction will be compared to the 6σ, if it is higher than the 6σ, it is known as an attack. The comparison between the previous and the improvement methods is described in Fig. 6. While Fig. 7 shows the flowchart of a proposed method.

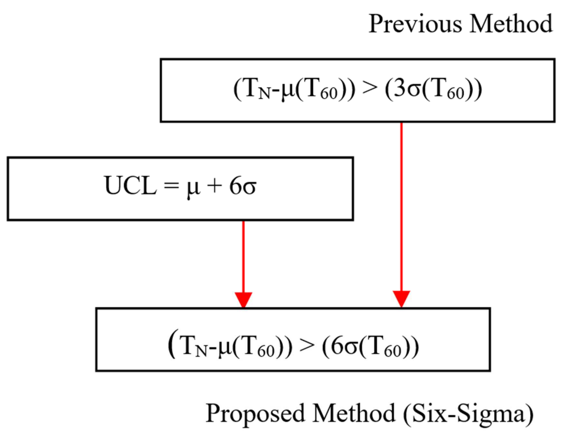

Previous Method
$(T_N-\mu(T_{60})) > (3\sigma(T_{60}))$
$UCL = \mu + 6\sigma$
$(T_N-\mu(T_{60})) > (6\sigma(T_{60}))$
Proposed Method (Six-Sigma)

Fig. 6 Method comparison

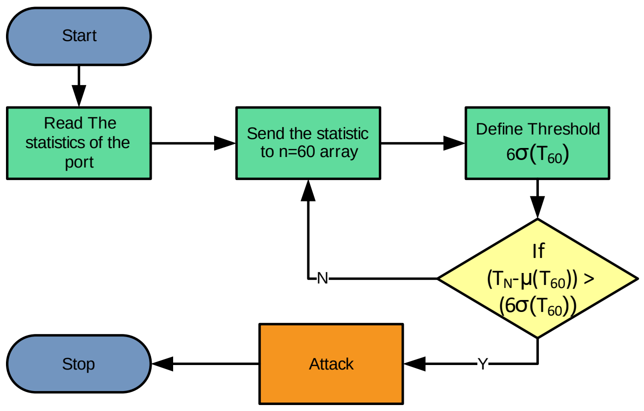

Start
Read The statistics of the port
Send the statistic to n=60 array
Define Threshold $6\sigma(T_{60})$
If $(T_N-\mu(T_{60})) > (6\sigma(T_{60}))$
N
Y
Attack
Stop

Fig. 7 Flowchart of designed method

We proposed the combination of above previous methods, changing the three-sigma from the SDN base DDoS Detection with the Six-Sigma from the traditional network. Fig. 7 shows the complete flowchart of the system as follows:

1) Read Port Statistics: The flow statistics pooled to the controller from the port of the switch, each eth port of the host will pool to the controller.

2) Send Statistics to a Bucket of Array: Every port statistics from the switch will be inserted into a bucket of the array that has size 60 if exceeded, the first entry value of the Bucket of the array will be deleted.

3) Define Threshold: Calculate the threshold of Six-Sigma from the 60 value of a bucket of the array.

4) Triggered by Attack: If the new entry (T_N) subtracted with the mean of a bucket of the array is larger than Six-Sigma of the bucket of the array, then the T_N

III. Results and Discussions

This section describes the detailed scenario that we used to the experiment. The detail of the scenarios is explained below. The experiment has been performed using simulation software tools Mininet framework, with the following network elements: 1 OpenFlow controller (Ryu Controller), 1 OVS Switch (OpenFlow switch), 2 Host, 1 host as an attacker and another host with normal traffic and 1 host as a victim. The topology of the simulation network that is used in this research can be shown in Fig. 8.

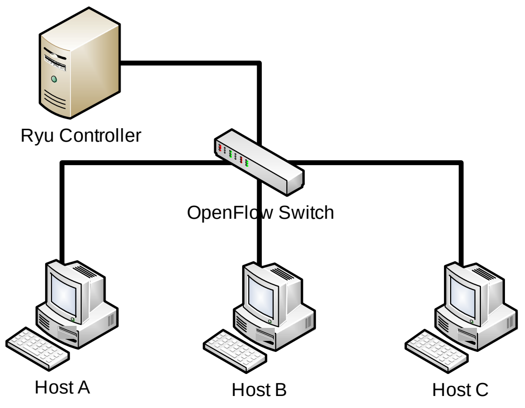


Fig. 8 Topology scenario of simulation

The network topology in this scenario is using 3 hosts such as Host 1, Host 2 and Host 3, 1 OpenFlow Switch, and 1 Controller. Host 2 will act as an attacker, Host 3 as a regular user and Host 1 as a victim. At the first time, Host 3 will send a normal packet to Host 1 and then Host 2 and 3 will send attack packet to Host 1. Ryu controller is selected because it provides distinct API software components. In this experiment, there are two scenarios with representing traffic in Telkom Corp. network to simulate normal traffic with sudden change and DDoS attack traffic. After analysis the Telkom Corp. Mean Router Traffic Gateway (MRTG); it is found that there is two type of traffic in Telkom network, uniform, and un-uniform.

A. Scenario 1

This scenario will test how both methods, previous method, and improved method, to detect normal traffic with a suddenly increased traffic, in unflat condition. The attacker sends 20M and the maximum legitimate traffic set to 6M. We choose the 6M value from the highest traffic value 5M added by 1M, just in case if there is an increase in traffic that exceeds the maximum traffic. However, it should be not too high from the average, so we believe the 6M represents the highest traffic on this traffic pattern. The testing is on the first traffic pattern mimicking the real traffic on Telkom ISP user. This testing will try to use 2 attackers. We called this traffic pattern as Traffic A.

B. Scenario 2

In this scenario, previous method and improvement method will be tested to detect attack traffic with the flat condition. The attacker sends 20M, and the maximum legitimate traffic set to 9M. We choose the 9M value from the highest traffic value 8M added by 1M, just in case if there is an increase in traffic that exceeds the maximum traffic. However, it should be not too high from the average, so we believe the 6M represents the highest traffic on this traffic pattern. The testing is on the first traffic pattern mimicking the real traffic. This testing will try to use 2 attackers. We called this traffic pattern as Traffic B.

On implementing experiment scenario, there are three main steps such as creates Mininet simulator environment where all simulation get a place, run controller with detection method in it, and run Iperf to generate IP traffic.

C. *Experiment Result of Scenario 1*

In this scenario, the real traffic data of Telkom Corp. is used, by mimicking them with traffic generator, first let the traffic generator run based on mimicking data traffic that represents active user traffic. In Fig. 9, at second 208, there is injected UDP traffic as an attack. Every second, the controller will pool 60 values to the array and calculate the mean and the 6sigma of it. The controller will always calculate this value and will compare the next value of traffic with the mean and Six-Sigma.

The Mean of 60 pool of traffic from second 147 to 207 is 1.49, the sigma value of the 60 pool traffic is 1.06. The traffic at second 208 is 20m as attack traffic; when it is subtracted from the mean, the value is above the threshold of both Six-Sigma and three-sigma. It is detected as an attack.

It used the real traffic data of Telkom Indonesia Corp., by mimicking them with traffic generator. First, we let the traffic generator run based on mimicking data traffic as active user traffic. In Fig. 10, at second 208, we inject the UDP traffic as legitimate. Every second, the controller will pool 60 values to the array and calculate the mean and the 6sigma of it. The controller will always calculate this value and will compare the next value of traffic with the mean and Six-Sigma.

D. *Experiment Result of Scenario 2*

We use the real traffic data, by mimicking them with traffic generator. First, we let the traffic generator run based on mimicking data traffic as infrequent user traffic, we want to know is the three-sigma, or Six-Sigma can detect an attack on the flat tendencies traffic.

In Fig. 11 at second 208, we inject the UDP traffic as Attack. Every second, the controller will pool 60 values to the array and calculate the mean and the 6sigma of it. The controller will always calculate this value and will compare the next value of traffic with the mean and Six-Sigma.

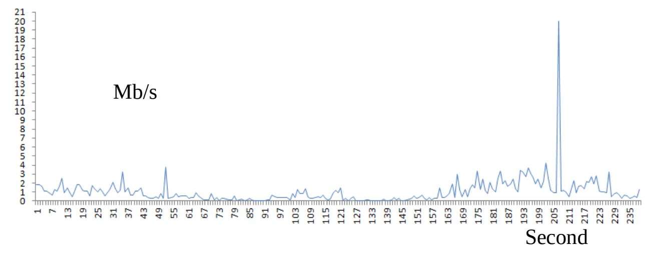


Fig. 9 20M traffic injected on second 208

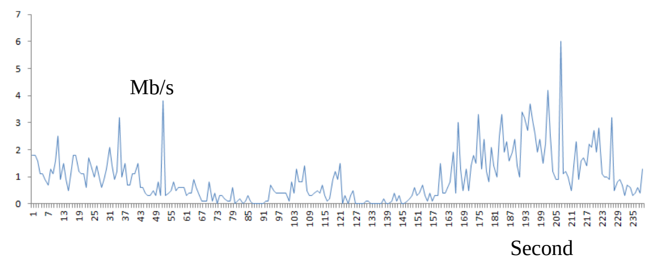


Fig. 10 6M legitimate traffic on second 208

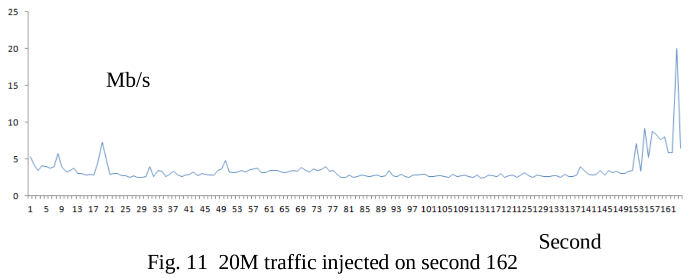


Fig. 11 20M traffic injected on second 162

The Mean of 60 pool of traffic from second 103 to 162 is 3.51, the sigma value of the 60 pool traffic is 1.73. The traffic at second 163 is 20m as attack traffic when it is subtracted from the mean; the value is above the threshold of both Six-Sigma and three-sigma. It is detected as an attack.

We use the real traffic data, by mimicking them with traffic generator. First, we let the traffic generator run based on mimicking data traffic as infrequent user traffic. In Fig. 12, at second 208, we inject the UDP traffic as legitimate. Every second, the controller will pool 60 values to the array and calculate the mean and the 6sigma of it. The controller will always calculate this value and will compare the next value of traffic with the mean and Six-Sigma.

E. *Data Presentation and Analysis*

The Mean of 60 pool of traffic from second 147 to 207 is 1.49 while the sigma value of the 60 pool traffic is 1.06. The traffic at second 208 is 6M as legitimate traffic when it is subtracted from the mean, the value of last traffic is above the threshold of three-sigma, and it is detected as an attack. However, otherwise, it bellows the Six-Sigma threshold, so it is not detected as an attack. To understand this we will see Table 2, the difference value of the last traffic value and the mean present by (Q1-M) column, the 3sigma or 6sigma present by (sigma 60) column. Those values are compared. If the values of the (Q1-M) are bigger than the Sigma (60), it categorized as DDoS.

The Mean of 60 pool of traffic from second 103 to 162 is 3.51 while the sigma value of the 60 pool traffic is 1.73. The traffic at second 208 is 9m as legitimate traffic when it is subtracted from the mean, the value of the last traffic is above the threshold of three-sigma, and it is detected as an attack. However, otherwise, it is bellowing the Six-Sigma threshold, so it is not detected as an attack. To understand this we will see Table 3, the difference value of the last traffic value and the mean present by (Q1-M) column, the 3sigma or 6sigma present by (sigma 60) column. Those values are compared. If the value of the (Q1-M) is bigger than the sigma (60), it categorized as DDoS.

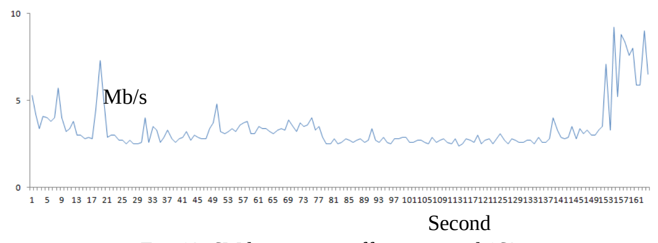


Fig. 12 6M legitimate traffic on second 162

TABLE II
COMPARISON BETWEEN TESTING SCENARIO 1

Sigma	Traffic	Attack	μ(60)	Q1-μ	σ (60)	Result
Previous Method (3sigma)	20M	Yes	1.49	18.51	3.19	Detected
	6M	No	1.49	4.51	3.19	Detected
Proposed Method (6sigma)	20M	Yes	1.49	18.51	6.38	Detected
	6M	No	1.49	4.51	6.38	Not detected

TABLE III
COMPARISON BETWEEN TESTING SCENARIO 2

Sigma	Traffic	Traffic Type	μ(60)	Q1-M	σ (60)	Result
Previous Method (3sigma)	20M	Attack	3.51	16.49	5.19	Detected
	9M	Normal	3.51	5.49	5.19	Detected
Proposed Method (6sigma)	20M	Attack	3.51	16.49	10.38	Detected
	9M	Normal	3.51	5/49	10.38	Not Detected

F. Discussion and Summary Research Findings

In the aspect of simplicity, SDN controller can be programmed with a high-level programming language to implement low-level rules forwarding hardware, to implement the detection mechanism; we only use 2 functions, for the packet statistic pooling and the detection function. In the traditional network, it is tough to program high-level programming language; it only has to use the low-level rules.

For the detection accuracy, Table 2 and Table 3 present the result of a testing scenario. The result is that both three-sigma and Six-Sigma can detect the DDoS Attack. However, the three-sigma also detected the legitimate traffic as anomaly traffic, in that scenario the three-sigma have the false positive rate of 50%, in the Six-Sigma they have 0% of false positive. It is due to the threshold that shaped by three-sigma are too low. On the contrary, the improved Six-Sigma could detect the anomaly traffic and let the legitimate traffic not detected as an anomaly. It is due to the Six-Sigma shaping threshold above the three-sigma shaped threshold. The false positive value that we get shows an extreme percentage due to lack of traffic scenario; it is because the data traffic patterns that we get from the ISP only have 2 types.

IV. CONCLUSIONS

Based on several experiments have been performed, the results show that the use of Six-Sigma as threshold have better accuracy than using three-sigma. Six-Sigma has lower false positive than three-sigma. It is because the three-sigma threshold is too low and the gap of the routine traffic is too narrow, so when there is the high increase of traffic, it will be detected as an attack. The Six-Sigma otherwise, shows much of gap for the legitimate traffic to expand and not detected as an attack. The results show the proposed method could improve the accuracy of DDoS attack detection on SDN environment, either in constant or fluctuating traffic, by reducing the false positive. The performance is about 50% more accurate than the previous method.

ACKNOWLEDGMENT

Authors thank Telkom University under Research and Community Service Bureau (PPM) program for publication incentive. Also, thank Telkom Indonesia Corp. for financial support. Last but not least our colleague in the graduate program of Telkom University.

REFERENCES

[1] B. B. Gupta, Manoj Misra, R. C. Joshi, *An ISP Level Solution to Combat DDoS Attacks using Combined Statistical Based Approach*, 2008.

[2] Mousavi, S.M *"Early Detection of DDoS Attacks in Software Defined Networks Controller."* Carleton University. Canada. https://curve.carleton.ca/system/files/etd/. 2014.

[3] Yadav, A., Radadiya, M., Tilva, M., Rohokale, V. *"SDN Control Plan Security in Cloud Computing Against DDOS Attack."* www.ijariie.com. 2016.

[4] C. Dillon, M. Berkelaar, "OpenFlow (D)DoS Mitigation," 2014

[5] S. Das, G. Parulkar, N. McKeown, "Unifying Packet and Circuit Networks," Below IP Networking (BIPN), November 2009. (S, G, & N, 2009)

[6] Alvaro Garcia de la Villa, Tuomas Aura, Aapo Kalliola, Distributed Denial of Service Attacks defenses and OpenFlow: Implementing denial-of-service defense mechanisms with software-defined networking, 2014.

[7] Saurav Das, Guru Parulkar, Nick McKeown. Unifying Packet and Circuit Switched Networks with OpenFlow. 2009

[8] Siamak Azodolmolky, software-defined network with OpenFlow, 2013

[9] Varun Tiwari, Rushit Parekh, and Vishal Patel. A Survey on Vulnerabilities of OpenFlow Network and its Impact on SDN/OpenFlow Controller. in World Academics Journal of Engineering Sciences 2014

[10] Wolfgang Braun, Michael Menth, Software-Defined Networking Using OpenFlow: Protocols, Applications and Architectural Design Choices, 2014.

[11] Chun-Yu Hsu, Pang-Wei Tsai, Hou-Yi Chou, Mon-Yen Luo, Chu-Sing Yang, 1A Flow-based Method to Measure Traffic Statistics in Software Defined Network, 2014.

[12] S. Akbar Mehdi, J. Khalid, and S. Ali Khayam Revisiting Traffic Anomaly Detection using Software-Defined Networking, 2011

[13] Open Networking Foundation, OpenFlow Switch Speci_cation v1.0, 2009