

6LoWPAN Performance Analysis of IoT Software-Defined-Network-Based Using Mininet-IoT

Dharma Yusuf Setiawan
Adaptive Network Laboratory, School of
Electrical Engineering
Telkom University
Bandung, Indonesia
dharmakemo@gmail.com

Sofia Naning Hertiana
Adaptive Network Laboratory, School of
Electrical Engineering
Telkom University
Bandung, Indonesia
sofiananing@telkomuniversity.ac.id

Ridha Muldina Negara
Adaptive Network Laboratory, School of
Electrical Engineering
Telkom University
Bandung, Indonesia
ridhanegara@telkomuniversity.ac.id

Abstract— Software Defined Network (SDN) is a new paradigm in network architecture. The basic concept of SDN itself is to separate the control plane and forwarding plane explicitly. In the last few years, SDN technology has become one of the exciting topics for researchers, the development of SDN which was carried out, one of which was implementing the Internet of Things (IoT) devices in the SDN network architecture model. Mininet-IoT is developing the Mininet network emulator by adding virtualized IoT devices, 6LoWPAN based on wireless Linux standards, and 802.15.4 wireless simulation drivers. Mininet-IoT expands the Mininet code class by adding or modifying functions in it. This research will discuss the performance of the 6LoWPAN device on the internet of things (IoT) network by applying the SDN paradigm. We use the Mininet-IoT emulator and the Open Network Operating System (ONOS) controller using the internet of things (IoT) IPv6 forwarding. Performance testing by comparing some of the topologies of the addition of host, switch, and cluster. The test results of the two scenarios tested can be concluded; the throughput value obtained has decreased compared to the value of back-traffic traffic. While the packet loss value obtained is on average above 15%. Jitter value, delay, throughput, and packet loss are still in the category of enough, good, and very good based on TIPPHON and ITU-T standards.

Keywords— *Software Defined Network, 6LoWPAN, Internet of Things, Mininet-IoT, ONOS Controller.*

I. INTRODUCTION

Nowadays, humans need tools that can help, and human jobs such as the Internet of Things (IoT), IoT communication networks must support a high amount of traffic and mobility [1]. Traditional IoT communication networks are inefficient and limited to meet the requirements for traffic and high mobility. The number of devices that are connected to the network can cause the network to be overloaded. We build an IoT network with add IoT traffic and its mobility technology. Software-Defined Network (SDN) and Network Function Virtualization (NFV) can be used to realize high traffic volume goals and mobility. SDN provides incorrect network abstraction between control fields in software components and data fields in network devices such as switches and routers [2]. Separation of the control plane and data plane makes network device resources programmable, automation, and network control [3].

In the SDN paradigm, network devices function as light as devices that only evolve data such as switches can side by

side with the control field in the form of software or other terms called controllers. SDN provides network abstraction, which separates the control plane in the software component and the data plane in network devices such as switches and routers [2]. The separation of control plane and data plane makes network equipment resources programmable, automation, and network control [2]. In the SDN paradigm, the function of network devices is replaced by devices that only transmit data like switches that can side by side with the control plane in the form of software or another term called the controller. Therefore, several researchers made several studies on networks SDN-based IoT. Because of this, several researchers made several studies on networks SDN-based IoT.

Research on [4], [5] concluded that the SDN in IoT communication networks could prioritize bandwidth to minimize wasted bandwidth resources. However, this research cannot represent an IoT network because it used a raspberry device, a simple topology, and has not yet evaluated a more comprehensive QoS performance.

Therefore, this research will discuss the evaluation of QoS performance with a more complex scenario, namely a larger number of topology and hosts. The research of the Development of QoS Provisioning Capable Cost-Effective SDN-based Switch for IoT Communications [5] cannot represent an IoT network because it still uses a raspberry device, a simple topology, and has not yet evaluated broader QoS performance. Therefore, this research will discuss the evaluation of QoS performance with a more complex scenario, namely topology, a larger number, and a larger number of hosts.

This paper contributes to developing an emulation topology with 6LoWPAN, which is using for IoT implementations commonly. Using 6LoWPAN is that it is proven low-cost power consumption and low bandwidth based on Ismaili et al. [6]. Therefore, we are conducted simulation research and performance analysis to create IoT communication networks using simulator software Mininet-IoT and measure QoS parameters such as jitter, packet loss, throughput, and latency.

II. PRELIMENARIES

A. Software Defined Network (SDN)

SDN is a new paradigm in architecture network. SDN's basic concept is to make an explicit separation between the

control plane and the forwarding plane [7]. SDN requires a protocol for communication between the control plane and the forwarding plane; some of the protocols developed are OpenFlow, NETCONF, OVSDB, etc.

In the SDN architecture, the control plane is logically centrally software-based. SDN controller can monitor the whole network. SDN simplifies network design and how network devices operate because network devices only need to execute commands from the SDN controller.

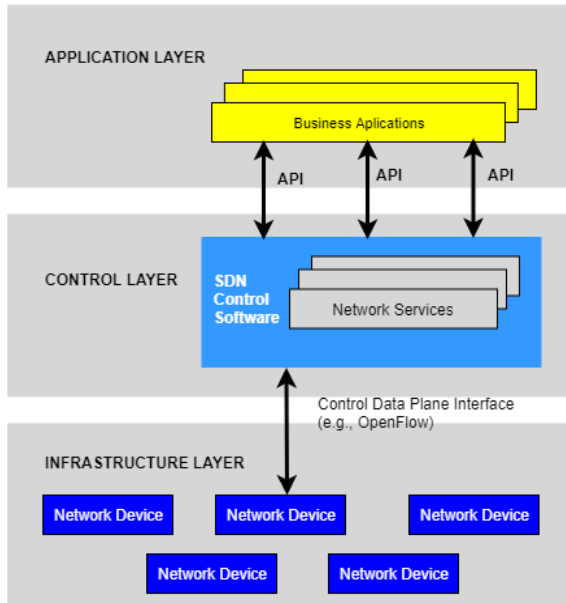


Fig. 1 SDN Architecture

According to Fig. 1, the Open Network Foundation (ONF), the architecture in SDN is categorized into three main layers, namely [7]:

1. Application Layer is an end-user application that utilizes SDN communications and network services such as security, quality of service, traffic engineering. northbound API becomes the barrier between the application layer and the control layer
2. Control Layer is a set of controllers that interact with applications and devices in this architecture. To communicate with infrastructure needs, help southbound API, and communicate with applications need northbound API [7].
3. Infrastructure Layer is consists of SDN devices, both physical and virtual, which perform data packet switching and forwarding. To communicate with the control layer requires the southbound API as an intermediary.

B. Open Network Operating System (ONOS)

ONOS is an open-source controller aimed explicitly at service providers to improve scalability, availability, and high performance [8]. For ONOS, network administration provides a REST API, a command-line interface (CLI), and a graphical interface (GUI) web-based. ONOS is written in the Java programming language for perform maintenance of ONOS applications using Open Service Gateway (OSGi), which can install, stop, start, update and uninstall without requiring a reload. Besides, ONOS provides security by tracking and performing closure to unauthorized access. Turning Transport Layer Security (TLS) and Hypertext Transfer Protocol Secure

(HTTPS) on the northbound interface and the southbound interface can be overcome.

C. Mininet-IoT

Mininet-IoT is developing the Mininet network emulator with added virtualized IoT devices, 6LoWPAN based on standards wireless Linux and 802.15.4 wireless simulation driver. Also, functions are added to support wireless devices in the Mininet network scenario to mimic a mobile station's characteristics, such as position and movement relative to the access point. Mininet-IoT extends the Mininet codebase by adding or modifying functions within it.

D. Internet of Things (IoT)

Internet of Things (IoT) is a network technology in physical devices that allows communication to exchange data between various hardware devices. IoT produces the concept of an internet connectivity network connected continuously, increasing efficiency in interactions between machines and saving more energy [9]. One of the IoT challenges is the availability of power-efficient devices capable of exchanging data routinely in lossy networks. Currently, standard internet protocols are not optimal in meeting these requirements. This challenge will give birth to a variety of interoperability solutions.

E. IEEE 802.15.4

IEEE 802.15.4 is a protocol standard at the physical and media access control (MAC) data links for the WPAN used on the Wireless Sensor Network (WSN). IEEE 802.15.4 sets up wireless interfaces for wireless applications, such as automation, remote sensing, and tracking. IEEE 802.15.4 is a flexible standard because it can be used in various topologies such as ad-hoc mesh, tree, and star. IEEE 802.15.4 is the basis of Zigbee and Wireless networks HART used in WSN [10].

F. 6LoWPAN

6LoWPAN has a meaning IPv6 over Low Power Wireless Personal Area Network. The basic idea is to use IP to connect small, low-power devices within the range of a deep, wireless network normal operation. This sensor's typical power consumption is 70 microA, and power consumption can be reduced to less than 0.3 microA when powered off. Wireless Personal Area Network (WPAN) itself is a network used to connect devices in a specific area where the network is wireless. IPv6 is used to improve this protocol's interoperability to adapt to the network revolution heterogeneous [11]. However, to be implemented in an 802.15.4 network, there are several challenges caused by the limitations of the 802.15.4 network:

1. **Power and Duty Cycle**
One of the keys of 802.15.4 technology is low power consumption and low bandwidth data communication. For this reason, the duty cycle (percentage of active time) is kept low to reduce power consumption, but this is different from IPv6-based internet data communication, which is always connected, continuous packet delivery, and large power consumption.
2. **Multicast**
IEEE 802.15.4 technology does not facilitate multicast and flooding on the network, causing a waste of power

and bandwidth. However, in IPv6, using multicast in some cases.

3. Bandwidth and frame size

IEEE 802.15.4 uses low bandwidth (maximum 250 kbps) and a frame size of 40-200 bytes. In the mesh topology, bandwidth and channel usage are reduced using multihop forwarding. The 802.15.4 standard uses a 127byte frame size with a MAC layer frame size of 72 bytes. However, the minimum frame size for the IPv6 standard is 1280 bytes, so fragmentation is required.

6LoWPAN supports IPv6 with an adaptation layer as defined in the IETF standard RFC6282 to optimize IPv6 over IEEE 802.15.4. The following is a comparison of the Internet Protocol layer with the 6LoWPAN communication protocol [12].

G. Quality of Service(QoS)

Quality of Service (QoS) is a method of measuring how well network and is an attempt to define characteristics and traits of a service. The results of the QoS analysis can be used as recommendations for the physical implementation of the internet network [13].

Table 1. TIPHON standard

Delay Category	Delay (ms)	Index
Very Good	<150	4
Good	150 - 300	3
Average	300 - 450	2
Bad	>450	1
Jitter Category	Jitter (ms)	Index
Very Good	0	4
Good	0 - 75	3
Average	75 - 125	2
Bad	>125 - 255	1
Packet Loss Category	Packet Loss (%)	Index
Very Good	0	4
Good	3 - 15	3
Average	15 - 25	2
Bad	25 - 100	1
Throughput Category	Throughput (%)	Index
Very Good	100	4
Good	75	3
Average	50	2
Bad	<25	1

Table 1 shows the Tiphon standards used in the analysis process. Throughput is the effective data transfer rate, measured in bps (bits per second). Throughput is measured based on the total number of successful packet arrivals observed at the destination during a specific time interval divided by the duration of that time interval.

Jitter or delay variation caused by variations in the delay time, the processing time of the data, and the time of recasting the packets at the end of the trip jitter. The difference between delay and jitter lies in time delay, jitter has an erratic difference in time delay. The difference due to the ability of different tools to respond to data each time. This difference causes data when crossing the network, the distance between the blocks of information is no longer uniform. This is what may be different from delay, which tends to have a constant delay at all times.

Packet Loss is a parameter that describes a condition that shows the total number of lost packets that can occur due to collisions and congestion in the network. Delay (Latency) is the time it takes data to travel the distance from origin to destination. Delay can be affected by distance, physical media, congestion, or extended processing time.

III. 6LoWPAN NETWORK-BASED SDN TOPOLOGY

A. Create Topology in Mininet IoT

In this work, we target topology 6LoWPAN network with SDN paradigm using emulator Mininet IoT. Mininet-IoT software is run on a computer for convenience in testing and data collection to realize a centralized gateway device. Each edge route or AP has a cluster itself; on the Mininet IoT emulator, the sensor device only functions as a node. In one cluster, several sensors form mesh topology, so in one cluster, all connected, bandwidth 6LoWPAN link depends on the number of sensors and clusters in one scenario testing. Virtual Machine specification for running Mininet IoT is one vCPU, 4 GB RAM, and 56 GB disk. The system uses Ubuntu 18.04 as it is a Long-Term Support version. The kernel used is 5.13 because Mininet-IoT only runs on kernel version above 4.18, which ONOS controller was running in container platform one machine with Mininet-IoT. Figure 1 shows our developed topology 6LoWPAN network with 12 hosts in Mininet-IoT, and we have 24, 36 hosts in this scenario.

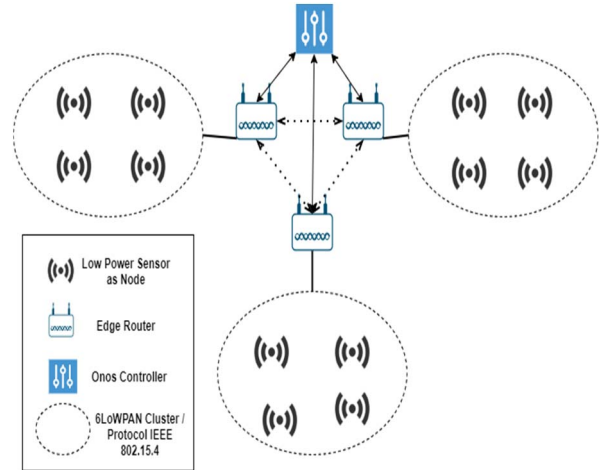


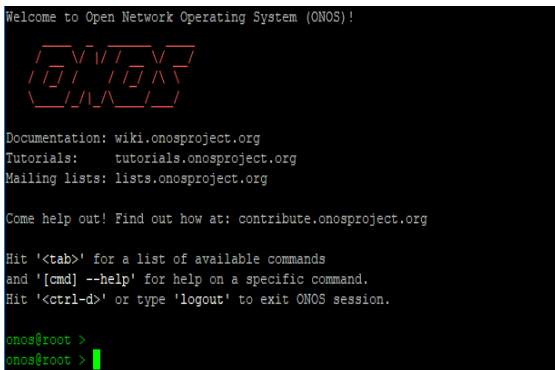
Fig. 2. 6LoWPAN network topology

In this paper, before running the ONOS container, we must prepare the Mininet-IoT emulator. We are using the repository on GitHub with the following command:

```
$ sudo git clone https://github.com/ramonfontes/mininet-  
iot.git  
$ sudo mininet-iot/util/install.sh -a
```

```
# running and activated apps ONOS Container
```

```
$ docker run -p 6653:6653 -p 6640:6640 -p 8181:8181 -p 8101:8101
onos > app activate reactiveforwarding
onos > apps -s -a-p 9876:9876 -d -name onos \> onosproject/onos:2.2.0
```



```
Welcome to Open Network Operating System (ONOS)!

Documentation: wiki.onosproject.org
Tutorials:    tutorials.onosproject.org
Mailing lists: lists.onosproject.org

Come help out! Find out how at: contribute.onosproject.org

Hit '<tab>' for a list of available commands
and '[cmd] --help' for help on a specific command.
Hit '<ctrl-d>' or type 'logout' to exit ONOS session.

onos@root >
onos@root >
```

Fig 3. ONOS CLI

In this research, we are testing and measuring our topology. First, Wireshark is one of the many Network Analyzer tools widely used by network administrators to analyze network performance, including its protocol. Wireshark is well-liked because the interface uses a Graphical User Interface (GUI) or graphic display. Second, Iperf is a tool for measuring the throughput in bandwidth of a network link to measure the Iperf required installed point to point. Both on the server and client-side. Iperf itself can be used to measure the link performance from the TCP and UDP sides.

B. Development 6LoWPAN protocol

In this research, we make the IEEE 802.15.4 protocol in Mininet-IoT, we define nodes and link in python script. IEEE 802.15.4 uses low bandwidth (maximum 250 kbps) and frame size 40-200 bytes. In the mesh topology, bandwidth and usage reduced channels using multihop forwarding. The 802.15.4 standard uses a 127byte frame size with a MAC layer frame size of 72 bytes. However, the minimum frame size for the IPv6 standard is 1280 bytes, so it takes fragmentation. 6LoWPAN supports IPv6 with an adaptation layer as defined in IETF standard RFC6282 to optimize IPv6 over IEEE 802.15.4. The following is a comparison of the Internet Protocol layer with the Communication Protocol 6LoWPAN [12].

C. Scenario Test

In this research, the test scenario used is Realtime Response Under Load (RRUL). This type of test simulates a network on the worst condition by putting a load on the network and then doing data retrieval; the scenario carried out has test parameters jitter, packet loss, throughput, delay, and CPU Usage Host used to run the Mininet-IoT. Which in this experiment uses 100Kbps traffic load.

In this test, we use Iperf for background traffic using a bandwidth of 100 Kbps. The last sensor's traffic as a sender to the primary sensor as a receiver in a particular cluster. There are two scenarios in this research. First, we use scenario 12 Host 3 clusters and 24 hosts 4 clusters. Second, we break down any edge router or cluster head that was previously clustered 1 to 2.

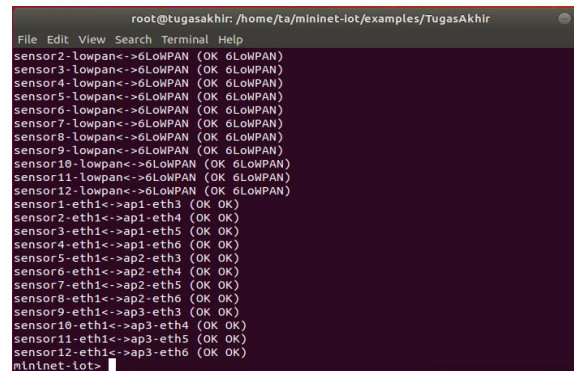
In the test results, the analysis was carried out to determine the performance of the IoT communication network with the SDN paradigm being built and whether it is as effective as possible to compare it to the IoT network without SDN. As

well as identifying the problems that occur when building an SDN IoT network. Then evaluated and a solution to the problem is sought for the next do improvement and development.

IV. EXPERIMENTAL RESULT AND DISCUSSION

A. 6LoWPAN Link Verification Test

In this part, we have practically tested our developed 6LoWPAN network. We set up an experimental testbed link network and device in our topology. However, we have deployed ONOS, which is based on Java language. In our research, Wireshark software is also employed to analyze the system operations. Figure 4 shows the listed link as we designed. It demonstrated that our topology has a 6LoWPAN link and also shows several devices.



```
root@tugasakhir: /home/ta/mininet-iot/examples/TugasAkhir
File Edit View Search Terminal Help
sensor2-lowpan->6LoWPAN (OK 6LoWPAN)
sensor3-lowpan->6LoWPAN (OK 6LoWPAN)
sensor4-lowpan->6LoWPAN (OK 6LoWPAN)
sensor5-lowpan->6LoWPAN (OK 6LoWPAN)
sensor6-lowpan->6LoWPAN (OK 6LoWPAN)
sensor7-lowpan->6LoWPAN (OK 6LoWPAN)
sensor8-lowpan->6LoWPAN (OK 6LoWPAN)
sensor9-lowpan->6LoWPAN (OK 6LoWPAN)
sensor10-lowpan->6LoWPAN (OK 6LoWPAN)
sensor11-lowpan->6LoWPAN (OK 6LoWPAN)
sensor12-lowpan->6LoWPAN (OK 6LoWPAN)
sensor1-eth1->ap1-eth3 (OK OK)
sensor2-eth1->ap1-eth4 (OK OK)
sensor3-eth1->ap1-eth5 (OK OK)
sensor4-eth1->ap1-eth6 (OK OK)
sensor5-eth1->ap2-eth3 (OK OK)
sensor6-eth1->ap2-eth4 (OK OK)
sensor7-eth1->ap2-eth5 (OK OK)
sensor8-eth1->ap2-eth6 (OK OK)
sensor9-eth1->ap3-eth3 (OK OK)
sensor10-eth1->ap3-eth4 (OK OK)
sensor11-eth1->ap3-eth5 (OK OK)
sensor12-eth1->ap3-eth6 (OK OK)
mininet-iot>
```

Fig. 4. 6LoWPAN Link

B. Results

In this research, we have two scenarios: an additional user and an additional cluster. In an additional user scenario, we have 12, 24, and 36 hosts. We discuss the results of tests that have been carried out 30 times by making each experiment's average results. It used to get the appropriate level of accuracy with the expected. After that, we are compared parameters jitter, throughput, packet loss, delay, and CPU usage host

a) *Jitter*: Figure 5 shows the results of the sample Jitter or delay variation. Jitter is directly proportional to both bandwidth and size. The values obtained are not much different and fall into the category good for TIPHON standard reference because of the resulting jitter value <75 ms. It can be seen that the topology with the same number of sensors also has jitter, which is not much different. The number of clusters affects the jitter value by a difference is not 1 to 2 ms.

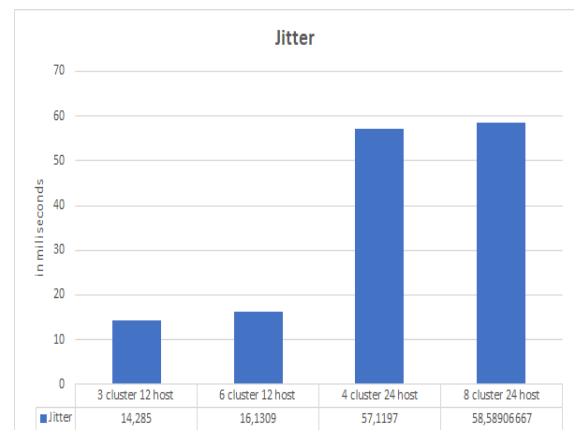


Fig. 5 Jitter's Result.

b) Delay: Delay is the time interval caused by the transmission process from one point to another point that is the goal. Figure 6 shows that the delay value increases along with the addition of sensors; the comparison of the delay value between sensors and clusters is different. The difference between the addition of sensors and clusters is approximately 20 ms to 29 ms. Based on the TIPHON standard, the resulting delay value in figure 6 falls into the good category.

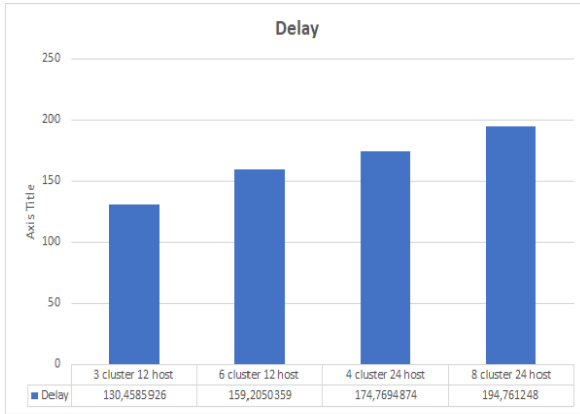


Fig. 6. Delay's result.

c) Packet Loss: Packet Loss is a packet wasted from source to destination due to congestion and data collisions. Figure 7 shows that the packet loss value is getting increases with the addition of sensors. The comparison of packet loss values between the addition of clusters and the addition of sensors is not much different. In the study conducted by Rachael M et al, packet loss values were not more than 1% for the number of sensors 10 to 30 [12]. Figure 7 shows that the packet loss value is obtained into the good category because more than 1% is based on TIPHON standards and ITU-T.

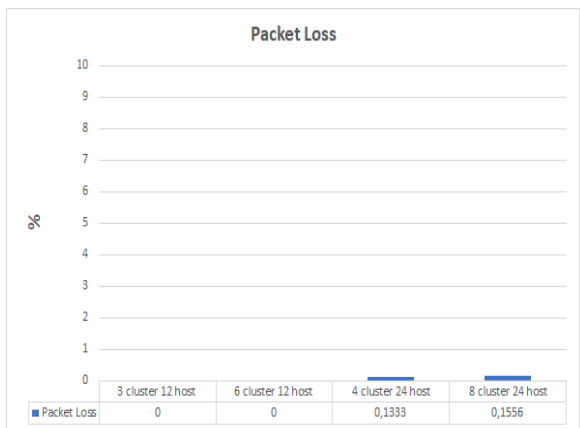


Fig 7. Packet Loss's result.

d) Throughput: Throughput is the effective data transfer rate, measured in bps (bits per second). Throughput is the total number of successful packet arrivals observed at the goal over a specified time interval divided by the interval's duration at that time. Figure 8 shows that the throughput value decreases with the addition of sensors. It can be concluded that the value in Figure 8 is entered in bad category because the IEEE 802.15.4 protocol uses low

bandwidth (maximum 250 kbps) depending on the number of sensors and clusters [12].

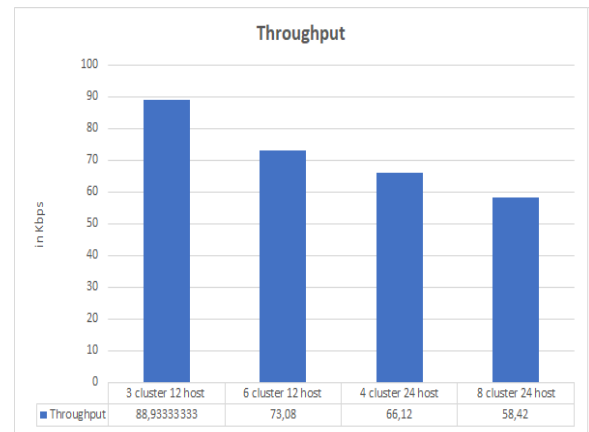


Fig. 8 Throughput's result

e) CPU Usage: The next measurement in this final project is CPU usage on the host. Figure 9 shows that if we applied topology with 3 clusters of 12 sensors, greater than 6 clusters of 12 sensors, the difference is not very big. Add clusters to each cluster head, and the addition of sensors puts a load on the host CPU running the emulator. Figure 9 shows that the CPU host is still in good condition because it is below 80% value.

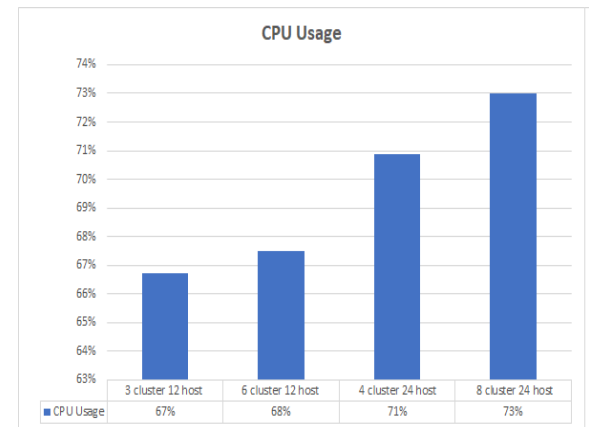


Fig. 9 CPU Usage result

Analysis of the results to achieve this study's objectives, network application SDN on 6LoWPAN device based on Open Network Operating System (ONOS) controller done with good results. The topology made able to meet several IoT network requirements with measurement specifications carried out as the research progresses. The difficulty is to understand the use of interfaces on the 6LoWPAN sensor to minimize the excessive use of interfaces. Furthermore, the network quality analysis results with the TIPHON and ITU-T reference standards are in a good category. Except that the resulting throughput value cannot be categorized as good because the resulting maximum 88.93 Kbps, the IEEE 802.15.4 protocol has limitations linked or bandwidth up to 150 Kbps. Lack in terms of this measurement is in terms of RAM usage.

V. CONCLUSION

Based on the results of research on this paper reported the results of best throughput, delay, jitter, and packet loss values

are the 3 AP 12 host topology, based on the table of hardware requirements used and the results QoS test chart of all topologies in good category by standard TIPHON and ITU-T. The following conclusions are obtained, adding an ethernet interface on each host or sensor to connect to the switch to connect each cluster. In testing, the jitter that has the smallest value in each scenario is 3 AP 12 hosts. Despite the comparison between additional cluster scenarios and the addition of a host switch is not much different. In testing packet loss, which has the lowest percentage in each scenario, 3 APs were 12 hosts, although each scenario's average is close to the same. The throughput test with the greatest value is 3 AP 12 host, the difference in throughput using 2 clusters per switch and 1 cluster per switch only a little. In the delay test, which has the smallest value in each scenario is 3 AP 12 hosts. In the CPU usage test, the smallest value in each scenario is three switches to 12 hosts.

REFERENCES

- [1] M. Ojo, D. Adami, and S. Giordano, "A SDN-IoT architecture with NFV implementation," 2016 IEEE Globecom Work. GC Wkshps 2016 - Proc., 2016.
- [2] N. Bizanis and F. A. Kuipers, "SDN and Virtualization Solutions for the Internet of Things: A Survey," *IEEE Access*, vol. 4, no. c, pp. 5591–5606, 2016.
- [3] A. C. G. Anadiotis, L. Galluccio, S. Milardo, G. Morabito, and S. Palazzo, "Towards a software-defined Network Operating System for the IoT," *IEEE World Forum Internet Things, WF-IoT 2015 - Proc.*, pp. 579–584, 2015.
- [4] O. Marzuqi, A. Virgono, and R. M. Negara, "Implementation model architecture software defined network using raspberry Pi: A review paper," *Telkomnika (Telecommunication Comput. Electron. Control.*, vol. 17, no. 3, 2019, doi: 10.12928/TELKOMNIKA.V17I3.8859
- [5] Q. H. Nguyen, N. Ha Do, and H. C. Le, "Development of a QoS Provisioning Capable Cost-Effective SDN-based Switch for IoT Communication," *Int. Conf. Adv. Technol. Commun.*, vol. 2018-October, pp. 220–225, 2018.
- [6] I. ALAOUI ISMAILI, A. Azyat, N. Raissouni, N. Ben Achhab, A. Chahboun, and M. Lahraoua, "Comparative Study of ZigBee and 6LoWPAN Protocols: Review," no. January, 2019.
- [7] A. C. G. Anadiotis, L. Galluccio, S. Milardo, G. Morabito, and S. Palazzo, "Towards a software-defined Network Operating System for the IoT," *IEEE World Forum Internet Things, WF-IoT 2015 - Proc.*, pp. 579–584, 2015.
- [8] The Open Networking Lab (ON.Lab). Introducing ONOS - a SDN network operating system for Service Providers. White Paper, 1:14, 2014.
- [9] Internet of Things state of art and challenges. RFC 8576, April 2019.
- [10] R. Wulandari, ANALISIS QoS (QUALITY OF SERVICE) PADA JARINGAN INTERNET (STUDI KASUS: UPT LOKA UJI TEKNIK PENAMBANGAN JAMPANG KULON LIPI), J. Tek. Inform. dan Sist. Inf., vol. 2, pp. 162172, 2016.
- [11] Hui, J., Thubert, P. (2011). "RFC 6282 : Compression Format for IPv6 Datagrams over IEEE 802.15.4-Based Networks". ISSN: 2070-1721, September 2011.
- [12] H. Fotouhi, D. Moreira, M. Alves, "mRPL: Boosing Mobility in the Internet of Things", *Science Direct Journal of Ad Hoc Networks* 26, 2015.
- [13] R. M. Huq, K. P. Moreno, H. Zhu, J. Zhang, O. Ohlsson, and M. I. Hossain, "On the benefits of clustered capillary networks for congestion control in machine type communications over LTE," *Proc. - Int. Conf. Comput. Commun. Networks, ICCCN*, vol. 2015-October, no. November 2016, 2015.