International Journal of Synergy in Engineering and Technology
IJSET

Development of SDN Controller Testbed Using Rasberry Pi 4

Nur Fatin Amirah binti Mohd Rodzi, Lukmanulhakim bin Ngah*

Faculty of Computer, Media and Technology Management, University College TATI, MALAYSIA

*Corresponding author: hakim@uctati.edu.my

KEYWORDS

Software Defined Network
Controller
Testbed
Raspberry Pi 4

ABSTRACT

Nowadays, people are living in a 'internet of things' world. Everything can be programmed over network and no longer depended on hardware and human interaction to transfer data. Software Defined Network (SDN) is one of the creations of this new kind of world which programmed efficient network configuration that improve network performance and monitoring, which will lessen the hassle of configuring devices. By focusing to SDN, this project will study the performance of SDN by Controller Testbed using Raspberry Pi done to measure the network performance efficiency by the parameter of TCP and UDP throughput. The main purpose of this research project is to Develop SDN Controller testbed using Raspberry Pi 4 by using TCP and UDP throughput as the main metric measurement. The hardware used is Raspberry Pi 4 and Raspberry Pi 3, and the tools used is called JPERF will be used to measure the throughput of TCP and UDP. The result from this test conclude that Raspberry Pi 4 are suitable and compatible hardware to conduct SDN network. From the result, we may know what the difference of network performances of SDN in Raspberry Pi 4, which resulting that Raspberry Pi 4 is more compatible than Raspberry Pi 3. This project also prove that Raspberry Pi 4 can support SDN network.

1.0 Introduction

Ensuring uninterrupted Internet services and providing high Quality-of-Services (QoS) is becoming burgeoning demand in fulfilling industry-specific regulations. This alternative is towards improving future Internet and service mobility to enhance overall user experience. As such, networking protocols have evolved significantly over the last few decades. However, traditional network configuration is time-consuming and error-prone due to manual configuration by network administrator. The traditional network architecture that integrates the forward plane and control plane into the same device cannot support these requirements [1].

The problem statement of this research is there is no project of SDN POX Controller by using Raspberry Pi 4 and Raspberry Pi B+ have an issue with not fit for using Open vSwitch and POX

Received 30 November 2021; received in revised form 17 December 2021; accepted 24 December 2021.

Controller [2]. The objectives of this research are to develop SDN controller testbed using Raspberry Pi 4. Next is to test network throughput TCP & UDP protocol. The third objective is to Compare CPU Load between Raspberry Pi 4 and Raspberry Pi 3 of their utilization on SDN

To solve this problem, the new network architecture method called Software Defined Network proposed. SDN is a new networking paradigm that separates the control and forwarding planes in a network. The network can be dynamically managed depending on networking policies [3]. SDN helps to detach control plane from data plane, which helps the administrator to manage the network centrally and can utilize bandwidth at its maximum, resulting in great flexibility.

2.0 Literature Review

Software-Defined Networking is a form of OpenFlow-enabled networks that utilize the OpenFlow protocols and an external Application Program Interface (API) to control the decision functionality of the network [4]. SDN is one of the major programmable networks gaining fast popularity among network administrators due to its simplicity and efficiency. Although traditional networks have been efficient over the years, the major difference between the traditional network architecture and SDN architecture is the SDN controller component. The SDN controller is a controlling element that controls underlying network devices via a well-developed programmable coding scheme [5].

OpenFlow is just an option for a control protocol in SDN, but it is the predominant one. As OpenFlow currently evolves, several versions of its specification exist; newer releases are more powerful, as their feature sets support IPv6, Multiprotocol Label Switching (MPLS), rerouting, metering, policing and more scalable control. However, most existing applications are based on version 1.0, whose feature set is rather limited [6].

The controller connected to every networking device (switches and routers) via a separate control network thus allowing it to control and monitor each device [7]. This means that once a packet sent from a node towards a destination interface, the switches and routers access the centralized control logic (SDN controller) for a decision on how packets forwarded and what route to take. Since the control function of the network resides within the SDN controller, it is necessary to make the controller resilient and full-tolerance. To achieve this, a tightly coupled cluster of SDN controllers used to control and manage the network devices [8]. This is especially needed in large environments such as enterprise networks and cloud computing. POX is an open-source controller for developing SDN applications.

POX controller provides an efficient way to implement the OpenFlow protocol which is the de facto communication protocol between the controllers and the switches. Using POX controller, you can run different applications like hub, switch, load balancer, and firewall. Tcpdump packet capture tool can be used to capture and see the packets flowing between POX controller and OpenFlow devices [9].

This paper puts together all elements to build a low-cost micro software defined data centre by leveraging off-the-shelf hardware and open-source software. We propose a system architecture for constructing a testbed/micro data centre for researching SDN enabled cloud computing. We focus on the cost-effectiveness of our setup by reusing existing equipment and coping with the budget and space limitations of an academic research laboratory. One of the most important benefits of SDN is that commodity hardware can be used to build networking devices. Therefore, we use Raspberry Pi, low-cost and small single-board computers, to create a small-scale data centre network [10]. To make a switch out of Raspberry Pi, we integrate each Pi with an Open vSwitch (OVS), which is one of the most widely used virtual switches in SDN and use POX controller as controller in this system.

Raspberry Pi is the name of a series of single-board computers made by the Raspberry Pi Foundation, a UK charity that aims to educate people in computing and create easier access to computing education. All over the world, people use Raspberry Pi to learn programming skills, build hardware projects, do home automation, and even use them in industrial applications. The

Raspberry Pi is a very cheap computer that runs Linux, but it also provides a set of GPIO (general purpose input/output) pins that allow you to control electronic components for physical computing and explore the Internet of Things (IoT) [11].

Raspberry Pi 4 Model B is the latest product in the popular Raspberry Pi range of computers. It offers ground-breaking increases in processor speed, multimedia performance, memory, and connectivity compared to the prior generation Raspberry Pi 3 Model B+, while retaining backwards compatibility and similar power consumption. For the end user, Raspberry Pi 4 Model B provides desktop performance comparable to entry-level x86 PC systems [12].

This product's key features include a high-performance 64-bit quad-core processor, dual-display support at resolutions up to 4K via a pair of micro-HDMI ports, hardware video decodes at up to 4Kp60, up to 8GB of RAM, dual-band 2.4/5.0 GHz wireless LAN, Bluetooth 5.0, Gigabit Ethernet, USB 3.0, and PoE capability (via a separate PoE HAT add-on). The dual-band wireless LAN and Bluetooth have modular compliance certification, allowing the board to be designed into end products with significantly reduced compliance testing, improving both cost and time to market [12].

3.0 Methodology

This project only uses Raspberry Pi 4, which is the latest technology of Raspberry Pi. This project concentrates on the network flow where it tests the network throughput TCP & UDP protocol and compare CPU Load of the network in SDN when using Raspberry Pi 4. This project established to develop SDN successfully by testing and proving that Raspberry Pi 4 could support SDN than other older Raspberry Pi technology. Therefore, the decision to use Raspberry Pi 4 is actually to enhance SDN network and proving that Raspberry Pi 4 is more enhance than Raspberry Pi B+ in supporting SDN.

SDN controllers' direct traffic according to forwarding policies that a network operator puts in place, thereby minimizing manual configurations for individual network devices. By taking the control plane off the network hardware and running it instead as software, the centralized controller facilitates automated network management and makes it easier to integrate and administer business applications. In effect, the SDN controller serves as a sort of operating system (OS) for the network. The hardware and software used were Raspberry Pi 4, Raspberry Pi 3, Raspberry Pi OS, Open vSwitch, POX controller and Iperf.

The test setup for the OpenFlow enabled Raspberry Pi that act as controller of POX controller which consists of a PC and Raspberry Pi 4. The goal of this test is to determine if the Raspberry Pi offers enough performance to run the installed Open vSwitch. To test the throughput, the JPERF command is run between the two hosts for 30 seconds to allow for flow rules to be placed by the controller without influencing the average values too much. The results will be taken by observing the transmission speeds of TCP and UDP. At the same time the CPU load will be compared. In the fourth stage, the devices will be setup for the testbed testing to make sure the capability and the functional of the device are working properly. In this stage, the proposed design will be implemented to test the device with the public environment in order to see how the device can work properly as requested. The setup of the device will be tested on SDN network using Raspberry Pi 4 as controller. From this setup, the network performance will be measured on parameters of TCP & UDP throughput. The outcome of this process is the result obtained.

4.0 Result

The expected result for this research project is to show the comparison of the result, in terms of the networking performances of Raspberry Pi 4 and Raspberry Pi 3. Testing procedure will be divided into two part which is network throughput TCP & UDP protocol, and compare CPU Load between Raspberry Pi 4 and Raspberry Pi 3. The metrics for TCP is bandwidth and for UDP are bandwidth and jitter. The test will be conducted first using Windows 10 and then Ubuntu 14.04.4. For the testing, the method that have been used is transferring 100MB, 200MB and 300MB of TCP

testing for bandwidth utilization, and transferring 25MB, 50MB and 100MB of UDP testing for bandwidth utilization.

4.1 Transmission Control Protocol (TCP) on Raspberry Pi 4

a) Server

The metrics used for TCP is bandwidth utilization. For the figure below, the first experiment of server side shows that the average bandwidth for 100MB is 621.00 Mbytes/s.

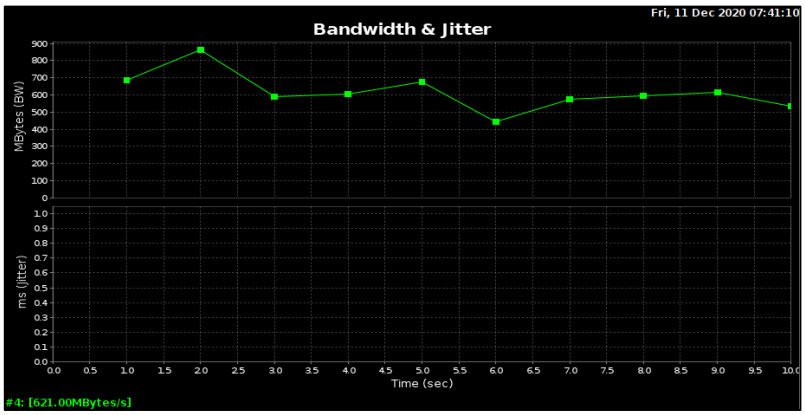

Figure 1: Bandwidth reading of 100mb bandwidth utilization

The metrics used for TCP is bandwidth utilization. For the figure below, the first experiment of server side shows that the average bandwidth for 200MB is 731.00 Mbytes/s.

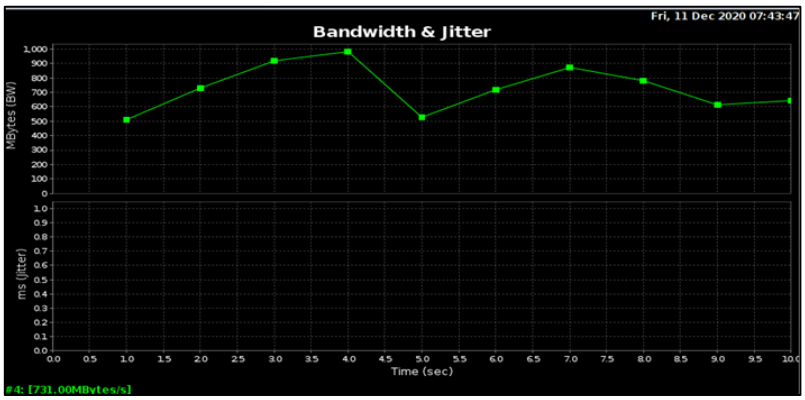

Figure 2: Bandwidth reading of 200mb bandwidth utilization

The metrics used for TCP is bandwidth utilization. For the figure below, the first experiment of server side shows that the average bandwidth for 300MB is 670.00 Mbytes/s.

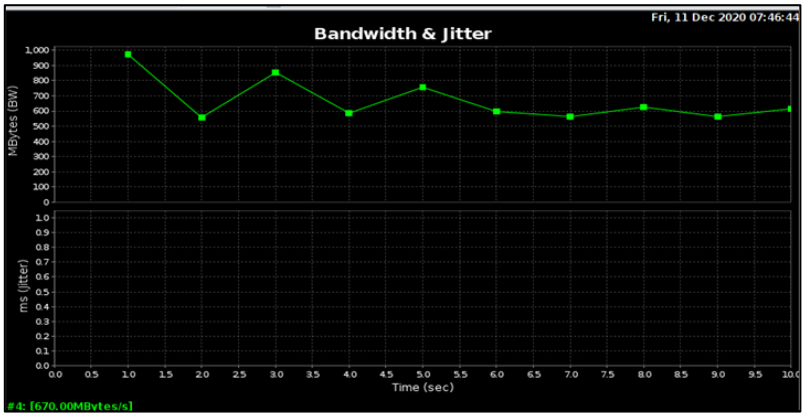

Figure 3: Bandwidth reading of 300mb bandwidth utilization

b) Client

The metrics used for TCP is bandwidth utilization. For the figure below, the first experiment of server side shows that the average bandwidth for 100MB is 623.00 Mbytes/s.

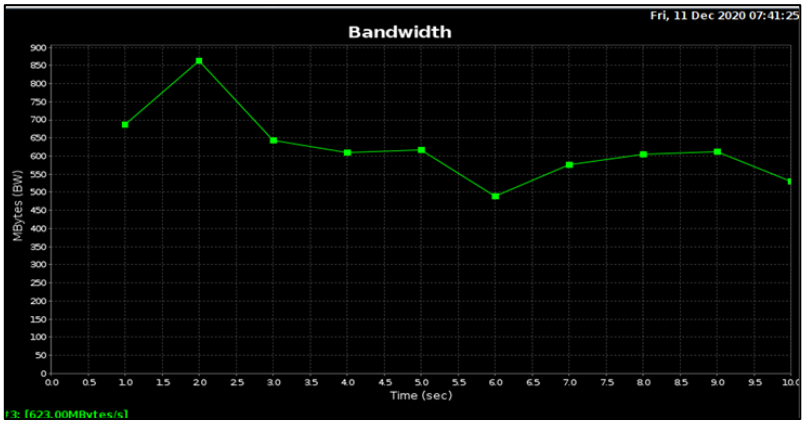

Figure 4: Bandwidth reading of 100mb bandwidth utilization

The metrics used for TCP is bandwidth utilization. For the figure below, the first experiment of server side shows that the average bandwidth for 200MB is 735.00 Mbytes/s.

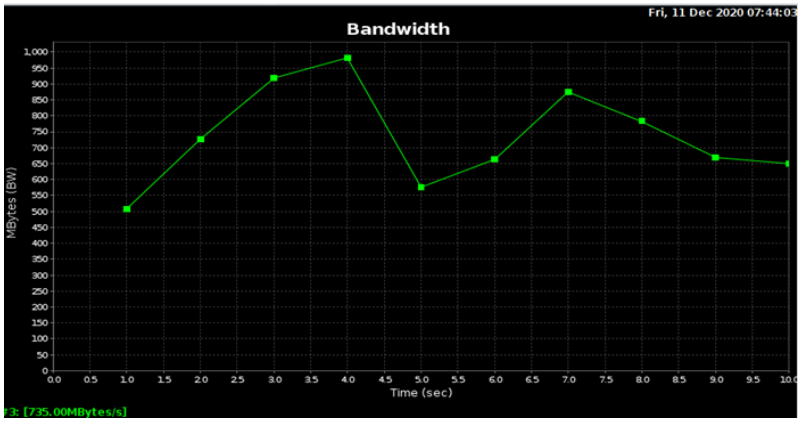

Figure 5: Bandwidth reading of 200mb bandwidth utilization

The metrics used for TCP is bandwidth utilization. For the figure below, the first experiment of server side shows that the average bandwidth for 300MB is 675.00 Mbytes/s.

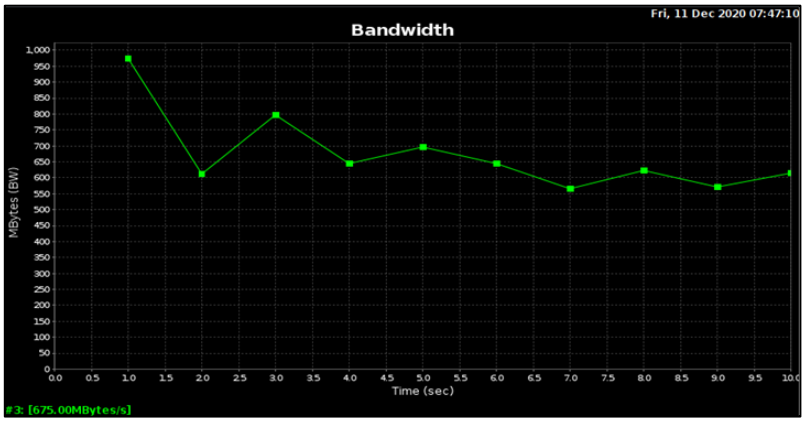

Figure 6: Bandwidth reading of 300mb bandwidth utilization

4.2 User Datagram Protocol (UDP) on Raspberry Pi 4

a) Server

The metrics used for UDP is bandwidth utilization and jitter for server side. For the figure below, the first experiment of server side shows that the average bandwidth utilization for 25 MBytes is 3.12 MBytes/s and the average jitter is 0.01ms.

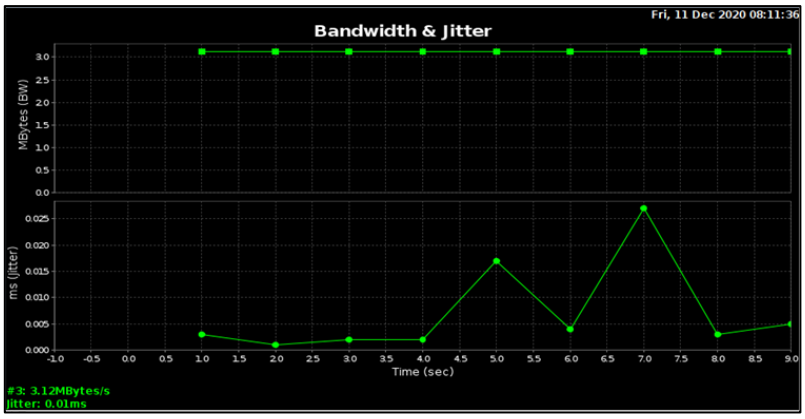

Figure 7: Bandwidth reading of 25mb bandwidth utilization

The metrics used for UDP is bandwidth utilization and jitter for server side. For the figure below, the first experiment of server side shows that the average bandwidth utilization for 50 MBytes is 6.25 MBytes/s.

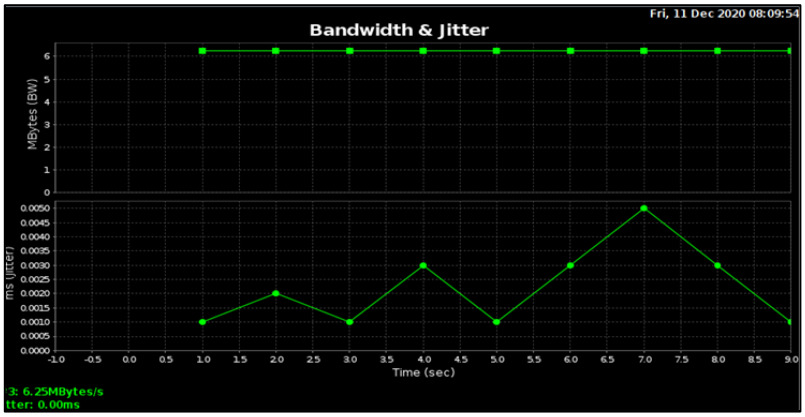

Figure 8: Bandwidth reading of 50mb bandwidth utilization

The metrics used for UDP is bandwidth utilization and jitter for server side. For the figure below, the first experiment of server side shows that the average bandwidth utilization for 100 MBytes is 12.50 MBytes/s.

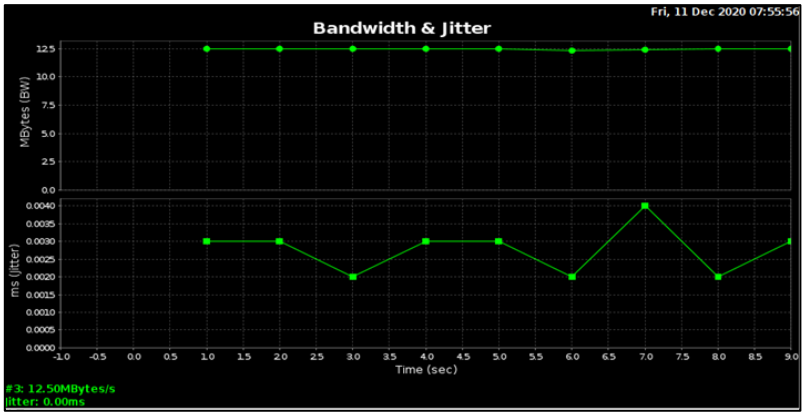

Figure 9: Bandwidth reading of 100mb bandwidth utilization

b) Client

The metrics used for UDP is bandwidth utilization for client. For the figure below, the first experiment of client side shows that the average bandwidth utilization for 25 MBytes is 3.12 MBytes.

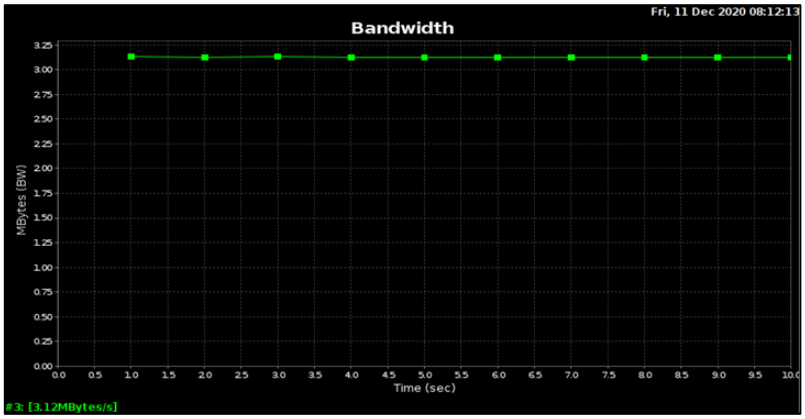

Figure 10: Bandwidth reading of 25mb bandwidth utilization

The metrics used for UDP is bandwidth utilization for client. For the figure below, the first experiment of client side shows that the average bandwidth utilization for 50 MBytes is 6.25 MBytes.

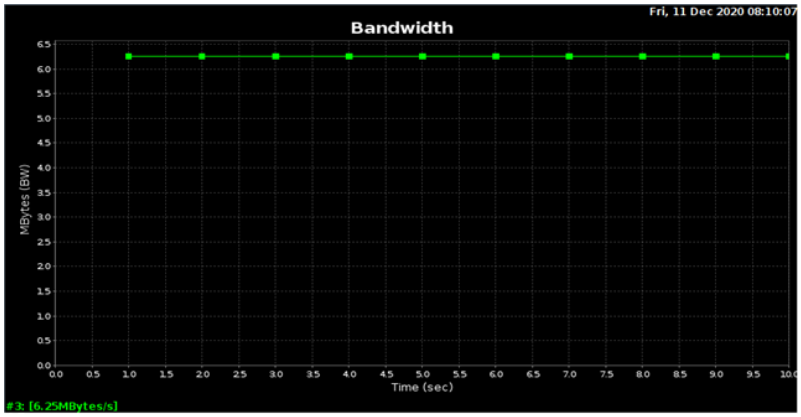

Figure 11: Bandwidth reading of 50mb bandwidth utilization

The metrics used for UDP is bandwidth utilization for client. For the figure below, the first experiment of client side shows that the average bandwidth utilization for 100 MBytes is 12.50 MBytes.

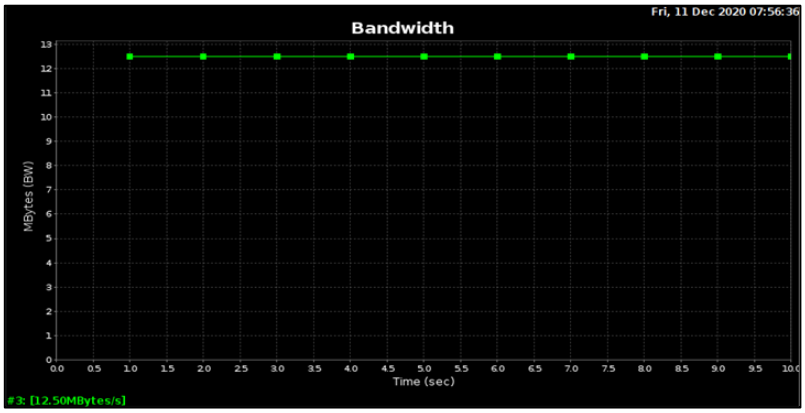

Figure 12: Bandwidth reading of 100mb bandwidth utilization

4.3 Comparison of CPU load between Raspberry Pi 4 and Raspberry Pi 3

CPU load was measure by CPU load average which is the number of processes which are being executed by CPU or waiting to be executed by CPU. For Raspberry Pi 4, the load average over the last 1 minute is 0.18, load average over the last 5 minutes is 0.59 and the load average over the last 15 minutes is 0.74.

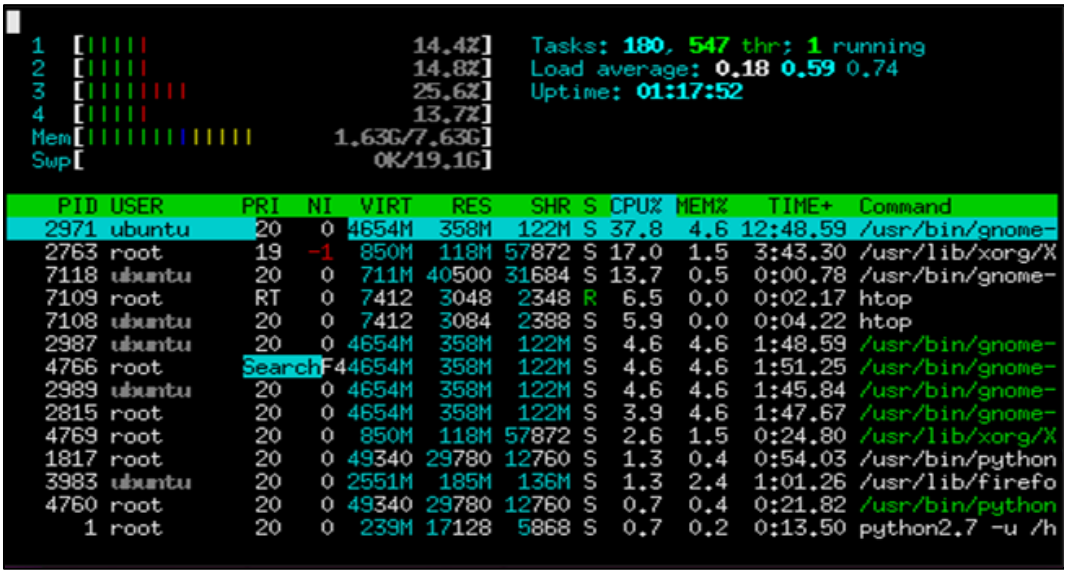

Figure 13: CPU load of Raspberry Pi 4

CPU load was measure by CPU load average which is the number of processes which are being executed by CPU or waiting to be executed by CPU. For Raspberry Pi 3, the load average over the last 1 minute is 2.45, load average over the last 5 minutes is 3.02 and the load average over the last 15 minutes is 2.79.

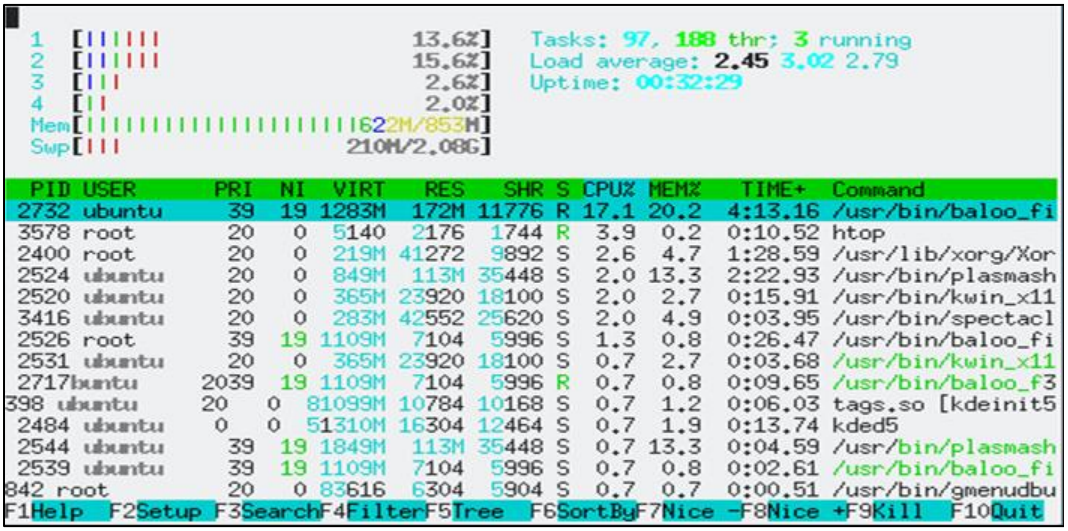

Figure 14: CPU load of Raspberry Pi 3

Table 1 shows CPU load average of Raspberry Pi 4 and Raspberry Pi 3. The CPU load average over the last 1 minute for Raspberry Pi 4 is 0.18 while Raspberry Pi 3 is 2.45. This means that, CPU of Raspberry Pi 4 was idle by 382% on average; no processes were waiting for CPU time

(0.18) while CPU of Raspberry Pi 3 was idle by 155% on average; no processes were waiting for CPU time (2.45). The CPU load average over the last 5 minutes for Raspberry Pi 4 is 0.59 while Raspberry Pi 3 is 3.02. This means that, CPU of Raspberry Pi 4 was idle by 341% on average; no processes were waiting for CPU time (0.59) while CPU of Raspberry Pi 3 was idle by 98% on average; no processes were waiting for CPU time (3.02). The CPU load average over the last 15 minutes for Raspberry Pi 4 is 0.74 while Raspberry Pi 3 is 2.79. This means that, CPU of Raspberry Pi 4 was idle by 341% on average; no processes were waiting for CPU time (0.74) while CPU of Raspberry Pi 3 was idle by 121% on average; no processes were waiting for CPU time (2.79).

Table 1: CPU load usage of Raspberry Pi 4 and Raspberry Pi 3

CPU load average	Time before execution (minute)		
	1	5	15
Raspberry Pi 4	0.18	0.59	0.74
Raspberry Pi 3	2.45	3.02	2.79

5.0 Conclusion

From the result, we can conclude that SDN are more compatible with Raspberry Pi 4 where it doesn't burden CPU where it has less load average when SDN was active while on Raspberry Pi 3, has higher value of load average which can lead to overload if it run much longer where this can cause users to be waiting for their programs to run on the CPU, and experiencing degraded performance which means that it's still compatible but less compatible than Raspberry Pi 4.

References

[1] M. A. I. M. Sakari, N. Yaakob, A. Amir, R. B. Ahmad, M. N. M. Warip, and Z. Ibrahim, "Performance analysis of Software Defined Network (SDN) in link failure scenario," IOP Conf. Ser. Mater. Sci. Eng., vol. 557, no. 1, pp. 6–11, 2019, doi: 10.1088/1757-899X/557/1/012028.
[2] F. Siebertz and P. K. Jonas, "Masterprojekt Software Defined Networking," 2014.
[3] R. Intan, C. H. Chi, H. N. Palit, and L. W. Santoso, "Preface," Commun. Comput. Inf. Sci., vol. 516, pp. 395–403, 2015, doi: 10.1007/978-3-662-46742-8.
[4] F. Hu, Q. Hao, and K. Bao, "A survey on software-defined network and OpenFlow: From concept to implementation," IEEE Commun. Surv. Tutorials, vol. 16, no. 4, pp. 2181–2206, 2014, doi: 10.1109/COMST.2014.2326417
[5] W. Unger, "Evaluating security of SDN controllers," no. April, 2016.
[6] W. Braun and M. Menth, "Software-Defined Networking Using OpenFlow: Protocols, Applications and Architectural Design Choices," Futur. Internet, vol. 6, no. 2, pp. 302–336, 2014, doi: 10.3390/fi6020302.
[7] A. S. Dawood and M. N. Abdullah, "A Survey and Comparative Study on Software-Defined Networking," Int. Res. J. Comput. Sci., vol. 3, no. no.8, pp. 1–10, 2016.
[8] B. A. A. Nunes, M. Mendonca, X. N. Nguyen, K. Obraczka, and T. Turletti, "A survey of software-defined networking: Past, present, and future of programmable networks," IEEE Commun. Surv. Tutorials, vol. 16, no. 3, pp. 1617–1634, 2014, doi: 10.1109/SURV.2014.012214.00180
[9] M. Fernandez, "Evaluating OpenFlow Controller Paradigms," ICN 2013, Twelfth Int. Conf. pp. 151–157
[10] A. Nadjaran Toosi, J. Son, and R. Buyya, "CLOUDS-Pi: A Low-Cost Raspberry-Pi based Micro Data Center for Software-Defined Cloud Computing," IEEE Cloud Comput., vol. 5, no. 5, pp. 81–91, 2018, doi: 10.1109/MCC.2018.053711669.
[11] "What is a Raspberry Pi?" https://opensource.com/resources/raspberry-pi (accessed Jul. 30, 2020).
[12] "RPI4-MODBP-8GB." https://my.element14.com/raspberry-pi/rpi4-modbp-8gb/raspberry-pi-4-model-b-cortex/dp/3369503 (accessed Jul. 30, 2020).