

**SKR 4308**

**Software Defined Network**

**Title:**

SDN network with Raspberry Pi

**Lecturer name:**

Prof. Dr. Mohamed bin Othman

**Group members:**

Mohamad Khairul Zamidi bin Zakaria (202180)

Muhammad Muzammel bin Imamuzel (202179)

Table of contents

[**1. Introduction**](#_b0542k1zjyd6) **3**

[**2. Problem statement**](#_ec09qcvhdxw2) **3**

[**3. Objectives**](#_sjflix2vob8x) **3**

[**4. Scope**](#_dbfddjb6xvi) **3**

[**5. Literature Review**](#_tmtugpxh9m7n) **4**

[**6. Methodology**](#_26xtw9j236kf) **4**

[6.1 Hardware and software requirements](#_2k185hwkw5oi) 4

[6.2 Topology](#_adfgarkkcy9c) 5

[6.3 Performance metric](#_76u2b9h83kij) 6

[6.4 Installation and configurations](#_6ogefco5flml) 6

[6.4.1 Installation of VirtualBox](#_23a0ehux331v) 6

[6.4.2 Installation of Mininet](#_wny1y5my4ic9) 8

[6.4.3 Installation of Raspberry Pi VM](#_urzq279zb0d6) 11

[6.4.4 Installation of ONOS](#_fzmupx27vpti) 13

[6.4.5 Configuration of topology in mininet](#_m82t9s1rn9x3) 17

[6.5 Testing connectivities](#_c2tvc7r26lzj) 19

[**7. Performance evaluation**](#_rq42ib14dv9y) **20**

[7.1 Scenario 1: UDP protocol with bandwidth 200 Mbits/s](#_vqsouyu18zfk) 20

[7.2 Scenario 2: UDP protocol with bandwidth 400 Mbits/s](#_1ou9gkpi0t4) 21

[7.3 Scenario 3: UDP protocol with bandwidth 600 Mbits/s](#_5ni7m7hsybgm) 21

[7.4 Scenario 4: UDP protocol with bandwidth 800 Mbits/s](#_811uryw49vxu) 22

[7.5 Results](#_3xs2y564zd79) 23

[**8. Conclusion**](#_73dnuwi0znr6) **24**

[**9. References**](#_9ia68ore5rvw) **25**

# 1. Introduction

Traditional computer networks still have severe flaws today. Because it employs a low-level setup and a configuration that takes a long time, especially for large-scale networks, such a device is still difficult to configure. All network devices will be focused on a single controller that can operate, administer, and monitor the network more efficiently due to Software Defined Network (SDN) technology.

In this project,we will be using Open Network Operating System (ONOS) based on SDN technology which is able to separate the control plane and the data plane on the network device.The ONOS will be installed on Raspberry-Pi 3 device to minimise the usage of PC on the network

# 2. Problem statement

1. How to use an ONOS controller as a control plane?
2. How to use Raspberry Pi as a controller with the Raspbian Operating System?
3. How to measure the Quality of Service (QoS)?

# 3. Objectives

1. To implement SDN network by using ONOS Controller on Raspberry PI 3
2. To measure the Quality of Service (QoS) TIPHON in the SDN network

# 4. Scope

We will start this project on VirtualBox in order to create virtual machines for Raspberry PI os and Mininet.

Then, we will try to configure the network based on the topology created which consists of 4 hosts, 3 switches and 1 controller which we will be using ONOS as the controller.The topology we created are based on the SDN network which will separate the control plane and the forwarding plane.

The result we look for after this configuration completes will be transfer(MBytes) , Bitrate, Jitter and packet loss.

The results we are looking for are not quite the same as the research paper as we want to see and experiment other things that differ from the article.

# 5. Literature Review

1. **Software Defined Network**

All network devices will be centred on one controller that can control, manage and monitor the network more effectively.

1. **OpenFlow**

Communication protocol that connects the controller with the data plane. OpenFlow give access to manipulate forwarding plane directly from network device.

1. **Open Network Operating System (ONOS)**

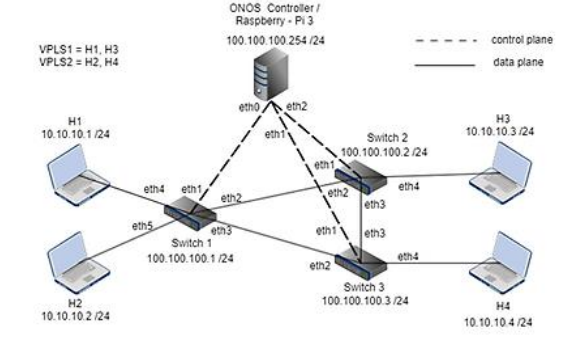
Open source SDN controller for building next-generation SDN/NFV solutions. Built to provide high availability, scale-out and network performance.

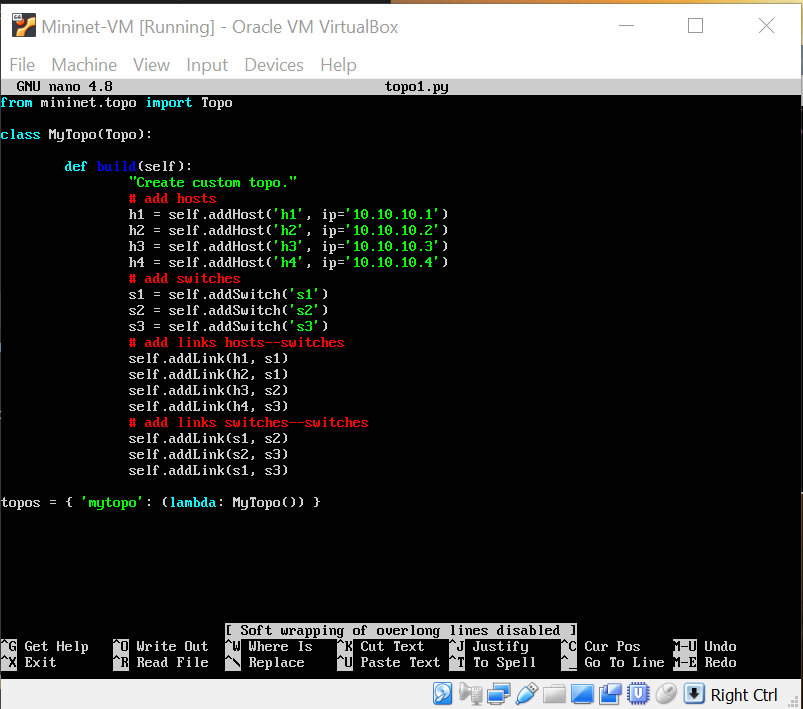
# 6. Methodology

## 6.1 Hardware and software requirements

|  |  |
| --- | --- |
| Hardware | Software |
| 1 x PC | VirtualBox  iPerf  Mininet  ONOS |

## 6.2 Topology





## 6.3 Performance metric

In our project, there are differences between our reimplementation of the SDN with Raspberry Pi with our referred research article. Based on the research paper, they use the UDP protocol and use background traffic to measure four things; throughput, delay, jitter, and packet loss. However, in our reimplementation, we did not use background traffic, but we set the bandwidth at the client side to 200Mbps, 400Mbps, 600Mbps, and 800Mbps and send the data using UDP protocol and measured four things, that are:

1. Data transfer
2. Bitrate
3. Jitter
4. Packet loss

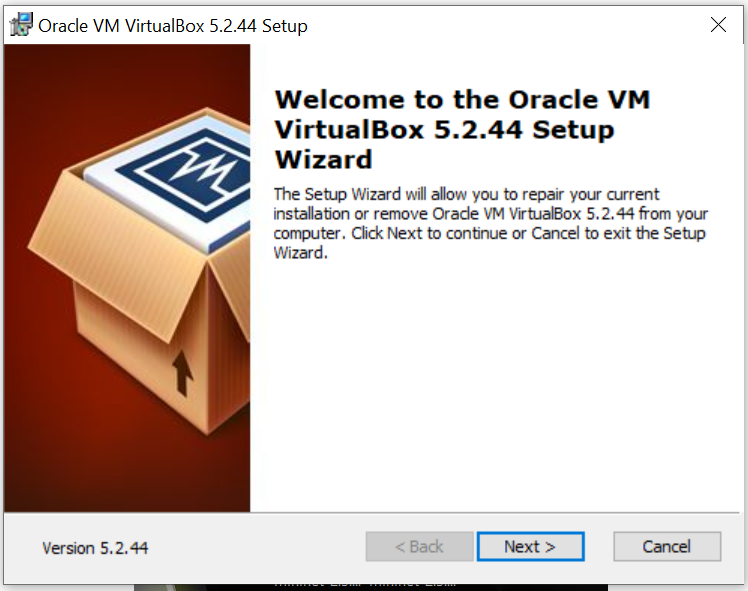
In our future enhancement, we can set the background traffic and measure the performance to compare the result with the research paper.

## 

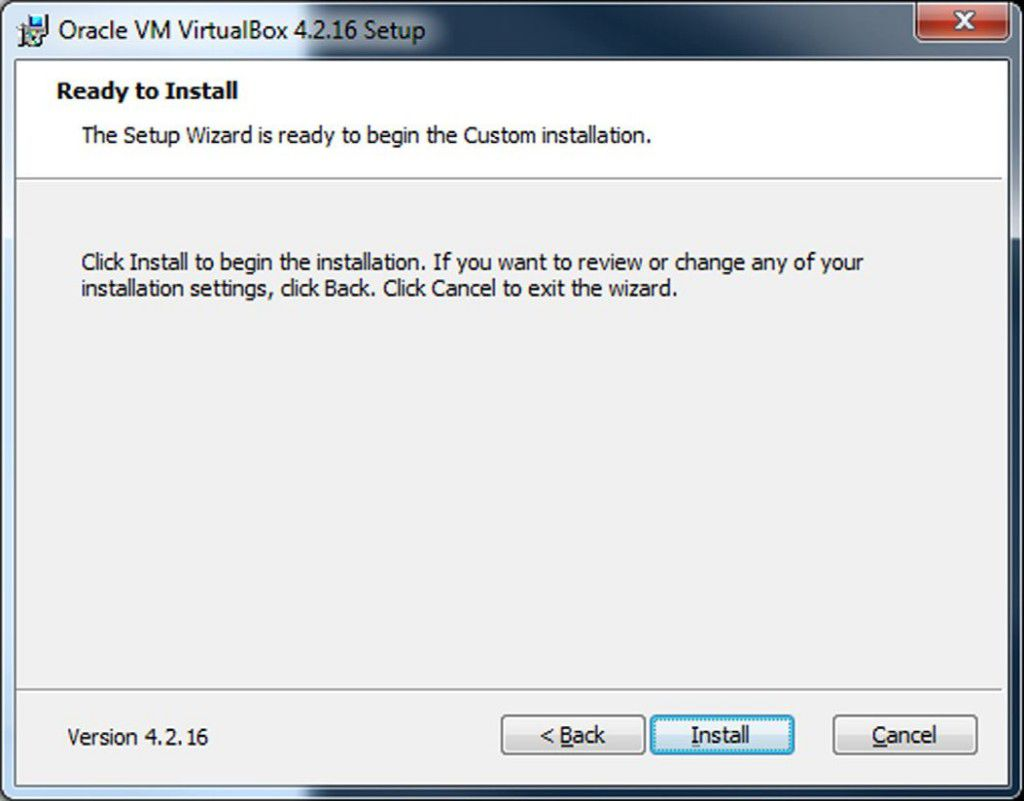
## 6.4 Installation and configurations

### 6.4.1 Installation of VirtualBox

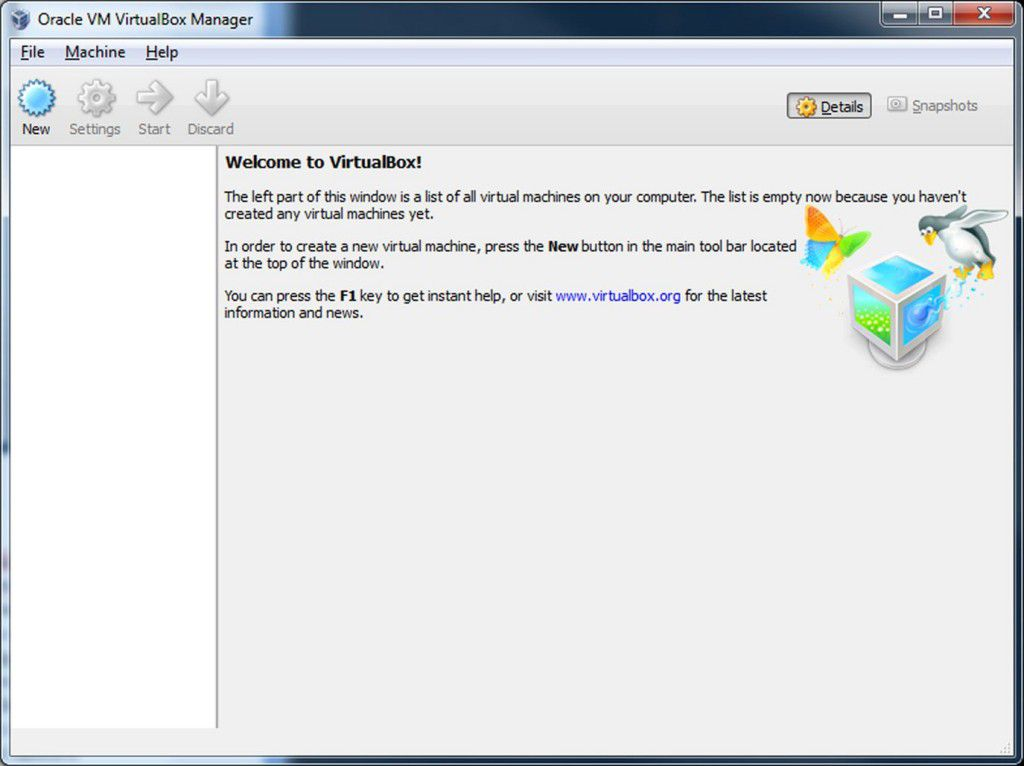
1. Visit the Virtual Box website: <https://www.virtualbox.org/wiki/Downloads> and download the latest version according to the version of your operating system Windows, Mac or Linux.
2. Run the VirtualBox-5.2.44-137108-Win.exe and the Oracle VM Virtual Box Installation window will appear. Press Next.



1. Follow step by step installation instructions. Click “Install” to start the installation and wait for the next step required until the installation is complete.

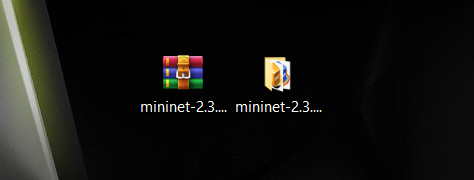


1. After finishing the installation process, the virtualBox Window can be opened and can be used.

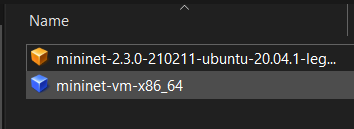


### 6.4.2 Installation of Mininet

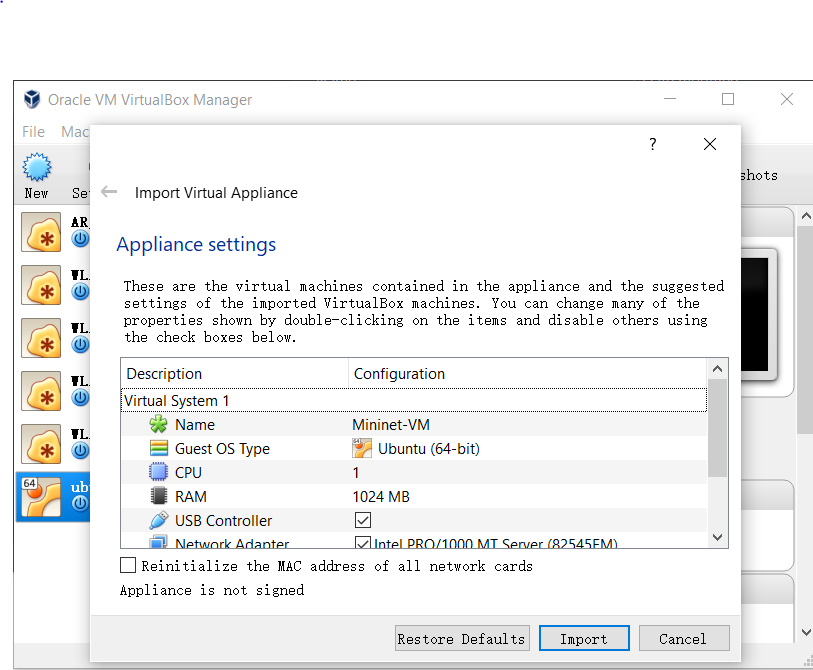
1. Visit the Mininet website : <https://github.com/mininet/mininet/releases/> and download the recommended installation.
2. After downloading the ovf zip file ,extract it and open the folder after the extraction.



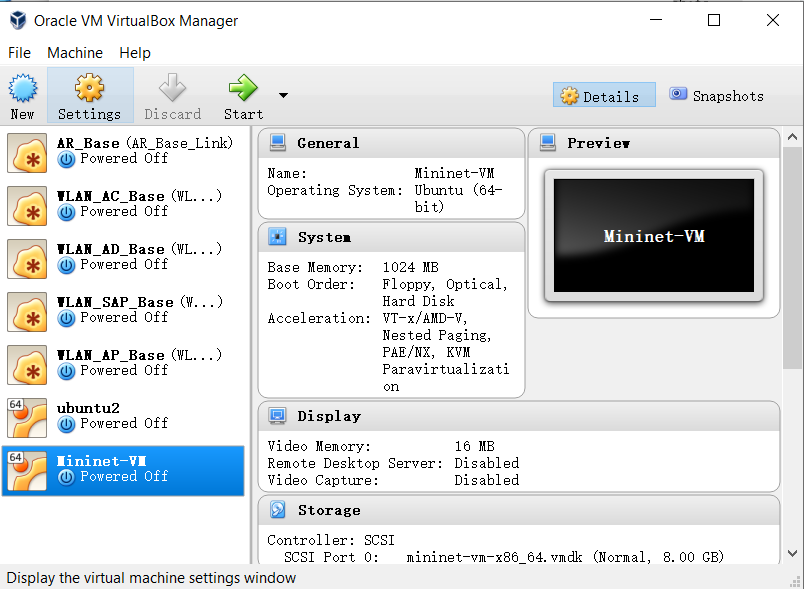
1. Double click on the mininet-2.3.0-210211-ubuntu-20.04.1-legacy-server-amd64



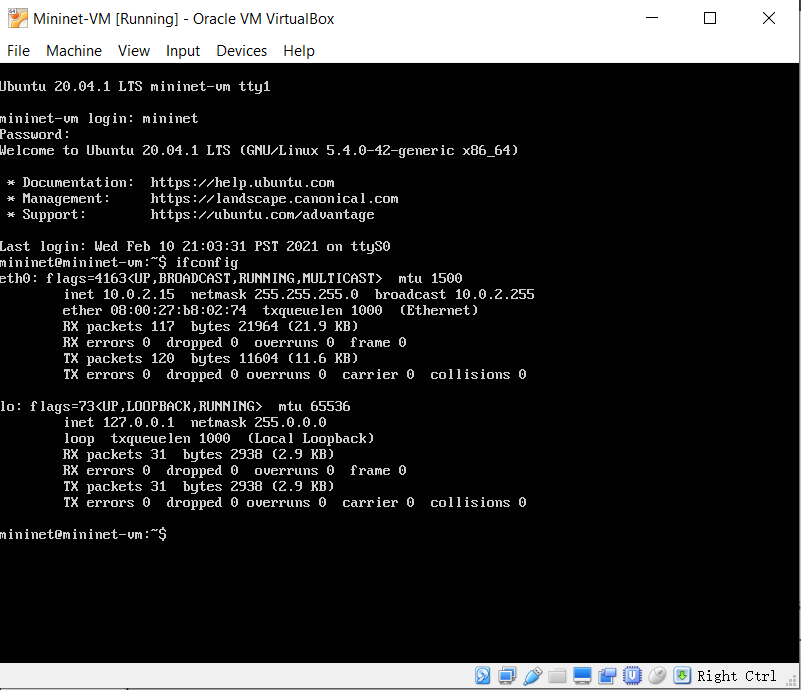
1. Import the virtual appliances into the virtual box



1. Finally, choose the Mininet-VM in the VirtualBox and run it.

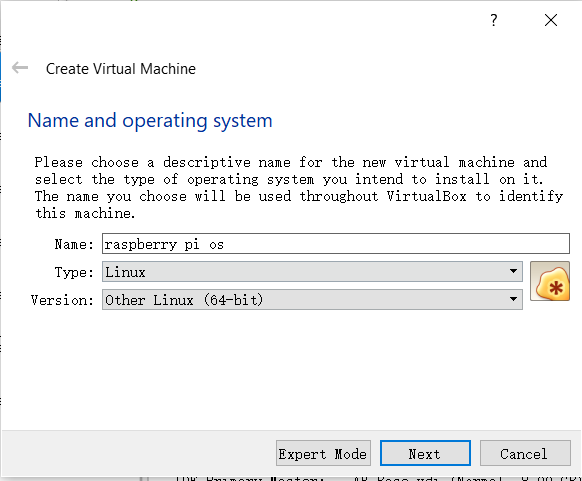


1. It should ask the user to put the username and password to login

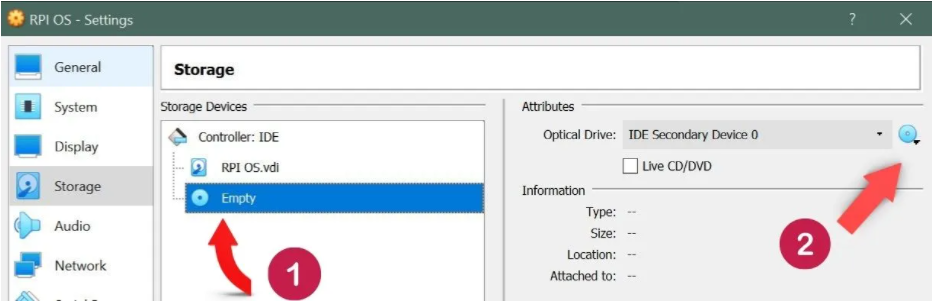


### 6.4.3 Installation of Raspberry Pi VM

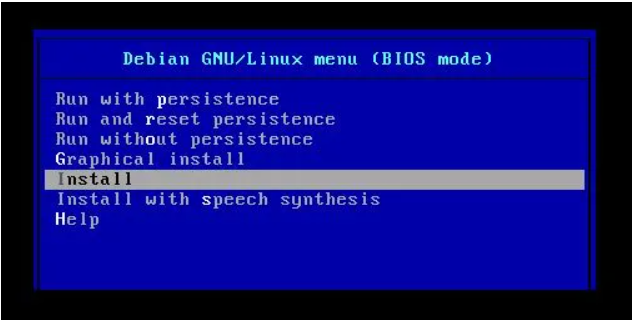
1. Download the raspberry os iso file from <https://www.raspberrypi.com/software/raspberry-pi-desktop/> here .
2. Open the VirtualBox and create new virtual machine



1. After creating the new virtual machine,follow the step below to choose the iso file downloaded before.



1. Start the raspberry pi os and install the the os in the bios

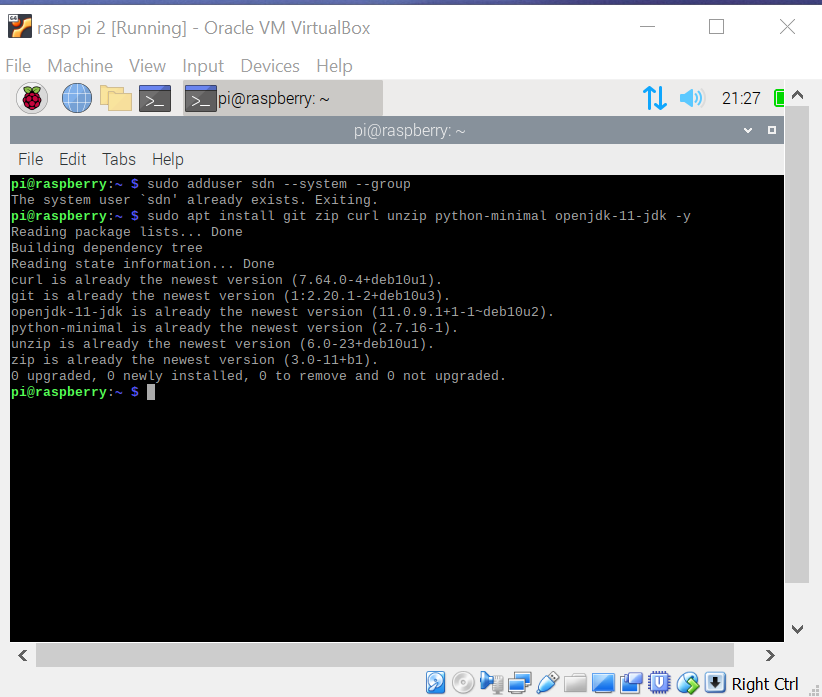
****

1. Restart the virtual machine and done.

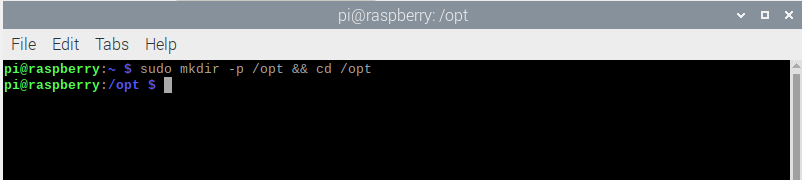
## 

### 6.4.4 Installation of ONOS

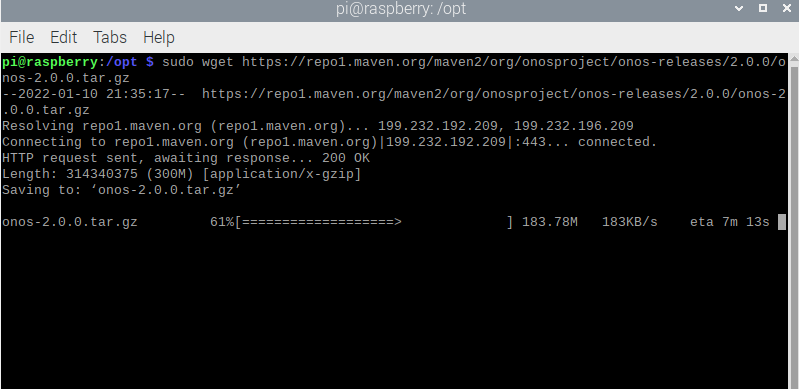
1. Open Network Operating System (ONOS) is the controller that will be used as a control plane to control our SDN network. The ONOS controller will be installed in the Raspberry Pi virtual machine. First, we need to add a user for ONOS installation using command *sudo adduser sdn –system –group*
2. Install Java JDK 11. In our case, our java jdk is already up-to-date.



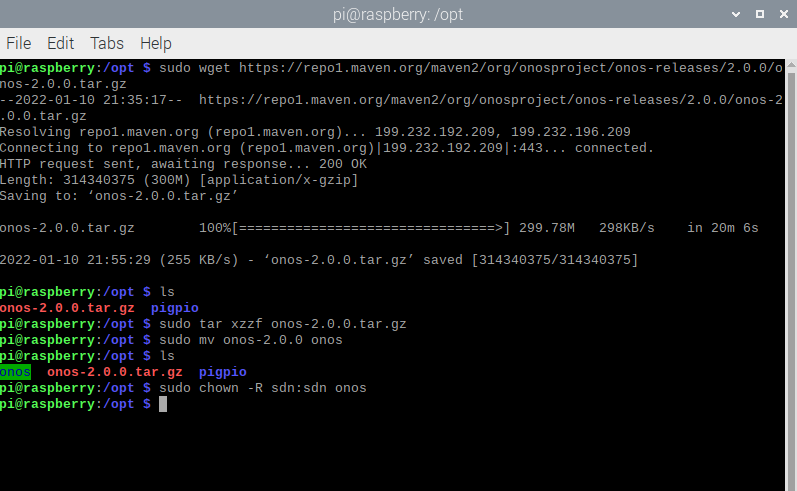
1. Create a directory “sudo mkdir -p /opt” as we will download and save the ONOS file in this directory.

****

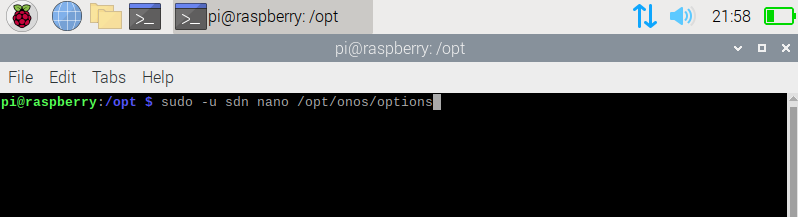
1. Next, we will download the onos version 2.0.0 through this link, [*https://repo1.maven.org/maven2/org/onosproject/onos-releases/2.0.0/onos-2.0.0.tar.gz*](http://repo1.maven.org/maven2/org/onosproject/onos-releases/2.0.0/onos-2.0.0.tar.gz) . The file size is around 300MB and it will take several minutes to download it.

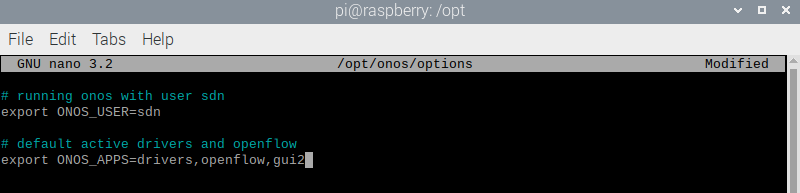


1. After finishing the download, we need to unzip the file, change the filename to *onos*, and create the ownership to the user that we created in the first step.

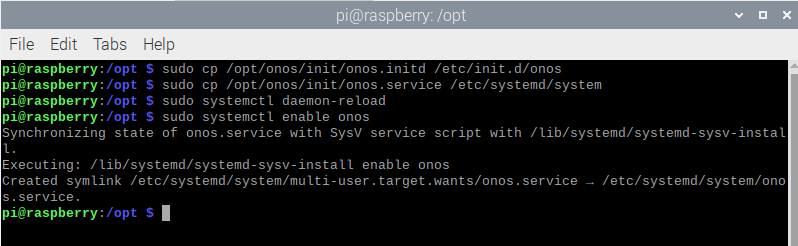


1. Next, we will set some settings for the startup of the ONOS. Go to directory */opt/onos/options* and add the scripts below, to run onos with user sdn, and set the default active drivers and openflow.

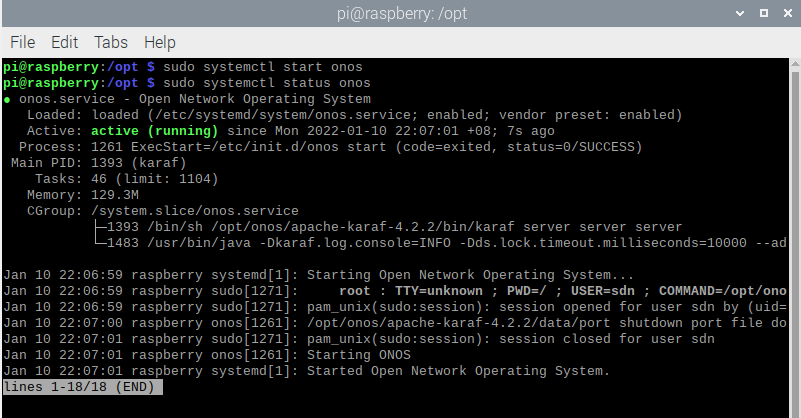




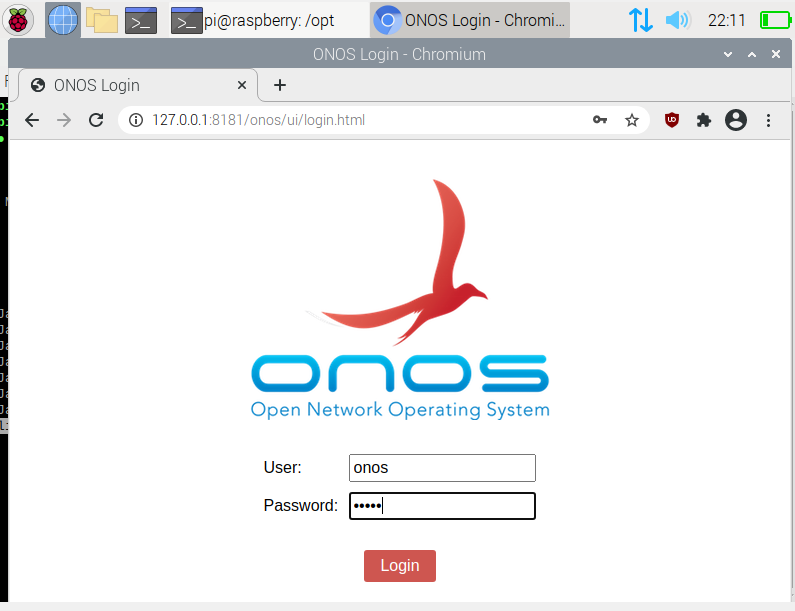
1. Install service file as below. Reload and enable onos, with this we already finish the installation of onos.

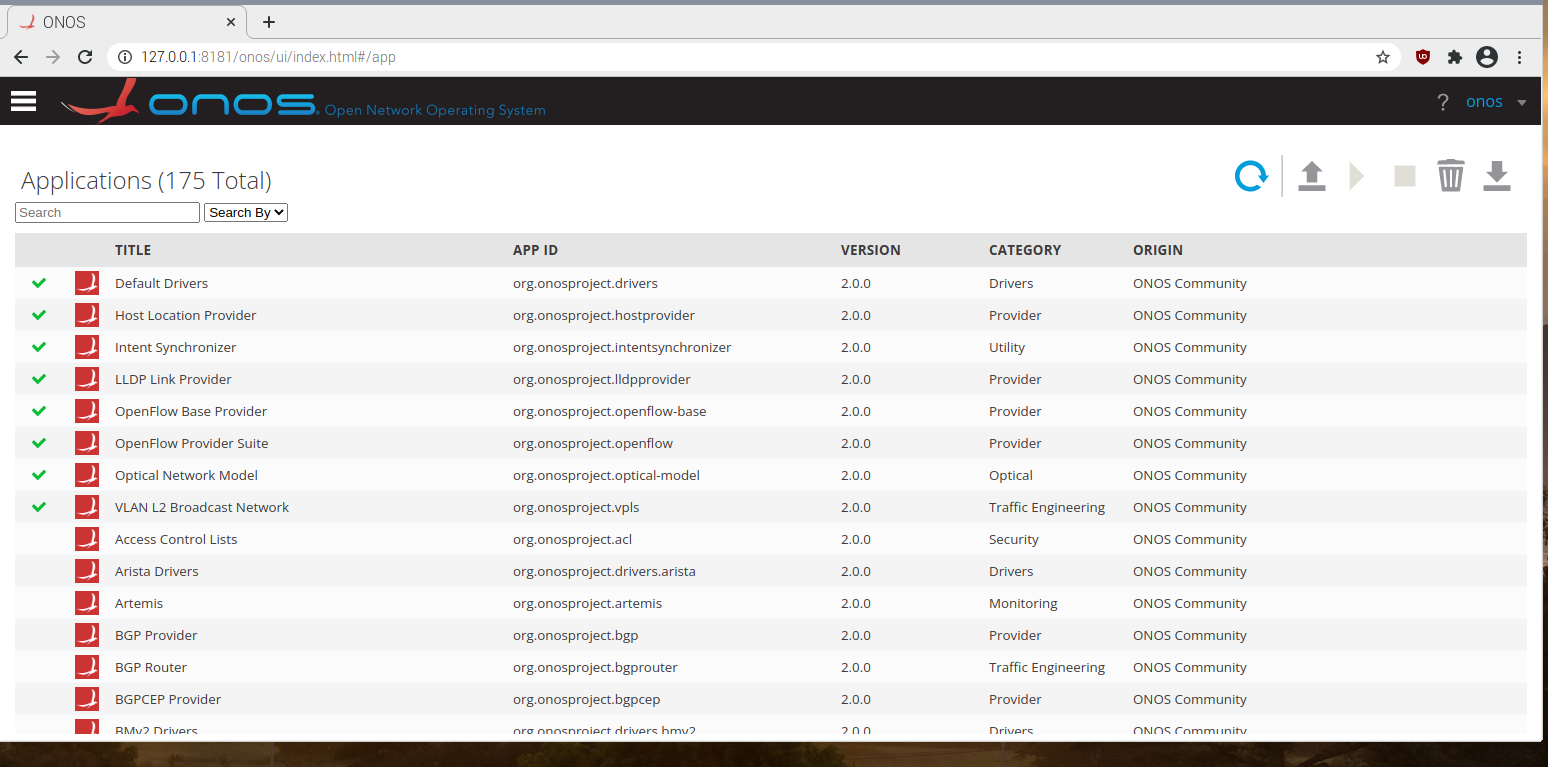


1. Start the ONOS controller using *sudo systemctl start onos.* Check the status of ONOS either running or not using *sudo systemctl status onos.*



1. Run ONOS in the GUI version. The address is 127.0.0.1, using port 8181, and navigate to the login page, *127.0.0.1:8181/onos/ui/index.html.* The username is onos and password used is rocks as defaults. In the application some of the features has been activated as we already set at the startup options.

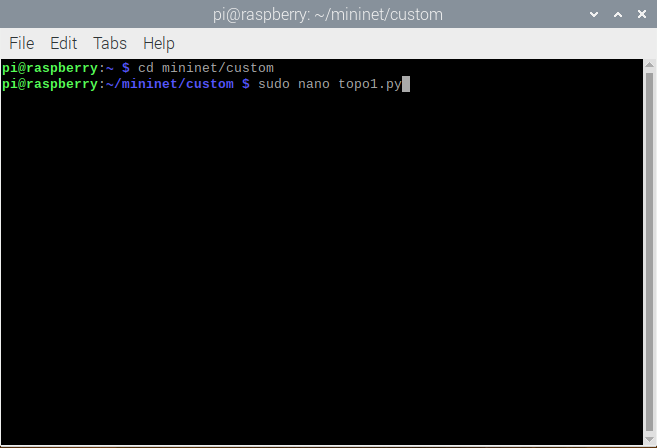




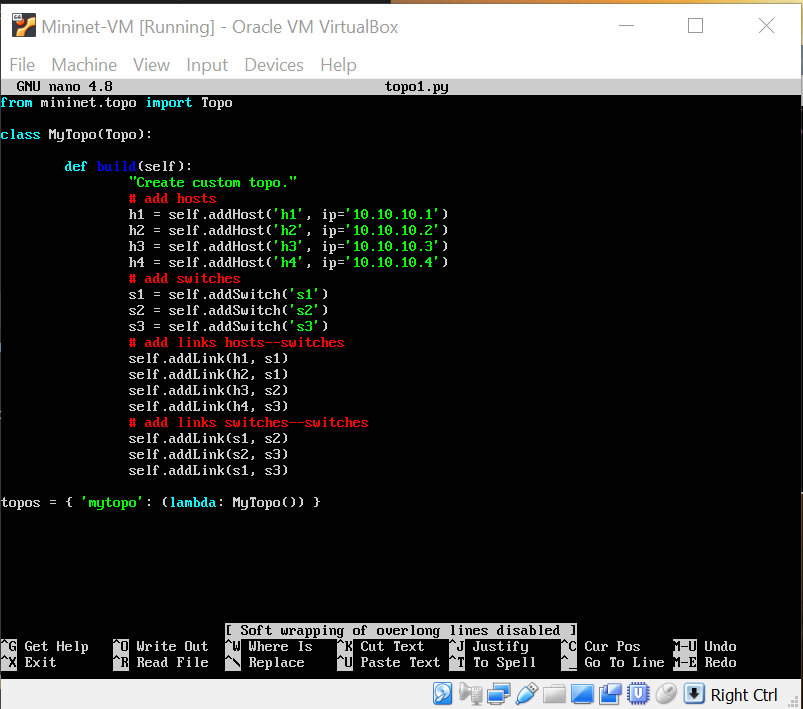
## 

### 6.4.5 Configuration of topology in mininet

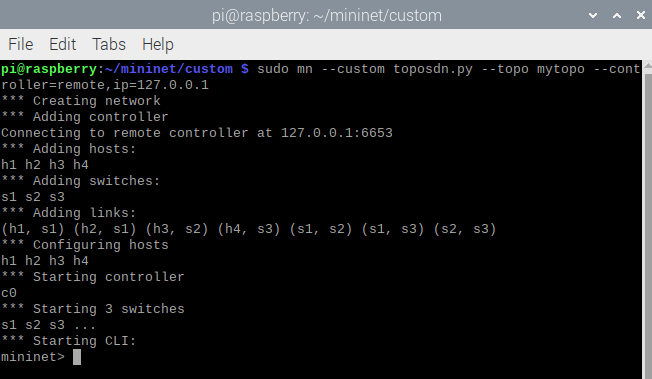
1. Go to directory mininet/custom
2. Create a python file as topo1.py with command “sudo nano topo1.py”



1. Write the python script for our topology. Here is our code.

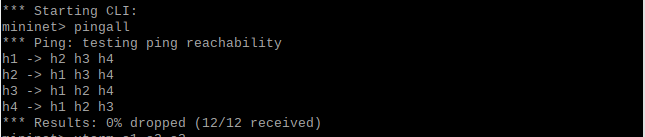


1. Then connect the topology with the ONOS controller using **sudo mn –custom topo1.py –topo mytopo –controller = remote,ip = 127.0.0.1** command.

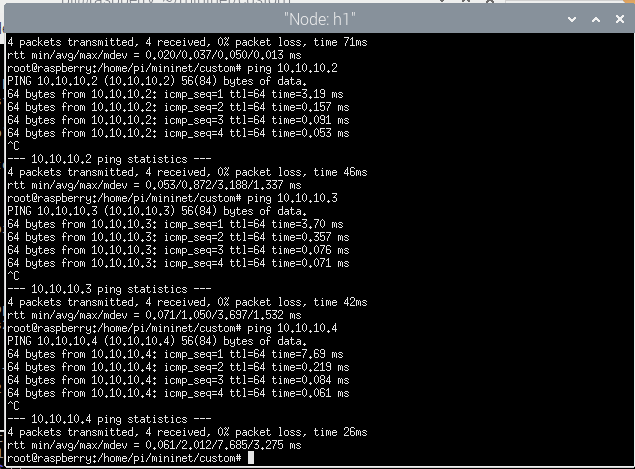


## 6.5 Testing connectivities

1. Trying to check the connectivity of all the hosts by using command ‘**pingall**’ in mininet. All the hosts are connected because 12/12 packets are received.



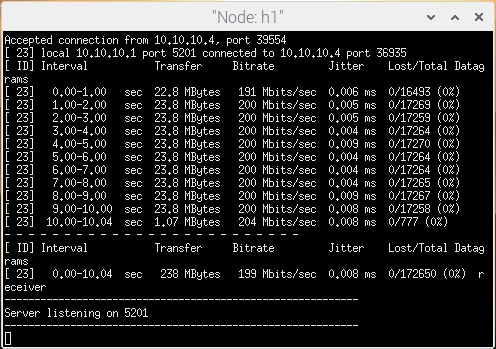
1. Open **xterm** for h1 and try to ping h2 (10.10.10.2) , h3 (10.10.10.3), and h4 (10.10.10.4). All ping is successful.

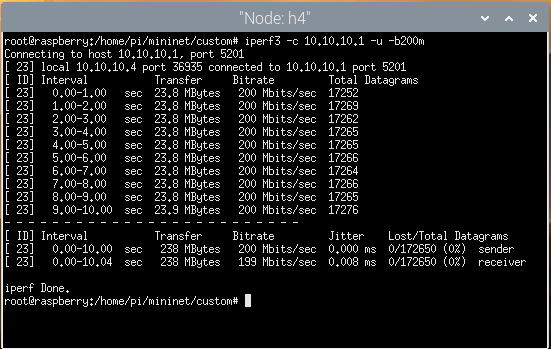


# 7. Performance evaluation

## 7.1 Scenario 1: UDP protocol with bandwidth 200 Mbits/s

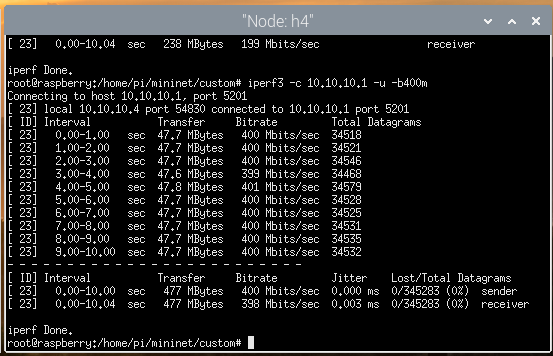
1. For the performance evaluation, we will use iperf3. To install iperf3, the command is “sudo apt install iperf3”, and it is already ready to use.
2. The Iperf3 server will be up at host1 and ready to listen at port 5201 by default.
3. First, at host 4, an iperf3 client with UDP protocol connected to the iperf3 server (10.10.10.1) with the bandwidth 200 Megabits/sec will send the packet in 10 seconds interval time.





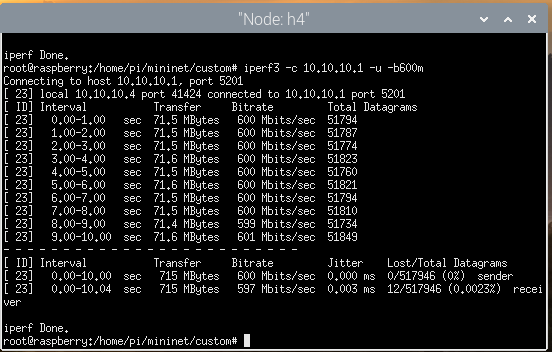
## 7.2 Scenario 2: UDP protocol with bandwidth 400 Mbits/s

1. The packet is the same as before but with different bandwidth. At host 4, an iperf3 client with UDP protocol connected to the iperf3 server (10.10.10.1) with the bandwidth 400 Megabits/sec will send the packet in 10 seconds interval time.



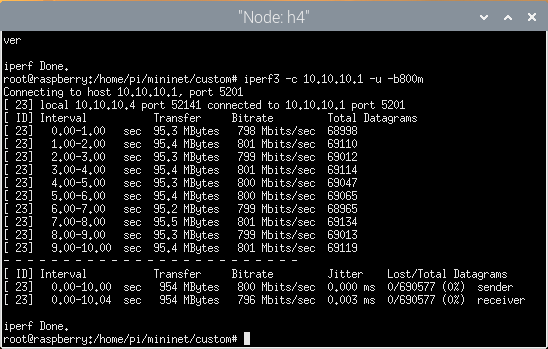
## 7.3 Scenario 3: UDP protocol with bandwidth 600 Mbits/s

1. The packet is the same as before but with different bandwidth. At host 4, an iperf3 client with UDP protocol connected to the iperf3 server (10.10.10.1) with the bandwidth 600 Megabits/sec will send the packet in 10 seconds interval time.



## 7.4 Scenario 4: UDP protocol with bandwidth 800 Mbits/s

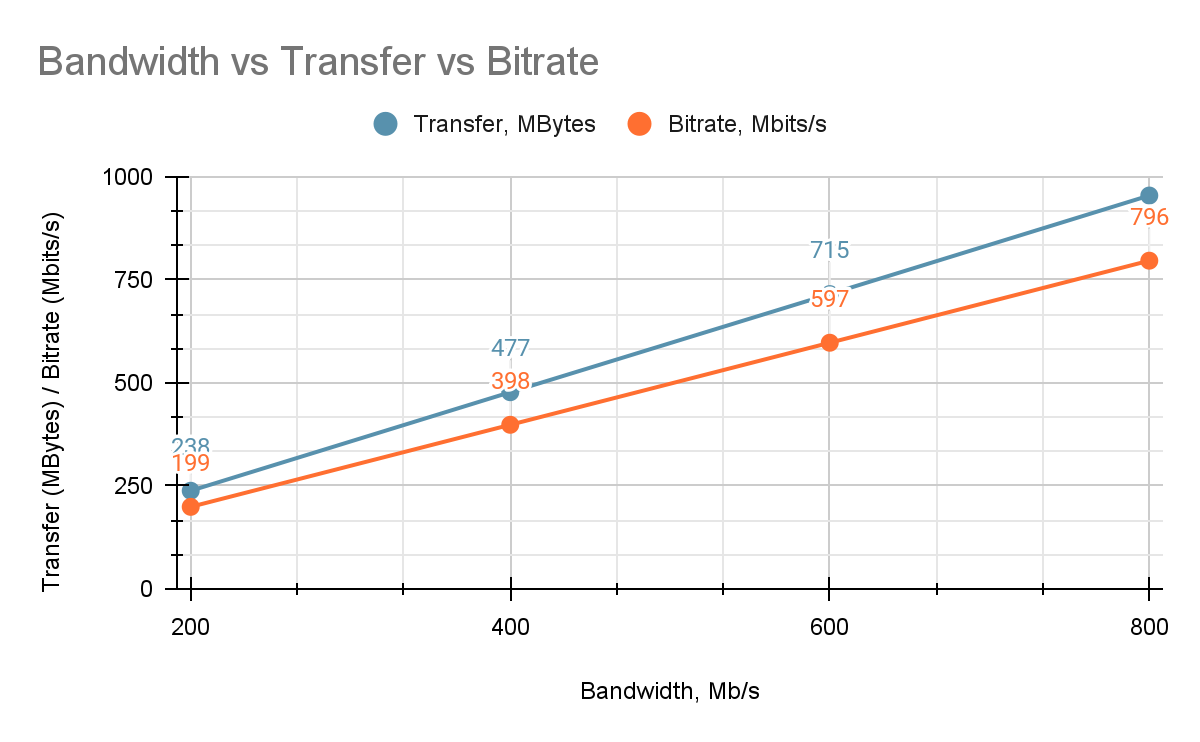
1. The packet is the same as before but with different bandwidth. At host 4, an iperf3 client with UDP protocol connected to the iperf3 server (10.10.10.1) with the bandwidth 800 Megabits/sec will send the packet in 10 seconds interval time.



## 7.5 Results

UDP protocol, interval time = 10 seconds

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Bandwidths, Mb/s | Transfer, MBytes | Bitrate, Mbits/s | Jitter, ms | Packet Loss, % |
| 200 | 238 | 199 | 0.008 | 0 |
| 400 | 477 | 398 | 0.003 | 0 |
| 600 | 715 | 597 | 0.003 | 0 |
| 800 | 954 | 796 | 0.003 | 0 |



# 8. Conclusion

1. We are able to recreate most of what are in the article chosen
2. The performance of the SDN network using Raspberry Pi is good as there is no packet loss and low jitter.
3. We are successfully pinging the entire node using the controller after connecting to the ONOS.

# 

# 9. References

Kurniawan, R. Z.

F. Ramadhan, R. Primananda, and W. Yahya, *“Implementasi Routing Berbasis Algoritme Dijkstra Pada Software Defined Networking Menggunakan Kontroler Open Network Operating System,”* J. Pengemb. Teknol. Inf. dan Ilmu Komputer; Vol 2 No 7, vol. 2, no. 7, pp. 2531–2541, 2017.

Ummah, *“Perancangan Simulasi Jaringan Virtual Berbasis Software-Define Networking,”* Indones. J. Comput., vol. 1, no. 1, pp. 95–106, 2016.

T. I. Bayu and E. E. Tahan, *“Software Defined Network (Sdn) Simulation Concept Using Raspberry Pi,*” J. Terap. Teknol. Inf., vol. 2, no. 2, pp. 1–11, 2018.

M. W. Putra, E. S. Pramukantoro, and W. Yahya, *“Analisis Perbandingan Performansi Kontroller Floodlight, Maestro , RYU , POX Dan ONOS Dalam Arsitektur Software Defined Network ( SDN ),”* J. Pengemb. Teknol. Inf. dan Ilmu Komput., vol. 2, no. 10, pp. 3779–3787, 2018.

D. S. R. Sari, R. Munadi, “Analisis Performansi Segment Routing Pada Software Defined Network Menggunakan Kontroler ONOS,” vol. 6, no. 2, pp. 3469–3476, 2019.

P. Monika, R. M. Negara, D. D. Sanjoyo, F. T. Elektro, U. Telkom, and L. I. Monitor, “Analisis Performansi Jaringan Software Defined Network Menggunakan Metode Intent Monitor And Reroute ( IMR ) Pada Controller ONOS,” 2019.

Kurniawan, R. C., Tulloh, R., & Irawati, I. D. (2021). VPLS on software defined network using ONOS controller based on raspberry-pi 3. *2021 IEEE Asia Pacific Conference on Wireless and Mobile (APWiMob)*. https://doi.org/10.1109/apwimob51111.2021.9435235