

Performance Evaluation of Different Controllers in SDN using Mininet Emulator

Ms. Florance G¹ and Dr R. J. Anandhi²

¹Computer Science and Engineering, New Horizon College of Engineering, Affiliated to VTU, Bangalore, Karnataka
Email: vijiflorance59@gmail.com

²Information Science and Engineering, New Horizon College of Engineering, Affiliated to VTU, Bangalore, Karnataka
Email: hod.ise@newhorizonindia.edu

Abstract—An innovative technique in network to monitoring and managing their infrastructure is Software Defined Network (SDN). This is targeted by partitioning the control and data plane of the network by programmable SDN controller. All the intelligence of SDN resides inside the controller. With the help of Open Flow Protocol, communications between the planes are controlled. The core idea of Open Flow is to control the routers or switches through programming from the centralized controller. The main benefit of SDN is to ensure that all logical decisions are taken at centralized way to reduce cost, obtain a better scalability and improve easier network management. This paper focus on main challenging issue is how the controller monitoring heavy data flow in the network and also analyzing the parameters as throughput and delay are compared with POX and Ryu controllers in mininet simulation.

Index Terms— Software Defined Network, Open Flow protocol, POX Controller, Ryu Controller, Mininet, Centralized Network.

I. INTRODUCTION

Traditional computer network is made up of physical component, and it is very difficult to adopt into dynamic characteristics. This will provide inconvenient when user need to change the requirement. Tightly coupled devices are not flexible to install the software based on the user requirements. Whenever the requirements getting change, administrators' individually change the software in each device (routers, firewalls, switches) and each hardware has autonomous software [1]. This is used to control the behavior of that hardware. On the other side, software is preferred over hardware in SDN. It also has a centralized controller, which provides more responsibility to manage network with dynamic, flexible, automated, and manageable architecture. The SDN controllers generally available are Cisco XNC, NEC Trema and Nicira NVP etc.

Software Defined Networking (SDN) provides network services to the user to do optimization and customization in the network environment. The important functionalities of SDN framework are programmable, centrally managed, and programmatically configured [2]. In order to increase the performance of the control plane, it is necessary to have a controller with high accuracy and decision is regulated to the forwarding plane to increase the capacity to handle the network traffic challenges.

The contrasting features of the two SDN OpenFlow controllers (OpenDayLight and NOX), where the Mininet emulator helps in distinguishing the performances of the OpenFlow controllers. The controllers NOX, Floodlight, POX, Ryu, and Beacon are differentiated with respect to dependability, safety, scalability and to

obtain different performing qualities (reaction time) and framework evaluation is found [1]. To the extent of my knowledge, not even one of these controllers are assessed for their performance for example, throughput, latency (delay), and for general data transfer in networking framework. When comparing Floodlight, NOX, Beacon, OpenDayLight controller, POX and Ryu controller produce better performance. The main motivation of this research article is to give a distinct investigation of two controllers (POX and Ryu controller) with respect to data transfer among the network environment. I also found few features of network performance like latency and throughput, for differentiating controllers [10].

This paper is set in the following pattern: Section II deals with the progress of how OpenFlow protocol used in SDN framework. Section III further talks about SDN controllers (POX, Ryu) and Mininet emulator. Section IV compares the results after evaluating performance found in two controllers in the III section. Section V finally concludes the paper.

II. SDN AND OPENFLOW PROTOCOL

SDN framework is primarily dependence on partitioning the control and data plane separately.

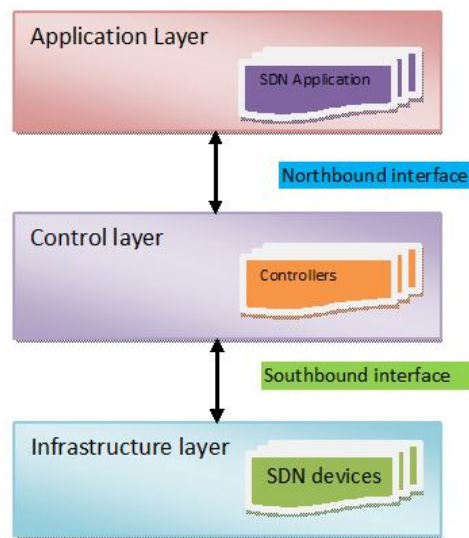


Figure 1. SDN Layered architecture

The framework consists of three important layers such as Application Layer, Control Layer, and Infrastructure Layer [3], [11] are shown in Fig 1.

- The prominence of the infrastructure layer is, to transfer data among the SDN devices, which helps to carryout packet switching and forwarding. The role of the Flow table is to record the rule by the controller and checking the data packets. The mapping table helps in storing communication among the SDN network.
- Control Layer is one of the prominent layer where the SDN controller is situated, which is used to control, manage, and monitor the entire behaviour related to SDN devices present in the data plane. In general, controllers are software, which provide the overall view of the whole network framework.
- The uppermost layer is application Layer, where network operator does the operation: The entire framework will be rapidly configured dynamically at this layer using SDN programs that are customized.

The communication is carried out in all three layers by using Application Program Interfaces (APIs) such as, southbound API and northbound API.

- Southbound API: Control plane directly communicate to data plane by using southbound interface. Using this interface, control plane can dynamically provide instructions to the data plane regarding their operations.
- Northbound API: Application layer mediates directly with the control layer by northbound interface. Using this interface administrator can rapidly give commands to the controller regarding the adjustments made in SDN devices at infrastructure layer.

A. Data transfer using SDN OpenFlow protocol

SDN OpenFlow protocol is a standard protocol used to provide mediation in between controller and data plane respectively. When packet arrive at OpenFlow switch, it forward back to controller to handle. Controller instructs the switch to insert flow input into the flow table of switch by using OpenFlow protocol. So that switch is entirely dependent upon controller resolution [1].

Fig 2 shows the data transfer in SDN network. In this one OpenFlow switch is latched to 3 hosts (host1, host2 and host3) and also latched to a SDN controller. Suppose, if the host 1 is willing to transfer data to host 3, first request will be direct to the switch. As switch is unable to handle the situation with this request (consider flow table is clear) and it will redirect to the controller to do required actions (discard or forward). The dashed line indicates that the request was directed to only controller not host 3. Once the request is assessed by controller, it is directly redirected to the host 3.

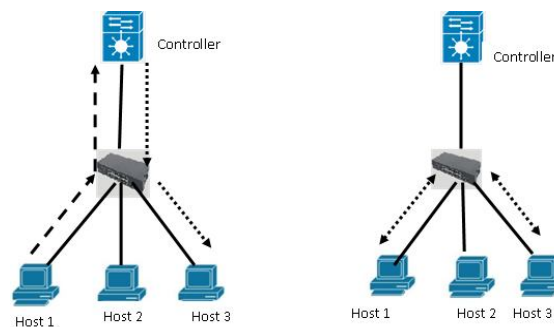


Figure 2. Data transfer in SDN

Controllers insert the flow input into the flow table of switch, commands the transfer of the request to host 3. The dotted line indicates the request forwarded from controller to host 3 through switch. After certain time if host 1 wishes to transfer data to host 3, first requests is send to the switch, this in turn is forwarded to host3 without controller decision, as flow entry already there in flow table and the flow is indicated as dotted line.

III. CONTROLLERS IN MININET EMULATOR

In SDN network multiple elements are needed to its environment [7]. At beginning we require controller carrying a network operating system, as POX and data forwarding plan are necessary to separated from the control plan with programming interface to provide efficient network management [6]. When researcher spend high cost to conduct their experiments is very difficult, to deal with this problem you can take the help of special simulator [4].

Mininet are popular open source network emulator suitable for creating networks of SDN elements includes Hosts, Switches, Controllers and Links that gives space for prototype development of large scale SDN network using only one computer. It allows the creation of an OpenFlow based SDN on an individual computer using Linux processes in the network [9].

A. Creating network

Single command used to build SDN network in Mininet [3].

```
$ sudo mn—topo linear, 3, 2
```

Virtual network is created with three switches and two hosts connected linearly, each host configured to respective switch.

B. Interacting with a network

In Mininet, from a single console the entire network can be managed and controlled [3], [4]. For example,

Mininet>h1 ping -c3 h2 → Used to ping h2 from h1

Mininet> nodes → List the set of nodes available.

Mininet>pingallfull →How hosts are connected and describe the detail of maximum, minimum and average time among two host.

Mininet>dump →Display the internet protocol address and process ID of host.

Mininet>net →Display the link available in the network.

Mininet>iperf → Test bandwidth among hosts.

C. POX controller and Ryu controller

The essential part of SDN controller is responsible for concentrating communication and control the behavior of all SDN devices of the network at infrastructure layer, giving a general perspective of the network. It backs the OpenFlow protocol to regulate with bottom layers instead of a secure channel. Nowadays there are many SDN OpenFlow controllers are shown in table I.

TABLE I. TYPES OF SDN CONTROLLERS

Controller Name	Language to be used
NOX	C++
POX	Python
Ryu	Python
Beacon	Java
Floodlight	Java
OpenDayLight	Java

POX Controller is the prime controller designated to facilitate SDN framework which is more effective than NOX OpenFlow controller which facilitates the OpenFlow version 1.0 [5], [8].

Ryu controller has been executed to construct effective SDN network with OpenFlow protocol [1]. This showed low bit rate, less average delay and high throughput when regulating the activities of the network. It is capable in assessing synopsis like, forwarding arrived packets to their destination, insert flow entry into flow table of switch, and commanding repeated packet will be managed by switches in the network.

IV. RESULT AND ANALYSIS

Mininet is popular open source settings to simulate the SDN to scrutinize the parameters of networking such as throughput and delay. In this research article, the characteristics of Ryu and POX controllers are assessed by forming a SDN framework which is installed in a virtual box. Once the network has been created, controller (POX and Ryu controller) is capable to distantly connect to a Mininet VM. Mininet application of the modeling sequence is shown in Fig 3.

```
mininet@mininet-vm:~$ sudo mn -topo linear,3,2
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2
*** Adding switches
s1 s2 s3
*** Adding links
(h1, s1) (h2, s1) (h1, s2) (h2, s2) (h1,s3) (h2, s3)
*** Configuring hosts
h1 h2
*** Starting controller
c0
*** Starting 3 switches
s1 s2 s3
*** Starting CLI:
```

Figure 3. 3 switches, 2 hosts linear Topology

It was obtained that, SDN network is developed with controller POX and Ryu, proper pings are received. Pingallfull is a command uses ICMP protocol to evaluate the connectivity amidst the set of host. Its role is to send packets to the destination host shows the pingallfull results from Mininet. It is scrutinizing bandwidth among the host using TCP and displays the quality of network with respect to bandwidth is shown in Fig 4.

The bandwidth is analyzed and shows in Fig 5, that the comparison of obtained bandwidths between two host using POX and Ryu controller. The assessment between hosts, the bandwidth was responsible majorly for functioning of controllers. By the time packet reaches, it already is aware of the path which gives the ascendance of the bandwidth between the hosts, instead of choking all the bands with packets to find its destination.

```

mininet> pingallfull
*** Ping: testing ping reachability
h1 -> h2 h3
h2 -> h1 h3
h3 -> h1 h2
*** Results:
h1->h2: 1/1, rtt min/avg/max/mdev 2.556/2.556/2.556/0.000 ms
h1->h3: 1/1, rtt min/avg/max/mdev 1.491/1.491/1.491/0.000 ms
h2->h1: 1/1, rtt min/avg/max/mdev 1.190/1.190/1.190/0.000 ms
h2->h3: 1/1, rtt min/avg/max/mdev 1.398/1.398/1.398/0.000 ms
h3->h1: 1/1, rtt min/avg/max/mdev 1.928/1.928/1.928/0.000 ms
h3->h2: 1/1, rtt min/avg/max/mdev 1.460/1.460/1.460/0.000 ms
mininet>

```

Figure 4. Testing ping in ms

```

mininet> iperf
*** Iperf: testing TCP bandwidth between h1 and h3
*** Results: ['28.4 Gbits/sec', '28.5 Gbits/sec']
mininet>

```

Figure 5. Testing bandwidth

In analyzing functionality of the controller during network handling, the parameter latency (Delay) and throughput is obtained. SDN network is created with 5, 15, 25 and 35 hosts by using linear topology, Ryu controller is 90.3 times slower than POX controller. As a result, Ryu will be 90.3 times slower than POX in terms of latency. Similar way calculate the throughput using Ryu controller is 95.9 times faster than POX controller, which says Ryu controller is better performance measure compared to POX controller.

The performance measure of SDN network in linear topology with respect to delay and throughput are obtained using Ryu and POX controller are shown in Fig 6 and Fig 7. In this comparative study shown the graph for parameters throughput and delay is determined when heavy congestion occur in the network. I have developed the network with different set of nodes. When the nodes are getting increases delay and throughput are changing based on the POX and Ryu controllers in mininet shown in Fig 6 and Fig 7.

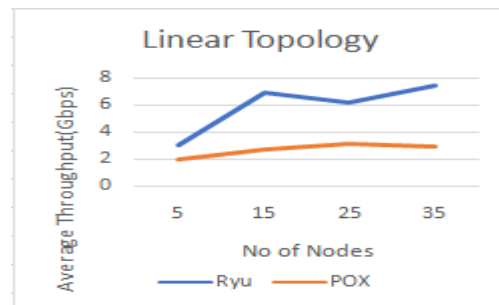


Figure 6. Average Throughput in Linear Topology

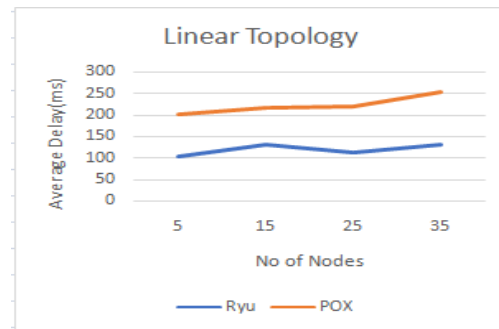


Figure 7. Average Delay in Linear Topology

V. CONCLUSION

Managing and monitoring the network devices by using SDN controllers in programmable way has increased ahead of the probability. In this research article SDN network has been designed and simulated by using two

controllers such as Ryu and POX controller in Mininet. The performance parameters (latency and throughput) have been measured when number of host is getting changed in network using different controllers. When compare to throughput and delay in Ryu controller, it provides tremendous good value than POX controller. Therefore simulation carried out using RYU controller is 95.9 times faster than the POX controller in terms of throughput and 90.3 times slower than POX controller in terms delay (latency) is analyzed in mininet stimulation.

REFERENCE

- [1] Rastogi and A. Bais, "Comparative analysis of software defined networking (SDN) controllers — In terms of traffic handling capabilities," 2016 19th International Multi-Topic Conference (INMIC), 2016, pp. 1-6.
- [2] Fancy and M. Pushpalatha, "Performance evaluation of SDN controllers POX and floodlight in mininet emulation environment," 2017 International Conference on Intelligent Sustainable Systems (ICISS), 2017, pp. 695-699.
- [3] Florance G and R. J Anandhi, "Study on SDN with Security Issues Using Mininet," *Advances in Parallel Computing*, Dec. 2021, vol. 39, pp. 104-113, in IOS Press.
- [4] Karamjeet Kaur, Japinder Singh and Navtej Singh Ghumman, "Mininet as Software Defined Networking Testing Platform", *International Conference on Communication, Computing & Systems At: SBS State Technical Campus, Punjab, India*, vol. 1, Aug 2014.
- [5] Ligia Rodrigues Prete, A. A. Shinoda, C. M. Schweitzer and R. L. S. de Oliveira, "Simulation in an SDN network scenario using the POX Controller," 2014 IEEE Colombian Conference on Communications and Computing (COLCOM), 2014, pp. 1-6.
- [6] R. Jmal and L. Chaari Fourati, "Implementing shortest path routing mechanism using Openflow POX controller," *The 2014 International Symposium on Networks, Computers and Communications*, 2014, pp. 1-6.
- [7] R. L. S. de Oliveira, C. M. Schweitzer, A. A. Shinoda and Ligia Rodrigues Prete, "Using Mininet for emulation and prototyping Software-Defined Networks," 2014 IEEE Colombian Conference on Communications and Computing (COLCOM), 2014, pp. 1-6.
- [8] S. Ramadona, B. A. Hidayatulloh, D. F. Siswanto and N. Syambas, "The simulation of SDN network using POX controller: Case in Politeknik Caltex Riau," 2015 9th International Conference on Telecommunication Systems Services and Applications (TSSA), 2015, pp.1-6.
- [9] S. Wang, "Comparison of SDN OpenFlow network simulator and emulators: EstiNet vs. Mininet," 2014 IEEE Symposium on Computers and Communications (ISCC), 2014, pp. 1-6.
- [10] Sm Shamim, Shahadat Shisir, Ahosanul Hasan, Mehedi Hasan & Arafat Hossain, "Performance Analysis of Different Openflow based Controller Over Software Defined Networking", : *Global Journals*, 2018, vol. 18.
- [11] M. Shin, K. Nam and H. Kim, "Software-defined networking (SDN): A reference architecture and open APIs," 2012 International Conference on ICT Convergence (ICTC), Jeju, Korea (South), (2012), pp. 360-361.