International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 08 Issue: 10 | Oct 2021 www.irjet.net p-ISSN: 2395-0072

Performance Evaluation of RYU SDN Controller Using Mininet

Nafees M Kazi[1], Dr S R Suralkar[2], Dr Umesh S Bhadade[3]

[1]SSBTs College of Engineering & Technology, Bambhori, Jalgaon
[2]SSBTs College of Engineering & Technology, Bambhori, Jalgaon
[3]Research Guide, KBC North Maharashtra University, Jalgaon

Abstract - —*Software Defined Networks (SDN) has attracted the researchers and industry due to their flexibility and programmability. SDN has been differentiated from traditional networks in terms of separation of the control plane and forwarding functions. The forwarding decisions are sent by the controller to switches and routers. The switches are responsible only for logical forwarding of the packets. Hence performance of any SDN network depends on the performance of the controller. Lot of SDN controllers are available. In this paper we have evaluated the performance of two well-known python base SDN controller RYU. Mininet is used as the simulation tool. The performance is evaluated for linear topology, tree topology and datacenter topology with varying scales. We have used D-ITG for performance evaluation. Iperf is also used for measuring the maximum available bandwidth. RYU controller performs better in terms of average delay, jitter, bitrate and throughput. The selection of the controller depends on the application requirements.*

Key Words*: RYU, SDN, Mininet, D-TIG, Iperf,API*

1. Introduction

The aim of Software-defined networking (SDN) is to make networks agile and flexible. The network control is improved in SDN. SDN has been successful in satisfying the changing needs if the enterprises and service providers. The network engineer or administrate is able to change the traffic from control plane without touching individual switches in the network. Switches are directed by the centralized SDN controller for delivering the services as per requirement. In the traditional networks, individual network devices were making the traffic decisions based on their routing

1.1 SDN architecture

Figure 1 shows the SDN architecture. As depicted in the figure, SDN architecture is divided into three layers: the application layer, the control layer and the infrastructure layer.

The application layer contains the typical network applications or functions organizations use. The applications are intrusion detection systems, load balancing or firewalls. In the traditional networks a specialized module is used for firewall or load balance. In SDN the separate appliance is replaced by the centralized controller which is responsible for management of the data plane.

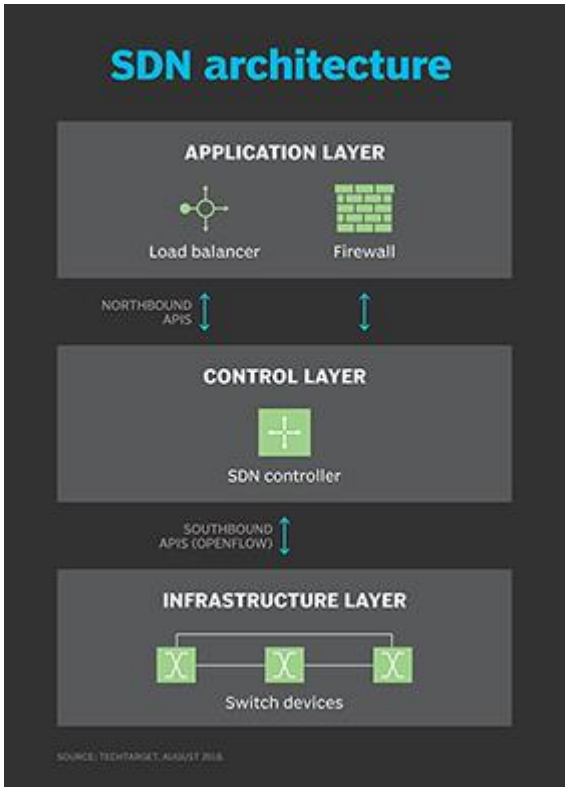

Figure 1: SDN Architecture

The three layers of SDN architecture are communicating through northbound and southbound APIs.

The controller is the brain of the SDN. It resides in the control plane. The controller is responsible for management of policies and network flow. The controller lies in the server. The physical switches in the network lies in the infrastructure layer. The communication between these three layers occurs using respective northbound and southbound application programming interfaces (APIs). Northbound interface is used for communication of applications to the controller. The southbound API are used for communication between controller and switches. Open Flow protocol is an example of southbound protocol

1.2 How SDN works

Initially the major focus of SDN was the separation of control plane and data plane. The control plane is responsible for decision making about packet forwarding through the network. The data plane is responsible for logical packet forwarding.

In the classic scenario of SDN, on arrival of the packet, the rules are built on the switch's proprietary firmware were telling the switch regarding packet forwarding. The centralized controller was sending these rules to the switch.

The switch is known as data plane device. It requests the controller for the guidance and provides with the traffic. Every packet having the same destination is treated as the same by the switch and sent along the same path.

In adaptive or dynamic mode of SDN, route request is sent to the controller by the switch for the packet which is not having any specific route. Adaptive routing is different from this. In adaptive routing the route request is dine through routers as well as algorithms which are topology specific. It is not sent through controller.

1.3 Benefits of SDN

SDN offers several advantages over traditional networks as below:

1. Administrators are able to change the rules including prioritization or blocking specific packets.
2. The role of the network administrator is to deal with the centralized controller and distribution of packets to the switches. He is not needed to configure the multiple devices connected to the network.
3. Due to this, the controller can monitor the traffic and security can be deployed.
4. The controller is able to reroute or drop the packet if found suspicious.
5. Using SDN, previously dedicated hardware can be virtualized. Hence the operational cost can be reduced.
6. Software-defined wide area network (SD-WAN) technology is another important aspect of SDN

1.4 Challenges with SDN

1. Although security can be achieved through the centralize controller, it is also a cause of concern. The centralized controller becomes the single point of failure.
2. There is no specific definition of SDN. Different vendors design different approaches according to their requirements.

I Related work

We have found the need to change the network architecture from root. SDN is the new changing environment for networks where control plane is separated from forwarding plane. In this paper authors have presented the introduction of SDN along with the architecture, application, different controllers, comparison and limitations of the SDN networks [1]

SDN has introduced the programmability and flexibility into the network by decoupling of control plane from data plane. SDN controller is the important network entity which allows to setting the policies of the network. In the paper [2] authors have compared the performance f new and different SDN controllers.

In [3] authors have compared SDN networks based on security, openflow, mininet, floodlight controller and virtual switches. Different virtual security functions are implemented. Hence network security is increased by avoiding loops and broadcasting. Availability of the network is increased by avoiding loops and eliminating different attacks such as congestion driven attacks, distributed denial of service attacks etc.

In SDN the traditional network is split into centralized control plane and programmable data plane. The controller provides flow to switches and optimizes the network performance. Hence the performance of the controller directly affects the performance of SDN controller. Also, the benchmarking tool to evaluate the performance must be reliable. In [4] authors have presented a comprehensive qualitative comparison of different SDN controllers. Also, quantitative performance is measures. 34 controllers are categorized and compared three benchmarking tools are used to compare 9 controllers.

SDN applications are mostly depend on controllers. Authors in [5] focus on the problem of choosing the right controller. A new method for controller's benchmarking is proposed. The method is divided in two steps. First the controllers are classified according to their features. In the second step the controller performance is evaluated using CBENCH tool where on it three controllers form the first step are considered. The methods is tested on 10 controllers. Authors found the open daylight controller to respond better for all constraints and requirements.

In SDN the architecture has moved from traditional fully distributed model to a more centralized model. This approach is characterized by the separation of control plane and data plane. A more flexible network is created by using the controller to manage the flow. The controller is responsible for making the forwarding decisions and switches only performs the logical forwarding. [6]

In SDN the traditional network is distributed in control plane and forwarding plane. This approach is characterized by the separation of control plane and data plane. The control plane contains the controller and the data plane performs the forwarding including switches and routers. The success and failure of the SDN network depends on the controller. In [7] authors have discussed the basic concepts of SDN and compared existing controllers. Different parameters used for

comparing controllers are cost, efficiency, ability to support various protocols etc. The controllers are compared based on programming languages.

The advent of smart grid technology has promoted the upgradation of substation automation technology. It has lead to the addition of processing and communication capabilities for controlling and protecting devices. Hence research is done for supporting such scenario. SDN is able to provide the tools for fulfilling the needs at low costs. In [8] authors provide the survey and comparison of different SDN controllers available.

One of the essential part of the SDN is its controller. Lot of research has been done on numerous controllers as well as comparison is done. The paper [9] tests the newly arrived SDN controllers including ONOS, LibFluid based controllers etc. The benchmarking tool used for evaluation is CBENCH. Results depicts the MUL, Beacon to be the best controllers. But selection of the controller depends on the application and user requirements.

Software Defined Networks depends in the centralized controller. The routing intelligence is decoupled from forwarding functions. It is necessary to understand the performance of the controller before its use. But as lot of controllers are available for research, it becomes difficult to choose the appropriate one. In [10] authors have compared the performance of ONOS, RYU, Floodlight and Open Daylight controllers. The performance parameters used are latency and throughput. Cbench is used as benchmarking tool. Also, controllers are compared based on their features.

SDN is able to handle the data forwarding and control plane separately. Programmability has given the central importance in SDN. Most of the researchers are attracting towards SDN due to proprietary and open source. In [11] authors have compared these two strategies of SDN. They have identified the operating principles of both strategies, strengths and weakness.

2 Experimental Setup-

During this experiment we have used Mininet as software simulation tool. Different network topologies used are linear, tree and datacenter topology with varying scales. Different steps followed during simulation are as follows

1. Start the Mininet
2. Create different network topologies
   a. Create linear topology with 5 switches and varying number of hosts: We have increased the number of hosts from 25, 50, 75,100 per switch. The command to create the linear topology with fixed 5 switches and 25 hosts per switch is
      Sudo mn –topo linear, 25, 5 –controller remote
   b. Create linear topology with varying number of switches: We have increased the number of switches from 8,16,24,32,40,48,56,64,72,80. The command to create the linear topology the 16 switches and 5 hosts is
      Sudo mn –topo linear,5,16 –controller remote
   c. Create the tree topology with varying depth: We have created the tree topology with varying depth using following command
      sudo mn –topo tree,depth=6 –controller remote
      We have evaluated the tree topology performance for depth = 1,2,3,4,5,6,7.
   d. Create a datacenter topology: We have created the custom datacenter topology. In this topology, the switches and hosts are divided into racks. The centralized switch controls the flows in the racks along with the controller. The rank of the datacenter topology defined the number of switches and hosts in each rack. We have evaluated the datacenter topology for 4,5,6,7,10,12,14,16,18 and 20 ranks.
      The python script 'datacenter.py' is placed in mininet/custom folder. The command to run custom python script is
      Sudo mn –custom datacenter.py –topo mytopo –controller remote

3 Start the RYU remote controller

RYU is the python based SDN controller. In RYU , various software components are provided with well-defined API. It becomes easy for the developers to create different network management and control applications. Different protocols are supported by RYU for management of network devices including OpenFlow.

In this experiment we have used simple_switch.py application of RYU controller. The application is in RYU/app folder.

The command to start the RYU controller is

cd /home/ubuntu/ryu && ./bin/ryu-manager --verbose ryu/app/simple_switch.py

4 Performance evaluation

To evaluate the performance of the network we have used the D-ITG traffic generator. Different performance parameters considered are

1. Average delay
2. Average jitter
3. Average bitrate
4. Throughput

Throughput of the network is the maximum available bandwidth in the network. We have used the iperf for measuring the throughput of the network.

Table I depicts the performance of the linear topology with 5 switches and different number of hosts per switches using RYU controller.

Table I Linear Topology with 5 switches for RYU controller

Number of Hosts Per Switch	Linear Topology			
	Average Delay	Average Jitter	Average Bitrate	Throughput
25	0.00003	0.000078	7.928062	6.22
50	0.000137	0.000069	8.003869	6.41
75	0.000073	0.000068	7.983175	6.61
100	0.000046	0.000093	7.996148	6.42

As shown in table I the performance of the RYU controller is better than controller for linear topology with 5 switches. The average delay, Jitter and Bitrate are low for the RYU controller. The RYU controller gives better throughput as compared to POX controller.

Table II depicts the performance of RYU controller for different number of switches using linear topology. During this experiment the number of switches are increased from 8,16,24,32,40,48,56,64,72,80. Number of hosts connected to each switch are kept constant as 5.

TABLE II LINEAR TOPOLOGY WITH 5 HOSTS PER SWITCH FOR RYU CONTROLLER

Number of switches	Linear Topology			
	Average Delay	Average Jitter	Average Bitrate	Throughput
8	0.00313	0.000074	7.982405	4.47
16	0.000417	0.000101	7.97266	2.09
24	0.000446	0.000162	7.751627	1.45
32	0.000611	0.000156	7.890272	1.64
40	0.00072	0.000192	7.075748	808 MB/s
48	0.000782	0.000162	7.476058	738 MB/s
56	0.000957	0.000338	7.200725	416 MB/s
64	0.000919	0.000215	7.694275	518 MB/s
72	0.000987	0.000304	7.653575	447 MB/s
80	0.001092	0.000212	7.993848	243 MB/s

As shown in table II, the average delay, average jitter and average bitrate is low for RYU controller.

A tree topology with varying number of depth (increasing number of switches and hosts) is used. Tree topology with 2 layers is shown in the figure 2.

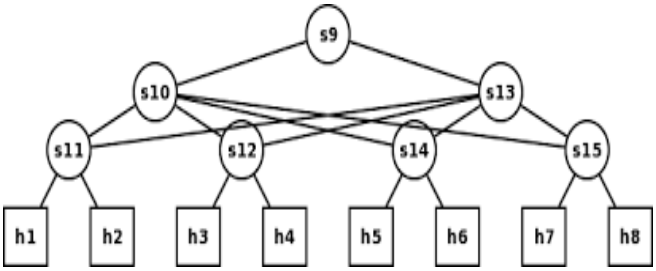

Figure 2: Tree topology with level =2

The tree topology is evaluated for varying depth from 1 ,2,3,4,5,6,7. Table V depicts the performance of the network using RYU controller.

TABLE III TREE TOPOLOGY PERFORMANCE USING RYU CONTROLLER

Number Layers	Tree Topology			
	Average Delay	Average Jitter	Average Bitrate	Throughput
1	0.000258	0.000078	7.986302	7.24
2	0.000271	0.000071	7.823257	7.51
3	0.000308	0.000121	7.995707	6.15
4	0.000339	0.0001	7.451616	5.06
5	0.000293	0.00018	7.982877	4.44
6	0.000372	0.000113	7.826859	4
7	0.000263	0.000098	7.980832	1.88

TABLE IV. DATACENTER TOPOLOGY USING RYU CONTROLLER

Number of racks	Datacenter Topology			
	Average Delay	Average Jitter	Average Bitrate	Through put
4	0.000317	0.000097	7.959375	5.94
5	0.000304	0.00009	7.996362	7.67
6	0.000324	0.000135	7.914234	6.84
7	0.000286	0.000085	7.883428	7.01
10	0.000305	0.000103	7.962812	7.32
12	0.00034	0.000162	7.705806	7.09
14	0.000298	0.000089	7.958661	7.56
16	0.000283	0.00008	7.987428	6.4
18	0.000313	0.000102	7.945782	6.18
20	0.000323	0.000131	7.713832	7.34

5. Conclusion

Software Defined Networks are differentiated from the traditional networks due to the separation of control plane from the forwarding functions. The controller in the control plane is responsible for making decisions about how to forward the packets. All instructions are given to the switches regarding packet routing. Decision regarding packet drop or packet forwarding in case of any suspicious packets is done by the controllers. Hence controllers are known as the brain of the SDN. Although lot of SDN controllers are available today, in this research we have used python based RYU SDN controller and evaluated the performance of the controllers in terms of average delay, average jitter and average bitrate along with throughput. D-ITG tool and iperf is used for measuring the performance of the network. The experiment is carried out using different scenarios. In the first scenario we have created the linear topology with 5 switches and varying number of hosts per switch. In second scenario we have used linear topology with 5 hosts per switch and varying the number of switches. In third case we have used tree topology with increasing depth. In the last scenario we have created custom datacenter topology with increasing number of switches and hosts. After evaluating the performance, we have concluded that For all topology under consideration, average delay, average jitter is, low for RYU controller. Average bitrate and throughput is more for RYU controller. Performance of the RYU controller does not vary with increasing number of hosts and switches in the network. The selection of appropriate controller will depend on the requirement and specification of the application.

6. REFERENCES

[1] Dr. Bhargavi Goswami, "Software Defined 6Journal of Innovative Research in Computer and Communication Engineering An ISO 3297: 2007 Certified Organization Vol.5, Special Issue 2, April 2017

[2] Chaymae EL KHALFI, Abderrahim EL QADI, Hamid BENNIS, "A Comparative Study of Software Defined Networks Controllers", ICCWCS'17, November 14–16, 2017, Larache, Morocco © 2017 Association for Computing Machinery. ACM ISBN 978-1-4503-5306-9/17/11

[3] Dobrin Dobrev ; Dimiter Avresky, "Comparison of SDN Controllers for Constructing Security Functions", Comparison of SDN Controllers for Constructing Security Functions", 2019 IEEE 18th International Symposium on Network Computing and Applications (NCA)

[4] Liehuang Zhu, Md Monjurul Karim, Kashif Sharif, Fan Li, Member, IEEE , Xiaojiang Du, Mohsen Guizani, "SDN Controllers: Benchmarking &

[5] Performance Evaluation", IEEE Journal on Selected Areas in Communications

[6] Omayma BELKADI and Yassin LAAZIZ, "A Systematic and Generic Method for Choosing A SDN Controller", International Journal of Computer Networks and Communications Security, VOL. 5, NO. 11, NOVEMBER 2017, 239–247

[7] V R SUDARSANA RAJU, "SDN CONTROLLERS COMPARISON", Proceedings of Science Globe International Conference, 10th June, 2018, Bengaluru, India

[8] Anna A. Semenovykh, Olga R. Laponina, "Comparative analysis of SDN controllers", International Journal of Open Information Technologies, Vol 6, No 7 (2018)

[9] Silvio E. Quincozes, Arthur A. Zopellaro Soares, Wilker Oliveira, Eduardo B. Cordeiro, Robson A. Lima, Debora Muchaluat-Saade, Vinicius C. Ferreira, Yona Lopes, ´ Juan Lucas Vieira, Luana M. Uchoa, Helio N. C. Neto, Leonardo F. Soares, ˆ Natalia C. Fernandes, Diego Passos, and Celio Albuquerque, "Survey and Comparison of SDN Controllers for Teleprotection and Control Power Systems", 978-3-903176-23-2 © 2019 IFIP

[10] Ola Salman ; Imad H. Elhajj ; Ayman Kayssi ; Ali Chehab, "SDN controllers: A comparative study", 2016 18th Mediterranean Electrotechnical Conference (MELECON)

[11] Lusani Mamushiane, Albert Lysko, Sabelo Dlamini, "A Comparative Evaluation of the Performance of Popular SDN Controllers", CSIR Pretoria, South Africa

[12] Muhammad H. Razaa , Shyamala C. Sivakumarb, Ali Nafarieha , Bill Robertsona," A Comparison of Software Defined Network (SDN) Implementation Strategies",2nd International Workshop on Survivable and Robust

Optical Networks Procedia Computer Science 32 (2014) 1050 – 1055

BIOGRAPHIES

I am N M Kazi working as a Assistant Professor in Electronics & Telecommunication department pursuing Ph.D in the field of Software Defined Network.