International Journal of Advances in
Soft Computing and Intelligent Systems (IJASCIS)
2024, Vol 03, Issue 02, 313-323
ISSN: 3048-4987

Original Paper

PERFORMANCE EVALUATION OF SDN CONTROLLERS: ANALYSING THE TCP TRAFFIC MANAGEMENT IN POX, RYU, AND ODL

Shanu Bhardwaj[a,*], Shailender Kumar[b]
Ashish Girdhar[c]

[a,b]Department of Computer Science and Engineering, Delhi Technological University, Delhi, India, shanubhardwaj1@gmail.com
[c]Department of Computer Science and Applications, Kurukshetra University, Kurukshetra, India,

ABSTRACT

Software-defined networking (SDN) is a transformative paradigm in the networking field that isolates the data plane and the control plane. The controller is one of the main entities in SDN that controls the flow of information in the network. Therefore, the research deals with a thorough performance differentiation of three prominent SDN controllers named POX, Ryu, and OpenDaylight (ODL). The study aims to evaluate the effectiveness of these controllers in controlling the traffic of the network, by focusing on performance parameters such as Transmission Control Protocol (TCP) mean, packet loss, as well as jitter. The experimental setup employed Mininet, a network emulator, to create a consistent virtual network environment for all controllers. Each controller was tested in isolated virtual machines, ensuring controlled and unbiased results.

The experimental results reveal distinct performance differences among the controllers. In the research experimentations, the highest TCP mean throughput, as well as superior performance among all controllers, is achieved by ODL consistently and exhibiting minimum loss of the data packets and jitter across all-time instances for high-demand, large-scale networks. This study demonstrates the crucial role of choosing the appropriate SDN controller based on specific network requirements, guiding network administrators and researchers in making informed decisions to ensure optimal network performance.

KEYWORDS

Software-defined networking, SDN controllers, Traffic analysis, TCP traffic management

1. INTRODUCTION

SDN is an extraordinary methodology in the field of the network that isolates the control plane from the physical plane, working by bringing together control and dynamic organization setup. The control and data planes are tightly coupled within individual devices, making traditional networks frequently rigid and complicated [1]. SDN defeats these restrictions by decoupling these planes, empowering incorporated network knowledge, working on administration, and upgrading adaptability. A global view of the network is made possible by this centralized architecture as shown in Fig. 1, which also speeds up the deployment of new services and applications, improves performance, and maximizes resource utilization [2].

Among the different SDN controllers accessible, Ryu, POX, and ODL are the absolute most broadly utilized regulators. Every one of these controllers offers novel elements and abilities, taking care of various use cases and necessities. Understanding the distinctions and genuine

uses of these controllers is fundamental for choosing the most proper answer for explicit systems administration needs [3]. Firstly, Ryu is an open-source SDN controller that is renowned for its simplicity, adaptability, and user-friendliness [4]. It supports various protocols, including OpenFlow, which is the standard for SDN communication between the control and data planes. Also, POX is another open-source SDN controller that has been instrumental in SDN examination and experimentation. It is intended to be lightweight and straightforward, making it a superb decision for learning and exploring different avenues regarding SDN ideas. Last but not least, the Linux Foundation supported the development of ODL, an open-source, scalable SDN controller. It plans to speed up the reception of SDN and NFV through a cooperative and straightforward improvement process. Additionally, traffic examination is vital in SDN controllers because of multiple factors, all of which add to the compelling administration and improvement of organization execution [5].

SDN controllers give a unified perspective on the whole network, empowering continuous observation and investigation of information traffic. Network administrators can dynamically optimize traffic flows, identify congestion points, and spot anomalies by utilizing this centralized control [6]. The capacity to program the organization through SDN controllers takes into account robotized traffic designing, further developing general organization proficiency and execution. Moreover, SDN works with the execution of cutting-edge safety efforts by empowering granular command over traffic streams and fast reaction to expected dangers. SDN controllers are a powerful tool for modern traffic analysis because of these capabilities, which improve end-user quality of service, reduce operational costs, and increase network reliability [7].

These controllers are put to use in a variety of situations to boost network agility, security, and efficiency. Because of its ease of use and capacity for rapid development, Ryu is frequently used in networks ranging from small to medium in size. POX is habitually utilized in instructive establishments and exploration labs to educate and investigate SDN ideas. Large-scale deployments in telecommunications and enterprise environments favor ODL due to its scalability and extensive feature set [8]. The significance of selecting the appropriate controller based on the requirements and goals of the network is emphasized by the fact that each controller has distinct advantages that make it suitable for particular applications.

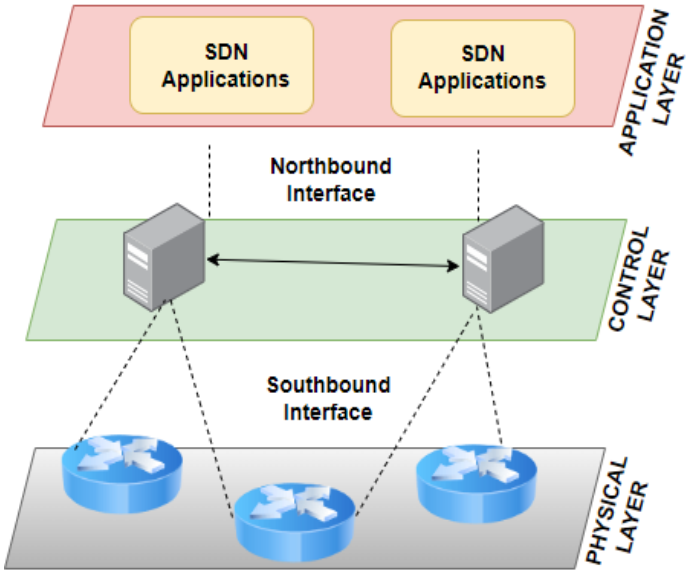


Figure 1. SDN Architecture

In this paper, the author presents a comparison of SDN controllers named Ryu, POX, and ODL. For this comparison, several performance parameters have been used, such as control response time, data flow efficiency, and network latency. Based on the results obtained from simulations, the author concludes which SDN controller is the best for traffic analysis. This study highlights the strengths and weaknesses of each controller based on different performance metrics, providing valuable insights into their effectiveness and suitability for network management and optimization.

2. BACKGROUND

This section highlights the distinct development histories and primary use cases of the SDN controllers.

2.1. Traditional Networking vs SDN

In conventional networks, traffic analysis is in many cases obliged by the decentralized idea of the network design, where control and information planes are implanted inside individual gadgets. Network managers should assemble information from various sources, which can be tedious and inclined to irregularities [9]. When analyzing traffic patterns or diagnosing issues, this decentralized approach frequently results in limited visibility and slower response times. SDNs, on the other hand, provide a centralized method for traffic analysis that gives a software-based controller access to the entire network. Anomalies can be detected, performance can be improved, and security policies can be enforced with ease thanks to this centralization, which makes it possible to monitor traffic flows in real time. SDNs' programmability also makes it easier to use cutting-edge analytics and machine learning algorithms, giving traffic management administrators greater insight and control [10]. As a result, SDNs make traffic analysis more accurate and efficient, allowing for quicker and more efficient responses to network conditions and potential threats.

Table 1. Properties of the SDN controllers.

Features	**RYU**	**POX**	**ODL**
Language Support	Python	Python	Java
Platform Support	Linux, Windows	Linux	Linux
Structure	Lightweight	Lightweight	Extensible
Virtualization	Mininet	Mininet	Mininet
Northbound API	REST etc.	RPC etc.	REST etc.
Southbound API	OpenFlow, BGP, etc.	OpenFlow	OpenFlow, NETCONF, etc.
Documentation	Good	Basic	Extensive
Scalability	Small scale	Small scale	Large scale
Modularity	Basic	Basic	High

2.2. Ryu controller

Ryu is a well-known open-source SDN controller that is known for being simple, adaptable, and simple to use. OpenFlow, the standard for SDN communication between the control and data planes, is one of the protocols that it supports as depicted in Fig. 2. Ryu gives a far-reaching set of libraries and instruments that work with the fast turn of events and sending of organization applications [11]. It is broadly utilized in scholarly examination and little to medium-sized creation conditions because of its direct engineering and broad documentation. Traffic engineering, network automation, and security monitoring are some of Ryu's real-time applications, making it an adaptable option for a variety of network scenarios.

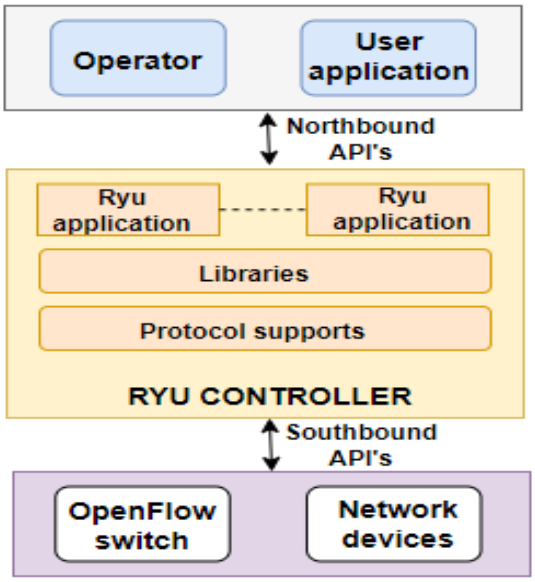


Figure 2. Architecture of Ryu SDN controller

2.3. POX controller

POX is another open-source SDN controller that has been instrumental in SDN examination and training [12]. It is intended to be lightweight and straightforward, making it a superb decision for learning and exploring different avenues regarding SDN ideas as shown in Fig. 3. POX's simplicity makes it a useful tool for developing and evaluating new SDN applications, even though it may lack the same level of sophistication and scalability as Ryu or ODL, also shown in table 1. POX is frequently utilized in experimental setups to validate novel networking concepts and in academic settings to teach SDN principles in real-world scenarios [13].

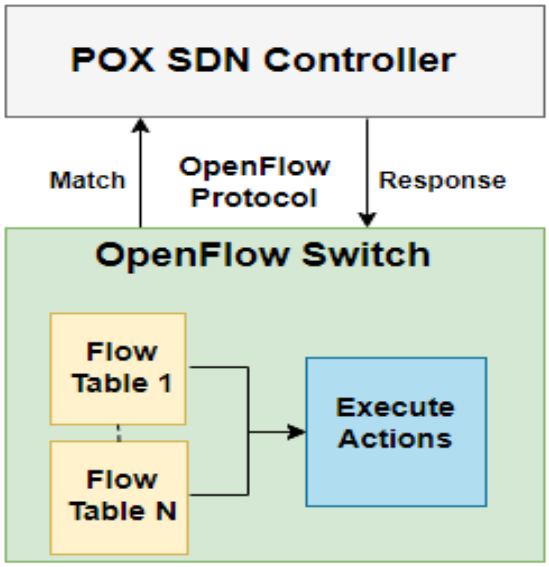


Figure 3. Architecture of POX SDN controller

2.4. ODL controller

ODL is a vigorous and versatile open-source SDN controller created under the Linux Establishment. Through a development process that is open and collaborative, it aims to

accelerate the use of SDN and NFV. OpenFlow, NETCONF, and BGP are just a few of the many southbound protocols that ODL can handle, making it ideal for large-scale production environments. Its modular design makes it easy to customize and integrate with a variety of network management tools [14]. ODL is broadly utilized in broadcast communications, server farms, and undertaking networks for errands like organization virtualization, administration coordination, and high-level network analytics [15]. Fig. 4 depicts the architecture of the ODL controller in the SDN environment.

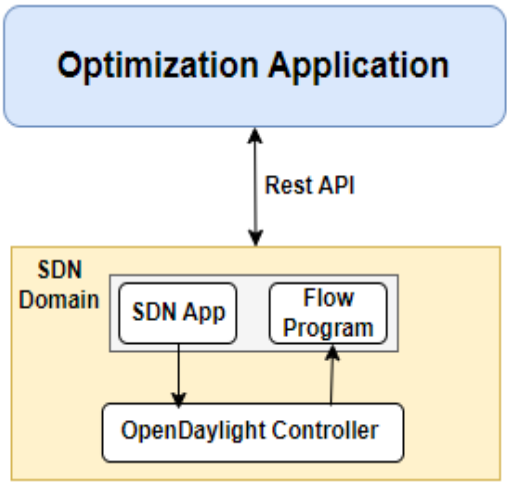


Figure 4. Architecture of ODL SDN controller

3. METHODOLOGY

To implement Ryu, POX, and ODL controllers for traffic analysis in an SDN environment, the first step involves ensuring that the servers or virtual machines being used have adequate resources and a compatible operating system, such as Ubuntu. Designing an appropriate network topology is crucial for effective traffic analysis, and this has been achieved using network emulation tools like Mininet to create virtual networks suitable for testing. The flow of the implementation of the proposed research work is depicted in Fig. 5 and Table 2.

The next step is the installation of the SDN controllers. For the Ryu controller, dependencies such as Python3 and related packages have been installed. Ryu itself can be installed using Python's package installer (pip). In a similar vein, cloning the POX repository and installing the POX controller requires Python 2.7 and pip. Beginning POX is direct with a basic order to start the controller. ODL, being more complex, includes downloading the ODL conveyance from the authority site, extracting the documents, and beginning the ODL utilizing the Karaf compartment. This setup guarantees that each of the three controllers is prepared for design and combination.

Each controller's traffic monitoring applications or components must be set up when the controllers are configured for traffic analysis. For Ryu, existing applications, for example, **simple_monitor_13.py** can be utilized, which has begun through the Ryu manager. On account of POX, traffic examination parts can be incorporated into the POX environment by beginning POX with these particular parts. ODL requires the establishment of elements like **old-l2switch-switch**, which work with traffic investigation. Sending traffic-checking applications inside ODL includes utilizing the Karaf climate to successfully deal with these highlights.

Coordinating the controllers with the network is the last step, where devices like Mininet assume a vital part. It is essential to install Mininet and construct a network topology that is capable of communicating with the SDN controllers. For this network to be managed and monitored, each controller must be configured. For instance, Mininet can begin with a predefined geography that interfaces with the SDN controllers, empowering them to accumulate traffic information and perform examinations. This integration takes into consideration ongoing traffic observing, anomaly detection, and execution improvement across the network, utilizing the abilities of Ryu, POX, and ODL.

Table 2. Steps of the implementation.

Steps to be followed	**Implementation**
Setting up the environment	• Hardware and software requirements • Network topology
Installing SDN controllers	• Ryu controller i. Install dependencies ii. Install Ryu • POX controller i. Install dependencies ii. Install POX • ODLcontroller i. Download ODL distribution ii. Extract the file tar –xvf distribution-karaf-x.x.x.tar.gz iii. Start ODL cd distribution-karaf-x.x.x./bin/karaf
Configuring the controllers for traffic analysis	• Ryu controller i. Develop or use existing Ryu applications for traffic analysis. ii. Start application ryu-managerpath/to/your/application.py • POX controller i. Create or use existing POX components for traffic analysis. ii. Start POX with the traffic analysis. • ODL controller i. Install necessary features for traffic analysis. ii. Develop or use existing ODL applications for traffic monitoring. iii. Deploy the application in the ODL environment using Karaf.
Integrating the controllers with the network	• Mininet setup i. Installation ii. Network Topology
Traffic analysis	• Data collection • Real-time monitoring

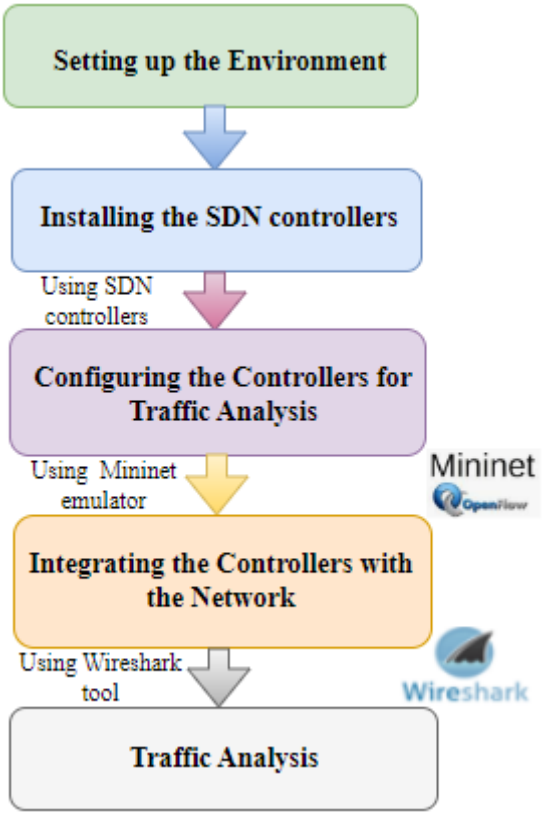


Figure 5. Implementation flow of the proposed SDN setup.

4. Results and Discussion

The results and discussion section of this research work delves into the comparative analysis of three prominent SDN controllers—Ryu, POX, and ODL, focusing on their implementation and efficacy in traffic analysis. The study begins with the successful installation and configuration of each controller within a Mininet virtual network environment, followed by an evaluation of their traffic monitoring capabilities. Key performance metrics such as latency, scalability, and ease of integration are examined to understand the practical implications of each controller.

4.1. TCP mean

In TCP mean [16], we used a methodical approach that included environment setup, traffic generation, and mean estimation to precisely examine how well the POX, Ryu, and ODL controllers handled TCP traffic within an SDN. This method ensures that we have methodically evaluated and differentiated each controller's traffic analysis capabilities.

The TCP was used to measure POX, Ryu, and ODL at time instances of 30 seconds, 20 seconds, and 10 seconds, respectively, for evaluation as shown in Fig. 6. The TCP mean is compared to various time instances in the table. For each time interval, the performance of the controllers is recorded. At the 30-second mark, ODL shows the highest throughput, followed by Ryu. At 20 seconds, ODL's throughput decreases, while Ryu and POX register 26.6 and 25.05, respectively. At 10 seconds, ODL maintains a high throughput of 32.49, Ryu peaks at 29.4, and POX records 23.6. These results suggest that ODL consistently outperforms the other controllers in terms of mean TCP throughput across all time intervals, while Ryu and POX show variable performance with Ryu generally performing better than POX.

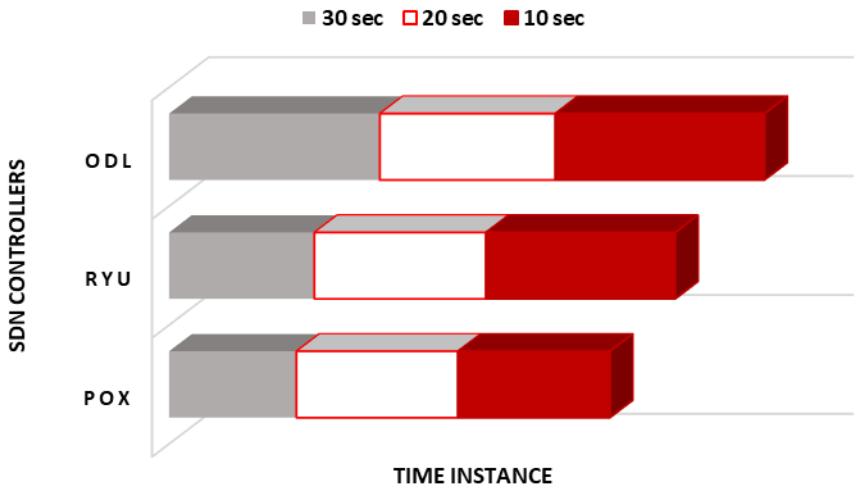


Figure 6. TCP means of POX, Ryu, and ODL controller.

4.2. Jitter and Packet loss

The bar graph illustrates packet loss percentages and jitter values (in milliseconds) for three SDN controllers POX, Ryu, and ODL, at three different time intervals: 10 seconds, 20 seconds, and 30 seconds as shown in Fig. 7.

At 10 seconds, POX exhibits the highest packet loss, followed by Ryu with a noticeable amount, and ODL with minimal packet loss. POX shows the highest jitter, while Ryu and ODL have significantly lower jitter values. At 20 seconds, Packet Loss: POX continues to have the highest packet loss, with Ryu showing moderate packet loss, and ODL maintaining a low packet loss rate. POX has the highest jitter once again, while Ryu and ODL show very low jitter. At 30 seconds, POX remains with the highest packet loss, Ryu has moderate packet loss, and ODL displays the least packet loss. POX still has the highest jitter, with Ryu showing less, and ODL having minimal jitter.

Fig. 8 provides an overview of network performance metrics specifically jitter and packet loss across different network controllers named POX, Ryu, and ODL at various bandwidths such as 500 Mbps, 600 Mbps, 700 Mbps, and 800 Mbps. Jitter measures the variability in packet arrival times, while packet loss represents the percentage of lost data packets. Among the controllers, Ryu demonstrates the best performance overall, with the lowest jitter and packet loss across most bandwidths, particularly excelling at 700 Mbps with minimal packet loss. ODL also shows strong performance but with slightly higher jitter compared to Ryu, especially at higher bandwidths. POX, while consistent in jitter, exhibits higher packet loss relative to Ryu and ODL. This indicates that Ryu is the most efficient in terms of both consistency and reliability of data transmission, while ODL performs well but with a bit more variability, and POX shows some limitations in maintaining low packet loss.

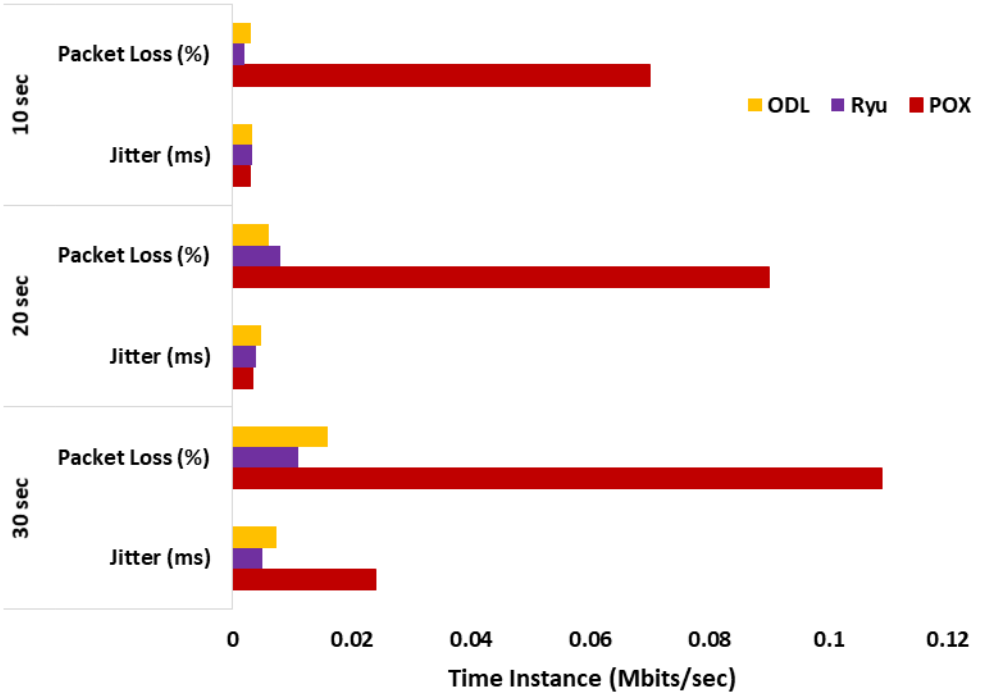


Figure 7. Jitter and packet loss of the SDN controllers in distinct time instances.

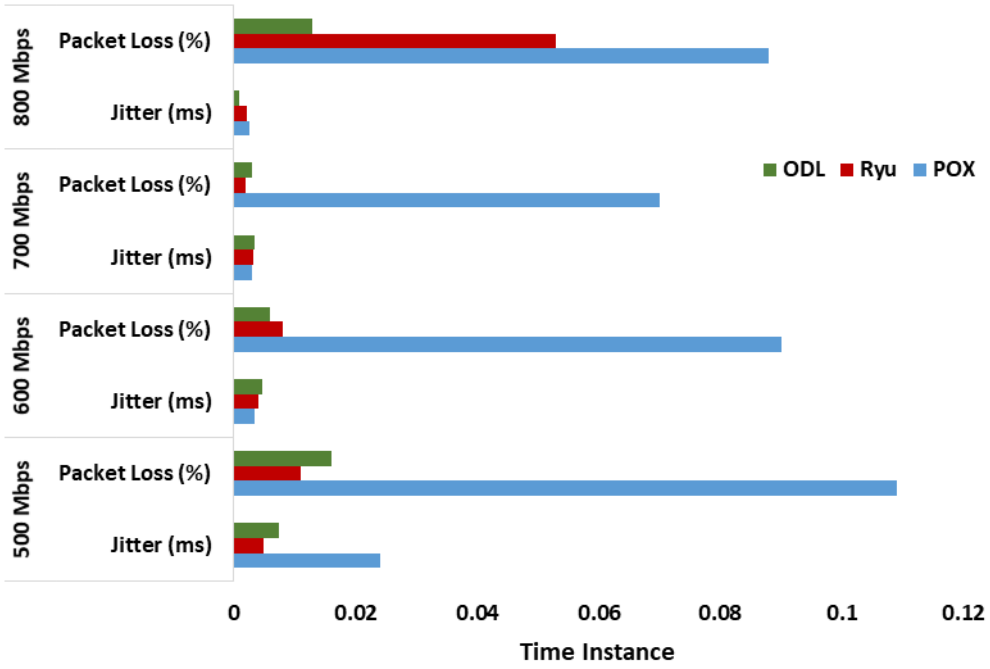


Figure 8. Jitter and packet loss of the SDN controllers in distinct bandwidth.

5. CONCLUSION

The comparative analysis of the SDN controllers POX, Ryu, and ODL, revealed significant performance differences in terms of TCP throughput, packet loss, and jitter. ODL consistently outperformed the others, demonstrating the highest mean TCP throughput, minimal packet loss, and the lowest jitter, making it ideal for large-scale, high-performance networks. Ryu showed moderate performance, surpassing POX but not matching ODL, indicating its suitability for medium-sized networks that require a balance between ease of use and performance. POX, with the lowest throughput and highest packet loss and jitter, is best suited for smaller networks, educational purposes, and experimental setups where simplicity is prioritized. This research underscores the critical role of choosing the appropriate SDN controller to ensure optimal network performance and reliability, tailored to specific network requirements and operational contexts.

Disclosure of Interests. The author has no competing interests to declare that are relevant to the content of this article.

REFERENCES.

[1] Chahal, J. K., Bhandari, A., & Behal, S. (2024). DDoS attacks & defense mechanisms in SDN-enabled cloud: Taxonomy, review and research challenges. Computer Science Review, 53, 100644.

[2] Abdi, A. H., Audah, L., Salh, A., Alhartomi, M. A., Rasheed, H., Ahmed, S., & Tahir, A. (2024). Security Control and Data Planes of SDN: A Comprehensive Review of Traditional, AI and MTD Approaches to Security Solutions. IEEE Access.

[3] Bhardwaj, S., & Girdhar, A. (2023). Network traffic analysis in software-defined networking using ryu controller. Wireless Personal Communications, 132(3), 1797-1818.

[4] Bhardwaj, S., & Girdhar, A. (2021, November). Software-defined networking: A traffic engineering approach. In 2021 IEEE 8th Uttar Pradesh Section International Conference on Electrical, Electronics and Computer Engineering (UPCON) (pp. 1-5). IEEE.

[5] Adanza, D., Gifre, L., Alemany, P., Fernández-Palacios, J. P., González-de-Dios, O., Muñoz, R., & Vilalta, R. (2024). Enabling traffic forecasting with cloud-native SDN controller in transport networks. Computer Networks, 250, 110565.

[6] Shirmarz, A., & Ghaffari, A. (2020). Performance issues and solutions in SDN-based data center: a survey. The Journal of Supercomputing, 76(10), 7545-7593.

[7] Bhardwaj, S., Panda, S. N., & Datta, P. (2020, December). Layer-Based Attacks in the Ternary Planes of Software-Defined Networking. In 2020 IEEE International Women in Engineering (WIE) Conference on Electrical and Computer Engineering (WIECON-ECE) (pp. 292-295). IEEE.

[8] Maleh, Y., Qasmaoui, Y., El Gholami, K., Sadqi, Y., & Mounir, S. (2023). A comprehensive survey on SDN security: threats, mitigations, and future directions. Journal of Reliable Intelligent Environments, 9(2), 201-239.

[9] Indrason, N., & Saha, G. (2024). Exploring Blockchain-driven security in SDN-based IoT networks. Journal of Network and Computer Applications, 103838.

[10] Jiang, W., Han, H., He, M., & Gu, W. (2024). ML-based pre-deployment SDN performance prediction with neural network boosting regression. Expert Systems with Applications, 241, 122774.

[11] Bhardwaj, S., & Panda, S. N. (2022). Performance evaluation using RYU SDN controller in software-defined networking environment. Wireless Personal Communications, 122(1), 701-723.

[12] Cabarkapa, D., & Rancic, D. (2021). Performance Analysis of Ryu-POX Controller in Different Tree-Based SDN Topologies. Advances in Electrical & Computer Engineering, 21(3).

[13] Rizaldi, M. I., Yusuf, E. A. S., Akbi, D. R., & Suharso, W. (2024). A Comparison of Ryu and Pox Controllers: A Parallel Implementation. Jurnal Online Informatika, 9(1), 1-9.

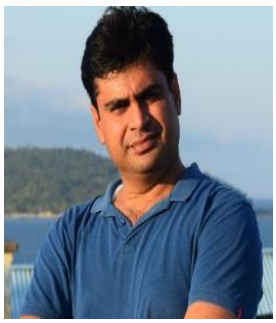
[14] Hassen, H., & Meherzi, S. (2024). Performance evaluation of centralised and distributed controllers in software defined networks. International Journal of Wireless and Mobile Computing, 27(2), 103-117.

[15] Singh, A., Kaur, N., & Kaur, H. (2022). Extensive performance analysis of OpenDayLight (ODL) and open network operating system (ONOS) SDN controllers. Microprocessors and Microsystems, 95, 104715.

[16] Adhikari, T., Kumar Khan, A., & Kule, M. (2024). ProDetect: A Proactive Detection Approach of the TCP SYN Flooding Attack in the SDN Controller. IETE Journal of Education, 1-12.

Authors

Shanu Bhardwaj is currently pursuing PhD in Department of Computer Science & Engineering at Delhi Technological University, Delhi, India. She has completed her B. Tech in Computer Science and Engineering at Kurukshetra University, Kurukshetra (India) and M.E. at Chitkara University, Punjab (India). Her research areas are Software-Defined Networking (SDN), Internet of Things (IoT), Wireless sensor network (WSN) and Network security. She has completed her internship on "Application of Big Data in Construction Law" from Tamkang University, Taiwan.



Prof. Shailender Kumar is currently working as Professor in Department of Computer Science & Engineering at Delhi Technological University since December 2018. He has total of more than 17 years of Teaching Experience. He has 20 SCI/SCIE and other publications. His major area of research are Network Security, Computer Networks, Database Management Systems and Machine Learning.

Dr. Ashish Girdhar is currently working as Assistant Professor in Department of Computer Science & Applications at Kurukshetra University since 2022. He has total of more than 12 years of Teaching Experience. He has done his PhD in area of Image Processing from Thapar Institute of Engineering and Technology, Patiala. He has 7 SCI/SCIE Publications and few other publications. His major area of research are Image Processing and Machine Learning.