Performance Evaluation of SDN Controllers: RYU and POX for WBAN-based Healthcare Applications

Lama Alfaify†, Nujud Alnajem† , Haya Alanzi†, Rawan Almutiri†, Areej Alotaibi† , Nourah Alhazri† , Awatif Alqahtani†

Lamaal-faify@hotmail.com nujud.alnajem@outlook.com Hayaalanzi802@gmail.com almutiri.rawan08@gmail.com Areej1213@gmail.com Nourahalhazri@gmail.com aqahtani1@ksu.edu.sa

† Computer Science and Engineering Dept. College of Applied Studies and Community Service, King Saud University

Abstract

Wireless Body Area Networks (WBANs) have made it easier for healthcare workers and patients to monitor patients' status continuously in real time. WBANs have complex and diverse network structures; thus, management and control can be challenging. Therefore, considering emerging Software-defined networks (SDN) with WBANs is a promising technology since SDN implements a new network management and design approach. The SDN concept is used in this study to create more adaptable and dynamic network architectures for WBANs. The study focuses on comparing the performance of two SDN controllers, POX and Ryu, using Mininet, an open-source simulation tool, to construct network topologies. The performance of the controllers is evaluated based on bandwidth, throughput, and round-trip time metrics for networks using an OpenFlow switch with sixteen nodes and a controller for each topology. The study finds that the choice of network controller can significantly impact network performance and suggests that monitoring network performance indicators is crucial for optimizing network performance. The project provides valuable insights into the performance of SDN-based WBANs using POX and Ryu controllers and highlights the importance of selecting the appropriate network controller for a given network architecture.

Keywords:

SDN, WBAN, Health Care, Ryu Controller, POX Controller.

1. Introduction

Recent advancements in wireless communications, mobile computing, and sensor technologies have allowed for the development of low-cost, small, lightweight, intelligent wireless sensor devices. These tiny devices can be strategically placed in critical areas of the human body and linked through a wireless network to establish a Wireless Body Area Network (WBAN). With the rapid development of wireless communication and semiconductor technologies, the field of sensor networks has expanded substantially to support a variety of applications, such as medical and healthcare systems. A Wireless Body Area Network (WBAN) is a special-purpose sensor network designed to operate autonomously in order to connect various medical sensors and appliances located inside and outside the human body [1]. WBAN is a wireless networking system that uses Radio Frequency (RF) to connect several tiny nodes with sensor or actuator capabilities [2]. The implementation of a WBAN for medical monitoring and other applications will provide both health care professionals and patients with cost-saving flexibility options [1]. WBAN has recently received a lot of interest from academia and industry [2].

Software-Defined Networking (SDN) is a potential approach for enhancing WBAN performance. SDN controllers are software programs that give a centralized view of the network's resources, allowing healthcare professionals to manage the network's behavior and resources depending on each patient's individual needs [22]. It is a technique that employs specialist software such as OPEN-FLOW, CDDA, and others, in which networks are set up and maintained programmatically using a small number of physical components to satisfy any organization's network requirements. SDN offers many advantages, such as on-demand provisioning, automated load balancing, and simplified physical infrastructure. Those who are struggling to get above the constraints of traditional networking are turning to SDN, which offers new perspectives on how networks are handled. SDN also enables the hardware to be controlled or managed from a centralized software application that is separated from the hardware itself by decoupling hardware from software, i.e., separating the control plane (which decides where to send traffic) from the data plane (which executes these decisions and forwards traffic) [27] (see Fig. 1).

The heterogeneous and complicated network structure of WBANs has several problems in terms of control and administration, such as heterogeneity, scalability, and energy efficiency, and WBAN devices have limited capabilities [4]. Low battery life, due to its tiny size,

Manuscript received July 5, 2023
Manuscript revised July 20, 2023
https://doi.org/10.22937/IJCSNS.2023.23.7.25

delays that increase the possibility of a misdiagnosis, and security breaches by malicious persons who can exploit the obtained data for unlawful purposes Since a WBAN must offer a communication path for multiple devices to interact with one another, scalability is crucial [5].

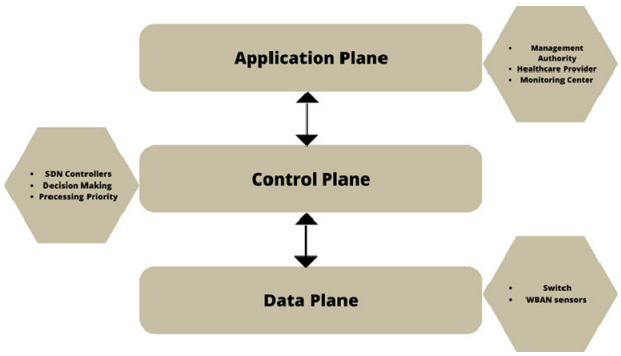

Fig. 1 SDN Planes

In healthcare systems, there are a wide range of devices and systems, each of which has its own configuration and management requirements. Therefore, when considering healthcare systems, it is necessary to shed light on managing network traffic. However, traditional network architectures rely on complex and rigid hardware-based systems, which make network management in healthcare systems more challenging [4].

Furthermore, each network device must be handled independently, which is not ideal. In WBAN, devices are mostly in a mobile state since they are attached to the human body, which is naturally expected to move around. This might imply the need to set new rules or enforce devices to send packets using certain paths. Furthermore, one of the most significant needs in WBANs is the privacy and security of patient medical data, which is a very sensitive kind of personal data, and both the doctor's and the patient's permission and authorization must be sought to use or see it so that the information is not misused [4].

In this paper, we will analyze the performance of two SDN controllers: POX and Ryu. POX and RYU are widely used among other controllers and are used to accelerate the development of new network applications [26]. In the literature, there are a considerable number of works that have evaluated the performance of one or more SDN controllers; however, as far as we know, there are not many works that have evaluated the performance of SDN controllers with three different types of network topologies and monitored more than one QoS metric. Furthermore, this work is carried out to determine the importance of adding a layer to WBANs-based SDN architecture that is responsible for mapping the most appropriate SDN controllers to enhance the performance of the applications based on their predefined QoS. For further illustration, before reaching a conclusion about which type of SDN controller to link OpenFlow-enabled switches to, we need to evaluate different SDN controllers. There is a possibility that one controller is more suitable for time-sensitive applications than another SDN controller, which might work better with an application that requires high throughput. However, as far as we know and based on our research, there is not much work related to WBAN that evaluates the performance of SDN controllers as a prior step to enhancing the performance of SDN-based WBAN applications by comparing the performance of different controllers. Therefore, we carry out this research to promote the importance of adding a tier within the network management layer to allow for selecting which SDN controller is more appropriate based on the application requirements. The paper consists of the following sections: In Section 2, we present some background knowledge related to the research and a number of related works. Methodology, results, and discussion are represented in sections 3, 4, and 5, respectively.

2. Background

In this section, we introduce the technologies and concepts related to the scope of the presented work, which include SDN controllers in healthcare, POX controllers, RYU controllers, and Mininet. We will provide a brief overview of these technologies and concepts.

2.1 SDN controllers in healthcare

SDN controllers are used in healthcare to monitor and control healthcare network infrastructure. SDN controllers are very beneficial in healthcare because they provide increased flexibility and control over network infrastructure. A healthcare provider, for example, might use an SDN controller to provide extra bandwidth to a high-traffic region of the network or to prioritize some types of network traffic over others, such as important patient data over non-critical data [22]. Additionally, SDN controllers in healthcare can improve network security by allowing for the centralized administration of security rules and access restrictions. This can aid in preventing illegal access to sensitive patient data as well as ensuring compliance with data privacy standards. SDN controllers, in general, give healthcare companies more control and flexibility over their network infrastructure, allowing them to better address the individual demands of their patients and enhance the quality of care they deliver [22].
Figure 2 illustrates a four-layered SDN-based WBAN architecture designed to collect and analyze patient data to enhance healthcare. The health-care Monitoring network layer is in charge of gathering data from the WBAN sensors attached to the patient's body. This data is sent to

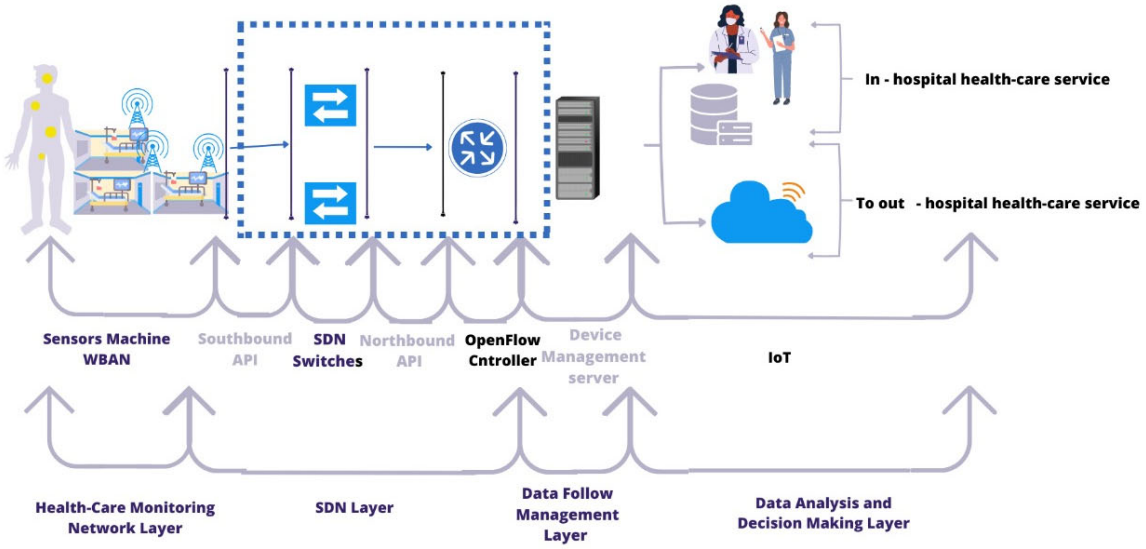

Fig. 2 SDN-Based WBAN Architecture [20]

the SDN-enabled switches in the SDN layer via the data plane, which is made up of HUBs that control the network's sensor nodes. The SDN layer is in charge of managing and controlling the network through the use of SDN technology. This layer is made up of various components, including the SDN controller, which runs on the control plane and is in charge of managing the network as well as dealing with changes in topology, traffic loads, and messages from other controllers. SDN switches are programmable devices that can be managed by the OpenFlow controller. The northbound API communicates with the control plane, allowing applications to communicate with the SDN controller, and the southbound API communicates with the data plane, forwarding requests to the SDN switches. The Data Follow Management Layer is responsible for device management, managing the configuration and maintenance of the WBAN sensors and SDN-enabled switches in the network. The Data Analysis & Decision-Making Layer is responsible for analysing the data collected from the WBAN sensors and making informed decisions based on that data. IoT devices are used in this layer to process the data and communicate with the SDN controller. This layer corresponds to the Application plane in SDN architecture.

In SDN, the control plane is separated from the data plane, and the controller is a software-based controller while the network devices are simple packet-forwarding devices programmable via an open interface in the data plane.
Overall, SDN controllers are an effective tool for healthcare businesses seeking to increase network efficiency, security, and performance. SDN controllers help healthcare providers better serve their patients and provide superior-quality treatment by giving them centralized control over network infrastructure [18].

2.2 POX controllers

POX (Pythonic Network Operating System) is a Python-based open-source OpenFlow/Software Defined Networking (SDN) controller [4]. New network applications may be developed and prototyped more quickly with POX. It is favored by academics, developers, and network operators since it is made to be readily adaptable and expandable. Pox Controller offers a straightforward and adaptable architecture that enables users to manage packet forwarding, resulting in a more effective and personalized network [4].
The following is a summary of some of the key characteristics of a POX controller:

- It can offer a Pythonic OpenFlow interface.
- It has reusable example parts for topology finding, path selection, etc.
- It has a Mininet simulator preloaded and can operate in any environment with an operating system.
- It is capable of supporting a virtual architecture and Graphical User Interface (GUI) that are identical to NOX's.
- It may perform more effectively than NOX implemented in Python.

2.3 RYU controllers

NTT Labs developed Ryu, one of the most famous controllers. It is a software-defined networking framework with APIs primarily used to accelerate the development and prototyping of novel network applications. The Ryu Controller is developed in Python and runs on the Linux operating system. It supports the NETCONF and OF-config network management protocols in addition to OpenFlow.

The Open Virtual Switch DataBase (OVSDB) library, which is used to set up switches to work with the OF protocol and lets you add, delete, or change rules in a flow table, and the NETConfig library, which helps you set up network devices, are both RYU controller components that make it easier to make network applications and manage networks [23].

2.4 Mininet

Mininet is a network simulation system that runs SDN switches, different end-hosts, and links between all the devices on a Linux kernel system. We use it to simulate SDN technology, and as a real system, we can operate the network. It uses lightweight virtualization to make a single system look like a complete network. Mininet Python APIs (Application Programming Interfaces) can help create and retain scenarios. By comparing it with any other network simulator, we find it easy and flexible to use, as it is best suited to an SDN environment. Mininet can perform reasonably well on a single laptop by leveraging Linux features, so we don't need real devices to do Simulation. We can also control the form of topology and the number and type of controllers and devices [24].

3. Related Work

Several previous studies have assessed the performance of several SDN controllers, including POX and Ryu. Abdullah et al. [19] assessed and compared several SDN controllers, such as NOX, POX, and Floodlight. The authors evaluated the performance of the controllers using several measures, including throughput, packet loss, and latency. The POX controller outperformed the others in terms of latency and throughput, while Floodlight had the lowest packet loss rate. Similarly, Askar and Keti [18] examined several SDN controllers, such as POX, Floodlight, Ryu, and OpenDaylight. The performance of the controllers was tested using numerous criteria, including throughput, latency, and scalability. According to the results, Ryu had the maximum throughput while Floodlight had the lowest delay. The performance of the controllers was tested using numerous criteria, including throughput, latency, and scalability. According to the results, Ryu had the maximum throughput while Floodlight had the lowest delay. The authors concluded that the selection of an SDN controller should be based on the unique application needs. Bholebawa and Dalal [17] used several measures such as latency, throughput, and jitter to compare the performance of POX and Floodlight controllers. The authors built the topology with the Mininet simulation program and tested the controllers with various traffic patterns. POX performed better in terms of latency and jitter, whereas Floodlight performed better in terms of throughput. References [6],[7],[8] propose cross-layer routing, also known as "secure cross-layer," which is used in a heterogeneous WBAN network across several layers. The main research of [9], [10], [11], and [12] is on applying clustering techniques for WBANs to maintain network connectivity, balance network center and edge energy consumption, adapt to changing topological structures, and improve network resilience as the number of nodes and distance between them increase. Other works, such as [13], [14], [15], and [16], focus on QoS-based routing to enhance the performance of WBAN applications by considering different QoS metrics such as latency, bandwidth, delay-jitter, communication delay, and buffer space. However, these studies aim to enhance the performance and/or security of WBAN applications, but there is no mention of utilizing the SDN approach. On the other hand, work in [17], [18], and [19] evaluates the performance of SDN controllers. For example, in [17], the efficient network simulator Mininet is used to compare the efficiency of both POX and floodlight controllers across various network topologies by analyzing network throughput and round-trip delay. Reference [18] evaluated the performance of POX and RYU controllers using Mininet, but they did it only for a single topology (i.e., a topology with only a single OpenFlow-enabled switch). Reference [19] evaluated the performance of SDN controllers using Dijkstra’s algorithm.

In conclusion, each SDN controller has advantages and disadvantages that should be considered when selecting an SDN controller for a specific application (e.g., healthcare applications). However, the proposed work expands on previous research by concentrating especially on the performance of POX and Ryu controllers in healthcare applications. This project will assure the comparability of the results and expand on the findings of previous research by employing comparable assessment measures and simulation tools.

4. Methodology

In this section, we provide an overview of the design and implementation of the simulations to evaluate the suitability of the POX and Ryu controllers for real-world networking scenarios. We first will define a topology to unify the testing environment for the sake of performance comparison among the controllers. Then, we will use the open-source simulation tool "Mininet" to build the topology; multiple topologies will be proposed, where we can use the POX or Ryu controller as part of that topology. Then, the performance of the aforementioned controllers will be analyzed with the aid of Mininet's command-line interface (CLI). Then, we will investigate the performance of the two controllers in terms of QoS metrics, which include throughput, bandwidth, and round-trip time between end-user nodes. Then, we will give a recommendation based on the result to reflect whether we see, based on the result, the importance of selecting SDN controllers dynamically based on the interested health care applications.

The simulation environment for the study consists of a virtual machine running Ubuntu 18.04 LTS with Mininet, MiniEdit, Python, and Wireshark installed. Two CPUs and 30GB of RAM were also made available to the virtual machine. These hardware resources are useful for helping run simulations and gather network data. Mininet is used to create network topologies, and Python scripts are used to automate the simulation process and analyze network data. Wireshark is used to capture, examine, and analyze network traffic. We configured the virtual machine with adequate hardware resources and network specifications for accurate simulation results.

Mininet Simulator utilizes lightweight virtualization technologies to construct numerous virtual hosts, switches, and connections on a single physical system, enabling you to establish a virtual network environment for testing and development needs. You may test network applications, simulate real-world network topologies, and assess network protocols and algorithms using Mininet. Additionally, Mininet has a straightforward command-line interface (CLI) that enables the creation, configuration, and management of virtual network components, including hosts, switches, and connections. Furthermore, you can automate network configuration and testing tasks with Mininet by using Python scripts.

Mininet has a straightforward command-line interface (CLI) that enables the creation, configuration, and management of virtual network components, including hosts, switches, and connections. Furthermore, you can automate network configuration and testing tasks with Mininet by using Python scripts. In addition, Mininet supports a number of network topologies, including custom, linear, tree, and mesh topologies. Although pre-defined network topologies are available, in this work, topologies are customized by specifying the number of hosts, switches, and links.

In this paper, three different network topologies (linear, single, and tree) are designed to evaluate the controllers' performance, and performance metrics such as bandwidth, throughput, and round-trip time are measured. The project evaluated the performance of POX and Ryu controllers in OpenFlow-enabled network topologies, and the controllers were configured to interact with network devices and manage network traffic. Furthermore, the results for each metric were compared for both controllers, and an OpenFlow switch, and a controller were used for each topology.

4.1 Single Topology

The single topology is a basic network architecture that consists of a succession of switches linked directly to the switch [21]. In this work, the single topology was used as one of the three topologies to evaluate the performance of the POX and Ryu controllers. The single topology was chosen because it provides a simple and straightforward network structure, which makes it easy to analyze the performance of the controllers. Additionally, the single topology is a commonly used topology in networking, which makes it a relevant choice for evaluating the controllers in a practical setting. The single topology consists of 16 hosts and a single OpenFlow-capable switch (see Fig. 3). The controllers (POX and Ryu) were remotely enabled in the Mininet console using their respective commands and were linked to the switch through the IP address 127.0.0.1. The controllers' performance in this architecture was assessed based on their ability to regulate bandwidth, throughput, and round-trip time. Comprehensively, the single topology provided a basic and feasible network structure for assessing the controllers' performance, allowing for an easy comparison of the two controllers' capacity to manage network traffic in a single network.

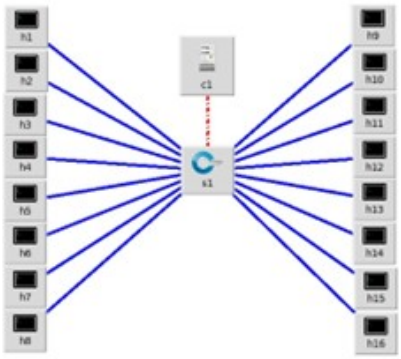

Fig. 3 Single Topology

4.2 Linear Topology

The linear topology in Miniedit is a sort of network architecture in which hosts are connected to switches in a linear way. Each switch is linked to the next until the last switch is reached, which is linked to a controller [19]. The linear topology is utilized to test the performance of the POX and Ryu controllers in handling network traffic in a linear network. Miniedit is used in the project to design a linear topology with 16 hosts. The hosts were linked to four switches, each of which is linked to another switch until the last switch is reached, which is linked to a POX or Ryu controller (see Fig. 4). The controller is remotely activated in the Mininet console using its relevant command and is linked to the OpenFlow-enabled switch via the IP address 127.0.0.1. The ability of the POX and Ryu controllers to manage bandwidth, throughput, and round-trip time in this linear architecture is used to assess their performance. Bandwidth refers to the amount of data that can be transmitted over the network in a given period, while throughput refers to the amount of data that is actually transmitted over the network in a given period. Round-trip time refers to the time it takes for a packet to travel from the source host to the destination host and back again. Linear topology was chosen because it provides a basic and uncomplicated network layout that allows for easy analysis of controller performance. Furthermore, linear topology is a widely used topology in networking, making it an appropriate choice for assessing controllers in a realistic scenario. Miniedit's linear architecture also enables the building of networks with a limited number of hosts and switches, which simplifies testing and decreases the processing resources required. Overall, the linear topology was a simple and realistic way to test the performance of the POX and Ryu controllers. It allowed direct comparison of the two controllers' ability to handle network traffic in a linear network.

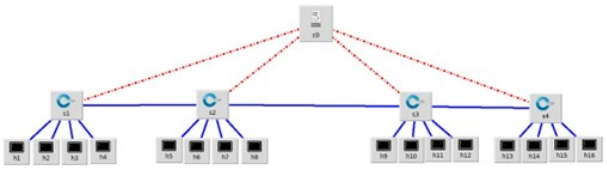

Fig. 4 Linear Topology

4.3 Tree Topology

The tree topology is a network architecture in which hosts and switches are linked in a hierarchical form, with a root switch at the top and branches of switches extending downward to connect to hosts [25]. The tree topology was utilized in the project to assess the efficacy of the POX and Ryu controllers in handling network traffic in a hierarchical network. Five OpenFlow-enabled switches were linked hierarchically in Miniedit to form a tree topology with 16 hosts (see Fig. 5). The controllers (POX and Ryu) were linked to the five underlying switches. The controllers' performance was assessed based on their ability to manage bandwidth, throughput, and round-trip time in this tree architecture. Furthermore, the tree topology was chosen because it provides a hierarchical network structure, which is useful in a variety of actual networking settings, such as a campus or workplace network. The tree architecture also enables the design of a network with a limited number of switches and hosts, which simplifies testing and minimizes the computing resources required. The controllers' performance was assessed using a variety of parameters, including bandwidth, throughput, and round-trip time. Overall, the tree topology provided a hierarchical network structure for assessing the performance of the POX and Ryu controllers, allowing for a direct comparison of the two controllers' capacity to handle network traffic in a hierarchical network. The project's performance measurements give important insights into the controllers' performance and appropriateness for usage in real-world applications.

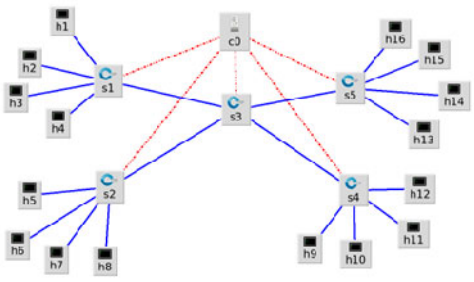

Fig. 5 Tree Topology

The Three above-described network topologies (linear, single, and tree) were designed to evaluate the controllers' performance and measure the following performance metrics: bandwidth, throughput, and round-trip time.

5. Results and Discussion

In this section, we present the captured metrics (throughput, bandwidth, and round-trip time) for each of the above-described topologies: single, linear, and tree.

- The presented work evaluates the performance of SDN-based networks where we use POX or RYU controllers to conduct an analysis of their performance in OpenFlow-enabled network topologies. The configuration of the controllers is responsible for defining how they interact with the network devices, including switches and hosts, and how they manage network traffic. To set up the network topology, we first use a command in Mininet to establish it, then export the Miniedit diagram to a Python file. Once the topology is created, we need to start the Pox or Ryu controllers for each one. The remote controllers, POX or Ryu, connect to the network switches through the IP address 127.0.0.1.

- The next step is to initiate the POX Controller, which listens on port 6653 and uses the OpenFlow protocol, as depicted in Figure 5.2. By enabling the verbose option, we can view details about the current operation of this controller on the terminal.

- After initiating the RYU controller, the controller listens on a port number specified in the Python file, with the default port being 6653. An OFPHandler is also utilized to manage all network OpenFlow traffic.

In the following, we present the results of the performance metrics evaluation for Software-defined Wireless Body Area Networks (SD-WBAN) using two SDN controllers, POX and Ryu. The performance metrics examined include bandwidth, throughput, and round-trip time (RTT).

5.1 Bandwidth Results

Table 1 represents the performance of the bandwidth rate for the different network topologies tested on Pox and Ryu controllers. The numbers in the table represent the measured bandwidth (in Gbits/sec) for each topology in each controller. In both controllers, the tree topology has the highest bandwidth, followed by the linear topology, and then the single topology. This is likely due to the fact that the tree topology provides more paths for data to travel through, which can increase overall network performance.

Table 1: Results of the Bandwidth Performance Values of the SDN Controller.

Topologies	Pox	Ryu
Tree	[10.4 Gbits/sec, 10.4 Gbits/sec]	[38.3 Gbits/sec, 38.5 Gbits/sec]
Linear	[23.6 Gbits/sec, 23.6 Gbits/sec]	[36.2 Gbits/sec, 36.2 Gbits/sec]
Single	[27.4 Gbits/sec, 27.4 Gbits/sec]	[30.0 Gbits/sec, 30.0 Gbits/sec]

5.2 Throughput Results

5.2.1 Single Topology

Figure 6 below shows the TCP throughput for a single topology with three different node-to-node routes that use both POX and RYU controllers. For each of the three pathways, the measurements are in GBytes/sec and are broken down into time intervals (0–1 sec, 1-2 sec, and so on). According to the figure, POX controllers often provide greater average TCP throughput than RYU.

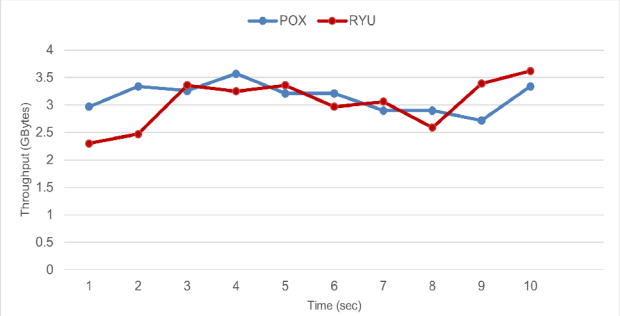

Fig. 6 H1 To H16 Throughput Comparison Between POX And RYU Controllers for Single Topology

5.2.2 Linear Topology

The results of TCP throughput tests utilizing both POX and RYU controllers for a linear architecture with three different node-to-node routes that use both POX and RYU controllers. For each of the three pathways, the measurements are in GBytes/sec and are broken down into time intervals (0–1 sec, 1-2 sec, and so on).

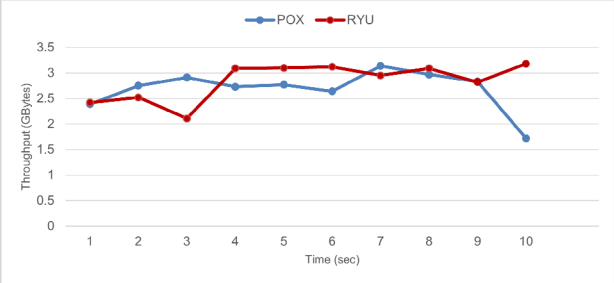

Fig. 7 H1 To H16 Throughput Comparison Between POX And RYU Controllers for Linear Topology

5.2.3 Tree Topology

The TCP performance measurements for a tree architecture with three alternative node-to-node links utilizing both POX and RYU controllers are presented in Fig. 7. For each of the three pathways, the measurements are made in GBytes/sec and are broken down into time intervals (0–1 sec, 1-2 sec, and so on). The result includes the average TCP throughput for each path in both controllers.

According to the results, the POX controller often provides greater average TCP throughput than the RYU controller for each path. Additionally, for both controllers, path h1 to h5 has the best average TCP performance, then path h1 to h12, and finally path h1 to h16.

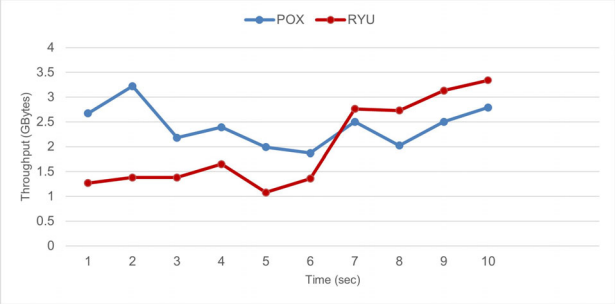

Fig. 8 H1 To H16 Throughput Comparison Between POX And RYU Controllers for Tree Topology

Based on the TCP throughput figures in the prior tables, it appears that the POX controller outperforms the RYU controller in general. This implies that POX can process and transport more data than RYU, making it more suitable for data-intensive networks that demand high throughput and transfer rates. RYU, on the other hand, appears to be better suited for smaller data requests since it is speedier and can handle smaller data transfer sizes more efficiently. This implies that RYU can service more requests in less time, making it an excellent solution for networks with a large rate of tiny data requests.

It's important to note, however, that these conclusions are based on the specific testing environment and may not necessarily apply to all network scenarios. Additionally, other factors can also impact network performance and should be considered when selecting a controller for a given network. In summary, the choice between POX and RYU controllers ultimately depends on the specific needs and requirements of the network being deployed. If the network requires high throughput and transfer rates for large amounts of data, POX may be the better choice. If the network has a high volume of small data requests, RYU may be more appropriate due to its faster processing speed.

5.3 Round Trip Time (RTT) Results

The RTT is a critical statistic for assessing network latency, and it has an influence on network performance and user experience. Lower RTT values imply faster reaction times and better network performance, whereas higher RTT values might lead to longer response times and poor network performance.

5.3.2 Single Topology

Fig. 9 displays the results of three node-to-node lines employing both POX and RYU controllers on a single topology. The figure also shows the minimum, average, and maximum RTT values in milliseconds for each of the three paths: H1 to H5, H1 to H12, H1 to H16. According to the previous statistics, the lowest and average RTT values for all three paths seem to be lower with the RYU controller than with POX. The maximum RTT values for

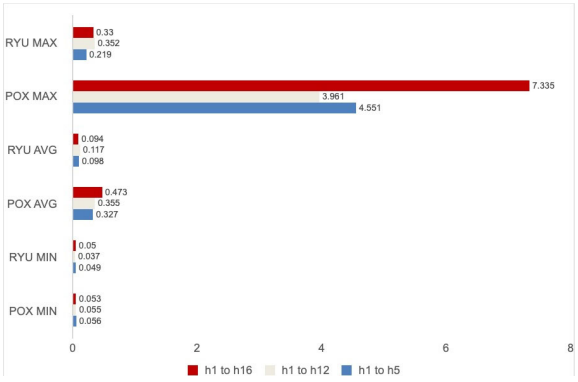

Fig. 9 RTT Delay of the Single Topology

all three paths are higher when using the RYU controller.

5.3.3 Linear Topology

The findings of three node-to-node pathways with both POX and RYU controllers on a linear topology are shown in Fig. 10 of Round-trip time (RTT). The figure also shows the minimum, average, and maximum RTT values for each of the three pathways in milliseconds. Similar to the previous results, it appears that the RYU controller has lower minimum and average RTT values than POX for all three pathways. However, the maximum RTT values for all three pathways are greater with the RYU controller.

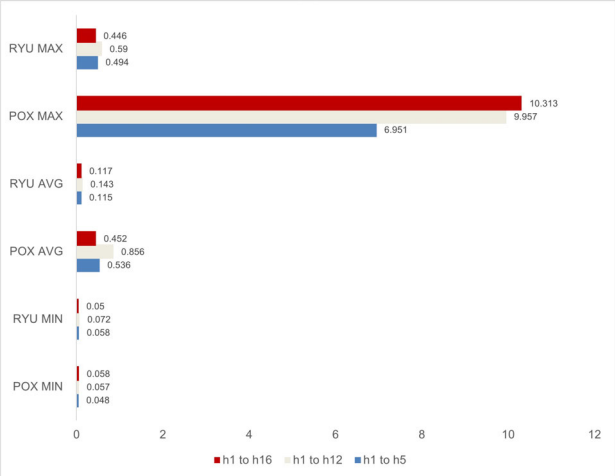

Fig. 10 RTT Delay of the Linear Topology

5.3.4 Tree Topology

Fig. 11 shows the results of three node-to-node pathways on a tree topology with both POX and RYU controllers. The figure also displays the minimum, average, and maximum RTT values for each of the three pathways in milliseconds. The lowest and average RTT values appear to be lower with the RYU controller than with the POX controller for all three routes. The maximum RTT values are greater with the RYU controller for paths h1 to h12 and h1 to h16, whereas POX has a higher maximum RTT value for paths h1 to h5.

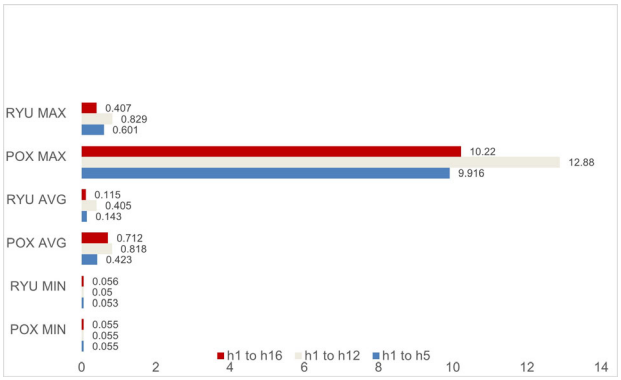

Fig. 11 RTT Delay of the Tree Topology

According to the findings shown in the previous figure, it appears that the RYU controller typically surpasses the POX controller in terms of RTT length and stability, according to the Round-trip time (RTT). With the RYU controller compared to POX, the minimum and average RTT values for all three pathways are often lower. This shows that, as compared to POX, RYU can offer quicker response times and reduced latency. Additionally, for all three pathways, the maximum RTT values are greater with RYU, indicating that RYU's RTT values may be more constant and reliable.

The POX controller, as opposed to RYU, has higher minimum and average RTT values for all three pathways, suggesting longer reaction times and greater latency. Additionally, for the paths h1 to h12 and h1 to h16, the maximum RTT values are higher with POX, which implies that POX may experience greater variations and less consistency in its RTT values. Comprehensively, the RTT results indicate that POX is more appropriate for networks that emphasize high throughput but are ready to accept greater latency and less reliable performance, whereas RYU may be more suited for low-latency networks that need quick reaction times and steady performance.

RTT, throughput, and bandwidth are all crucial parameters for assessing the performance of a network. The results shown in the previous presented figures (Fig.9 , Fig. 10 and Fig.11) above demonstrates that the chosen network controller can have a significant impact on these metrics.

The term "bandwidth" refers to the volume of data that a network can transfer in a specific amount of time. The RYU controller outperformed the POX controller in terms of bandwidth performance across all three examined topologies. With a bandwidth of 38.5 Gbits/sec for the RYU controller and 10.4 Gbits/sec for the POX controller, the tree topology offered the best bandwidth performance for both controllers. The single topology had the lowest bandwidth performance for both controllers, with the RYU

controller achieving a bandwidth of 30.0 Gbits/sec and the POX controller achieving a bandwidth of 27.4 Gbits/sec. These findings suggest that a network's capacity to transport massive amounts of data quickly and efficiently might be significantly impacted by the choice of network controller.The quantity of data that can be processed between two nodes in a second is referred to as throughput. According to the study's throughput findings, the RYU controller is more appropriate for networks with lower amounts of data, while the POX controller is better suited for networks with large amounts of data. The RYU controller surpassed the POX controller in terms of throughput in the linear topology, while the POX controller exceeded it in the tree and single topologies. These results imply that network architecture and data transmission size should be taken into account when selecting a network controller.

When a packet travels from a source node to a destination node and is received as a response, this is referred to as the RTT. The RYU controller surpasses the POX controller in terms of time, duration, and stability of each trip time, with no major variations noted, according to the RTT measurements gathered for the study. For optimization and troubleshooting purposes, it might be helpful to know the lowest, maximum, and average RTT values. In conclusion, the findings imply that selecting a network controller can significantly affect network performance. In terms of bandwidth performance, the RYU controller generally outperformed the POX controller; however, the POX controller was better suited for data-intensive networks in terms of throughput. In terms of RTT measurements, the RYU controller fared better than the POX controller, demonstrating that it is a more rapid and stable controller. Measuring network performance parameters like bandwidth, throughput, and RTT can help enhance and optimize network performance.

4. Conclusion

In this study, we assessed the performance of two SDN controllers, POX and Ryu, over three network topologies: linear, single, and tree. We used Mininet, an open-source simulation tool, to build the network topologies, then Python in conjunction with Mininet's command-line interface (CLI) to assess the controllers' efficiency in terms of bandwidth, throughput, and round-trip time.

The choice of network controller can significantly affect network performance, according to research, which evaluated the findings using both POX and Ryu controllers. The Ryu controller outperformed the POX controller in terms of bandwidth performance across the three topologies under evaluation. When it comes to throughput, the RYU controller is better suited for networks with less data than the POX controller, which is better for networks with more data. In terms of round-trip time (RTT), the Ryu controller fared better than the POX controller, proving that it is a speedier and more dependable controller. In order to enhance and optimize network performance, the study underlines the significance of monitoring network performance metrics, including bandwidth, throughput, and RTT. This paper also demonstrates the importance of selecting the best network controller especially when considering healthcare applications where some of them there is a need to provide the service with zero delay as much as possible. Furthermore, other applications that requires for example video streaming can seek network management that maximize throughput. Moving forward, there are several possible areas for future. For example, expanding the network's scalability by adding more switches and hosts. This allowed us to assess how the controllers handle larger quantities of traffic and how network capacity affects their performance. In addition to bandwidth, throughput, and round-trip time, we might look at other performance indicators such as packet loss, delay, and jitter. This would allow us to acquire a more complete understanding of the controllers' behavior in different network scenarios. In addition, comparing the performance of POX and Ryu to other SDN controllers, such as OpenDaylight and Floodlight, would help us establish which controller works best in various network setups. This would offer insight into the strengths and shortcomings of various controllers and aid in deciding which to employ in certain settings.

References

[1] M. Yaghoubi., K. Ahmed, and Y. Miao, (2022). Wireless body area network (WBAN): A survey on architecture, technologies, energy consumption, and security challenges. Journal of Sensor and Actuator Networks, 11(4), 67.

[2] Md. T. Arefin, M. H. Ali, and A. K. M. F. Haque, "Wireless Body Area Network: An Overview and Various Applications," Journal of Computer and Communications, vol. 05, no. 07, pp. 53–64, 2017, doi: 10.4236/jcc.2017.57006.

[3] K. Hasan, X.-W. Wu, K. Biswas, and K. Ahmed, "A Novel Framework for Software Defined Wireless Body Area Network."

[4] M. Cicioğlu and A. Çalhan, "SDN-based wireless body area network routing algorithm for healthcare architecture," ETRI Journal, vol. 41, no. 4, pp. 452–464, 2019, doi: 10.4218/etrij.2018-0630.

[5] M. Yaghoubi, K. Ahmed, and Y. Miao, "Wireless Body Area Network (WBAN): A Survey on Architecture, Technologies,

Energy Consumption, and Security Challenges," Journal of Sensor and Actuator Networks, vol. 11, no. 4, p. 67, 2022.

[6] P. T. Sharavanan, D. Sridharan, and R. Kumar, "A Privacy Preservation Secure Cross Layer Protocol Design for IoT Based Wireless Body Area Networks Using ECDSA Framework," J Med Syst, vol. 42, no. 10, Oct. 2018, doi: 10.1007/s10916- 018-1050-2.

[7] H. Ben Elhadj, J. Elias, L. Chaari, and L. Kamoun, "A Priority based Cross Layer Routing Protocol for healthcare applications," Ad Hoc Networks, vol. 42, pp. 1–18, May 2016, doi: 10.1016/j.adhoc.2015.10.007.

[8] J. C. Correa-Chica, J. Felipe Botero-Vega, and N. Gaviria-Gómez, "Cross-layer designs for energy efficient wireless body area networks: A review," Revista Facultad de Ingenieria, vol. 2016, no. 79, pp. 98–118, 2016, doi: 10.17533/udea.redin.n79a10.

[9] Y. Qu, G. Zheng, H. Ma, X. Wang, B. Ji, and H. Wu, "A Survey of Routing Protocols in WBAN for Healthcare Applications," 2019.

[10] V. Bhanumathi and C. P. Sangeetha, "A guide for the selection of routing protocols in WBAN for healthcare applications," Human-centric Computing and Information Sciences, vol. 7, no. 1. Springer Berlin Heidelberg, Dec. 01, 2017. doi: 10.1186/s13673-017-0105-6.

[11] F. Fanian and M. Kuchaki Rafsanjani, "Cluster-based routing protocols in wireless sensor networks: A survey based on methodology," Journal of Network and Computer Applications, vol. 142. Academic Press, pp. 111–142, Sep. 15, 2019. doi: 10.1016/j.jnca.2019.04.021.

[12] J. Anand and D. Sethi, "Comparative analysis of energy efficient routing in WBAN," in 3rd IEEE International Conference on , Institute of Electrical and Electronics Engineers Inc., Jul. 2017. doi: 10.1109/CIACT.2017.7977373.

[13] H. Taleb, A. Nasser, G. Andrieux, N. Charara, and E. Motta Cruz, "Wireless technologies, medical applications and future challenges in WBAN: a survey," Wireless Networks, vol. 27, no. 8, pp. 5271–5295, Nov. 2021, doi: 10.1007/s11276-021-02780-2.

[14] S. Yahiaoui, M. Omar, A. Bouabdallah, E. Natalizio, Y. Challal, and Y. C. An, "An energy efficient and QoS aware routing protocol for wireless sensor and actuator networks," International Journal of Electronics and Communications Archiv für 66

Elektronik und Übertragung-stechnik, vol. 83, pp. 193–203, 2018, doi:

10.1016/j.aeue.2017.08.045ï.

[15] K. Z. Ghafoor, L. Kong, D. B. Rawat, E. Hosseini, and A. S. Sadiq, "Quality of service aware routing protocol in software-defined internet of vehicles," IEEE Internet Things J, vol. 6, no. 2, pp. 2817–2828, Apr. 2019, doi: 10.1109/JIOT.2018.2875482.

[16] S. AlQahtani and A. Alotaibi, "A route stability-based multipath QoS routing protocol in cognitive radio ad hoc networks," Wireless Networks, vol. 25, no. 5, pp. 2931–2951, Jul. 2019, doi: 10.1007/s11276-019-02014-6.

[17] I. Z. Bholebawa and U. D. Dalal, "Performance analysis of SDN/openflow controllers: POX versus floodlight," Wirel Pers Commun, vol. 98, no. 2, pp. 1679– 1699, Jan. 2018, doi: 10.1007/s11277-017-4939-z.

[18] S. Askar and F. Keti, "Performance Evaluation of Different SDN Controllers: A Review," 2021, doi: 10.5281/zenodo.4742771.

[19] M. Z. Abdullah, N. A. Al-awad, and F. W. Hussein, "Performance Comparison and Evaluation of Different Software Defined Networks Controllers," International Journal of Computing & Network Technology, vol. 06, no. 02, pp. 36–41, May 2018, doi: 10.12785/ijcnt/060201.

[20] B. Preveze, A. Alkhayyat, F. Abedi, A. M. Jawad, and A. S. Abosinnee, "SDN-Driven Internet of Health Things: A Novel Adaptive Switching Technique for Hospital Healthcare Monitoring System," Wirel Commun Mob Comput, vol. 2022, 2022, doi: 10.1155/2022/3150756.

[21] I. Z. Bholebawa and U. D. Dalal, "Design and Performance Analysis of OpenFlow-Enabled Network Topologies Using Mininet," International Journal of Computer and Communication Engineering, vol. 5, no. 6, pp. 419–429, 2016, doi: 10.17706/IJCCE.2016.5.6.419-429.

[22] L. Zhu et al., "SDN Controllers: A Comprehensive Analysis and Performance Evaluation Study," ACM Computing Surveys, vol. 53, no. 6. Association for Computing Machinery, Feb. 01, 2021. doi: 10.1145/3421764.

[23] A. T. Albu-Salih, "Performance evaluation of ryu controller in software defined networks," Journal of al-qadisiyah for computer science and mathematics, vol. 14, no. 1, p. Page-1, 2022.

[24] H. M. Noman and M. N. Jasim, "POX controller and open flow performance evaluation in software defined networks (SDN) using mininet emulator," in IOP conference series: materials science and engineering, IOP Publishing, 2020, p. 012102.

[25] Y. Zhang, M. Chen,Performance evaluation of Software-Defined Network (SDN) controllers using Dijkstra's algorithm. Wireless Netw 28, 3787–3800 (2022). https://doi.org/10.1007/s11276-022-03044-3

[26] S. Shamim, "Performance analysis of different open flow based controller over software defined networking," Global Journal of Computer Science and Technology, vol. 18, no. C1, pp. 11–15, 2018.

[27] D. S. Rana, S. A. Dhondiyal, and S. K. Chamoli, "Software defined networking (SDN) challenges, issues and solution," Int J Comput Sci Eng, vol. 7, no. 1, pp. 884–889, 2019.

Lama Alfafie recently graduated from King Saud University with a bachelor's degree in computer networks. She has a strong interest in cybersecurity and has acquired technical and administrative expertise in areas such as network security and cybersecurity.

Haya Al-Anzi received a bachelor's degree in computer networks from King Saud University. She is currently a coop trainee at KACST in the Cybersecurity Institute.

Nujud Alnajem is doing her bachelor's degree in computer networks at King Saud University. She is currently a coop trainee at KACST in the Cybersecurity Institute.

Rawan Almutiri holds a bachelor's degree in computer networks from King Saud University. She is currently a trainee in the General Administration of King Saud bin Abdulaziz University for Health Sciences at the Information Technology Department in the field of enterprise resource planning (ERP) Oracle.

Areej Alotaibi holds a bachelor's degree in computer networks from King Saud University. She will start her training in the IT department at Dalah Hospital.

Nourah Alhazri holds a bachelor's degree in computer networks from King Saud University. She is currently working as a trainee at Saudi Aramco.

Awatif Alqahtani is an assistant professor, Computer Science and Engineering Dept. College of Applied Studies and Community Service, King Saud University. She received her PhD degree from Newcastle University, UK. Her research interests include Cloud computing, IoT, SLA, and their applications.