Evaluating the SDN POX Controller on a star topology IoT network

Everton Roberto Zanotelli
Computer Institute
Federal University of Bahia UFBA
Salvador, Brazil
evertonzanotelli@ufba.br

***Abstract*—The inevitable growth of connected devices and the multitude of services that they now provide are demanding an intelligent approach on the IoT architecture. Such request can be met by an emerging technology for network transformation as Software-defined networking (SDN) that can dynamically adapt the network to different behaviours, optimize resource allocation, enhance security, and enable seamless scalability. In the SDN architecture, the controller takes on a crucial and central role as the main component of that network. In large SDN networks, there may be multiple controllers or controller domains that work together to manage the network and this is the most common scenario for IoT networks. In this paper we are going to emulate an IoT network on a star topology using the POX controller measuring it's performance in therms of Throughput, Latency and Round Time Trip.**

***Index Terms*—IoT, SDN, Mininet, POX, NOS**

I. Introduction

The Internet of Things (IoT) has experienced significant growth, transforming various industries. To effectively handle the challenges posed by IoT deployments, a robust network infrastructure is crucial. Software-Defined Networking (SDN) plays a key role in achieving this [1]. SDN allows for dynamic control and management of IoT devices and traffic, enabling organizations to build efficient networks tailored to optimize IoT performance and security.

The integration of SDN in the IoT landscape offers several benefits. It enables the creation of customized virtual networks, ensuring optimal resource allocation and low latency for time-sensitive IoT devices. SDN also enhances security by providing centralized management and granular access control [2]. By leveraging SDN technology, organizations can effectively manage and control their IoT deployments, ensuring reliable connectivity and efficient resource utilization.

A. Internet of Things

The Internet of Things represents a chain of interconnected physical devices, sensors, and actuators that communicate and exchange data over the network. Its vast potential has resulted in rapid adoption across various industries like healthcare, manufacturing and transportation. However, supporting a large number of diverse devices with varying data rates and communication patterns poses significant challenges for traditional network architectures [3].

IoT networks demand a scalable infrastructure capable of handling the massive influx of data generated by numerous devices. Additionally, IoT devices often have resource-constrained hardware, limited power, and may operate in dynamic environments, further complicating network management [4]. As a result, network protocols and architectures need to be tailored to accommodate the unique characteristics of IoT, such as low-power consumption, device heterogeneity, and mobility support.

B. Software-Defined Networking

Software-Defined Networking (SDN) has emerged as a promising paradigm to overcome the difficulties to deploy an reliable and secure IoT network. SDN decouples the control plane from the data plane, centralizing network management and providing a holistic view of the network. [5] By leveraging SDN, network administrators can dynamically control and reconfigure network behavior through a centralized SDN controller, offering increased flexibility, scalability, and programmability.

The SDN architecture separates the network's control logic from the underlying physical infrastructure, enabling network operators to define and enforce policies consistently across the entire network. This programmability empowers researchers and developers to experiment with novel network protocols, routing algorithms, and resource management techniques. In the context of IoT, SDN can significantly improve network performance, adaptability, and security [5].

C. Mininet

To evaluate the efficacy of SDN controllers and network protocols, researchers often rely on network simulation tools. One such widely-used tool is Mininet, a open-source network emulator that enables the creation of virtual network topologies and facilitates the evaluation of SDN architectures and applications. Mininet provides an efficient and cost-effective platform for replicating real-world network scenarios, allowing researchers to assess the impact of various factors on network performance [6].

The ability to simulate diverse network topologies and experiment with different traffic patterns makes Mininet an invaluable tool for evaluating the performance of SDN-based IoT networks. It allows researchers to investigate the effects of controller algorithms, network congestion, and other factors that influence throughput, latency, and jitter.

D. Pox SDN Controller

In this paper, our primary focus is on evaluating the Pox SDN controller's effectiveness in enhancing the performance of an IoT network. Pox is an open-source SDN controller written in Python that provides a flexible and extensible platform for network management [7]. It has gained considerable popularity due to its ease of use and extensive support for various network protocols [8].

We aim to assess the impact of the Pox controller on network throughput, latency, and RTT in an IoT environment. By conducting experiments on a simulated IoT network using Mininet, we will measure the performance of the Pox controller in a IoT oriented network topology. Through our evaluation, we seek to provide insights into the strengths and limitations of the Pox controller, offering valuable guidance for researchers, network administrators, and IoT practitioners.

E. IoT Network Topology

The star network topology is a popular choice for IoT networks due to its ability to delegate communication complexity to the central node. This topology allows all other nodes to communicate according to their time or frequency slot, while the central switch serves as the gateway to the internet. In case of a device failure, the rest of the devices are not affected [9]. Additionally, an IoT network based on star topology can easily identify and isolate a faulty node since each node has its own separate connection with the switch. However, a point of concern is that with data packets having to pass through the central node, the network has a single point of failure.

II. Metodology

To identify research gaps and establish research objectives, a comprehensive review of existing literature on network traffic analysis using SDN controllers is conducted. Next, a custom network is constructed for the SDN environment using the Mininet tool. The next crucial step involves implementing an open-source SDN controller within the Mininet topology. Once the network is established, data traffic is generated from the source to the destination node. Finally, an evaluation of the SDN network's performance is undertaken using various metrics. The proposed work is compared with the existing state of the art to discern its contributions and advancements.

A. Proposed Analysis

The SDN architecture can be implemented using three main components: network devices, a controller, and an application. The controller is a primary entity in the SDN network, receiving commands from the application layer and forwarding them to the network layer. The application layer communicates with the SDN controller through various APIs.

Due to the high cost of implementing the physical framework of SDN architecture using real testbeds and hardware, simulators or emulators are often used for research in the SDN environment. For this research paper, experimentation was performed using Mininet, an open-source network emulator. The SDN architecture was created to transfer data traffic between nodes via a specific SDN controller in the emulator.

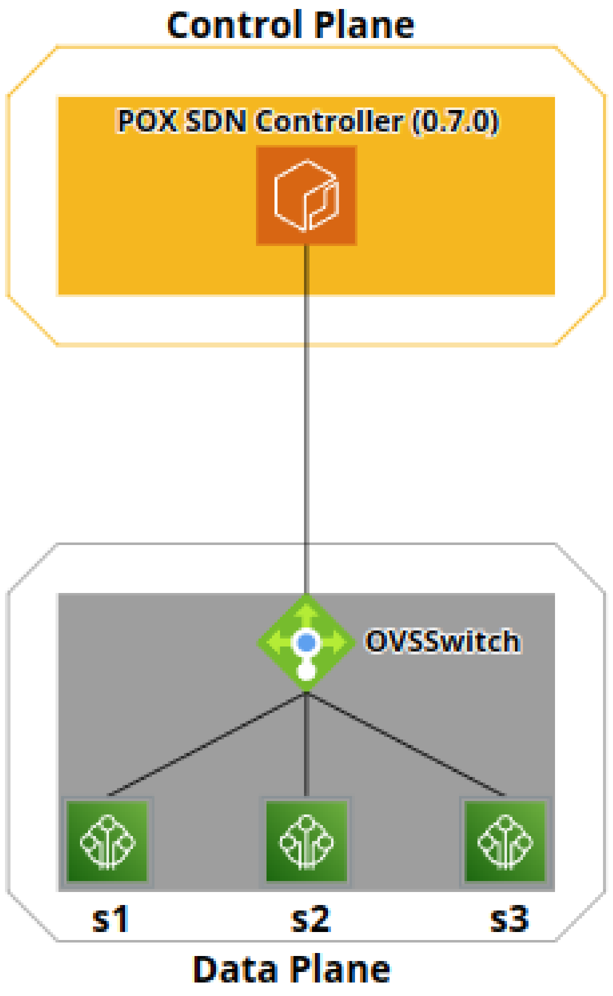

Fig. 1. Proposed Architecture.

The architecture that is being evaluated in this study is composed of several key components. At the core of the architecture is the POX SDN Controller, which serves as the central management point for the IoT gateway. The gateway itself is represented by an OpenFlow switch, which provides connectivity for three IoT sensors. These sensors are responsible for collecting and transmitting data to the gateway, where it can be processed and analyzed by the controller.

III. Result Analysis

A. Bandwidth

POX is an open-source SDN controller that can be evaluated using the iperf benchmark tool to generate TCP data traffic and measure bandwidth performance. During TCP traffic, the source node initiates a connection by sending a TCP SYN request packet to the destination node through the SDN switch. The bandwidth of TCP traffic between nodes is measured by examining the number of packets transmitted to the destination host using various processes in a regulated test. Here, the iperf command has been executed for 10 s. The maximum data transferred between s1 to s2 is 35.6 Gbps, s1 to s3 is 40.3 Gbps, s2 to s3 is 45.1 Gbps.

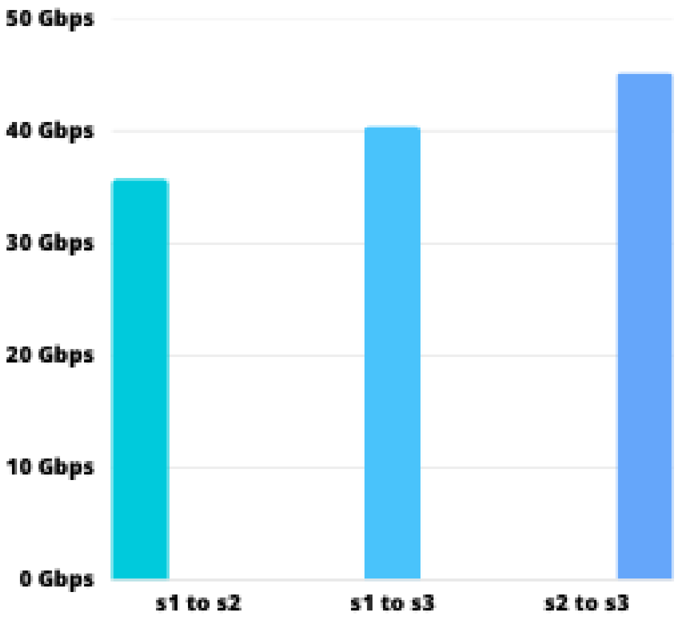

Fig. 2. Bandwidth

B. Throughput

Throughput refers to the maximum amount of data that can be transmitted between two network nodes in one second. In TCP traffic, one node acts as the client and the other as the server. To measure the throughput of a controller, we used the iperf3 utility. The throughput was tested on the client side over a 10-second period, with data collected from the opposite end of the network. The result of this test is the throughput value. The highest and lowest throughput is 41.4 and 38.8 Gbps between s1 to s2, 42.8 and 38.8 Gbps between s1 to s3, 41.1 and 39.7 Gbps between s2 to s3 respectively.

C. Round-Trip Time

Round trip time (RTT), also known as ping test time, is the total duration taken for a data packet to travel from a specific source host to the destination host and for the acknowledgment packet to return to the source. The ping utility uses the Internet Control Message Protocol (ICMP)

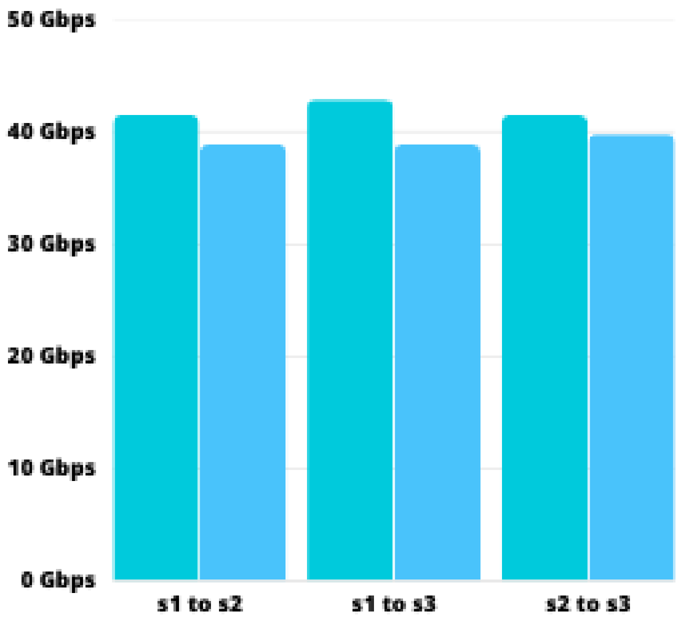

Fig. 3. Throughput

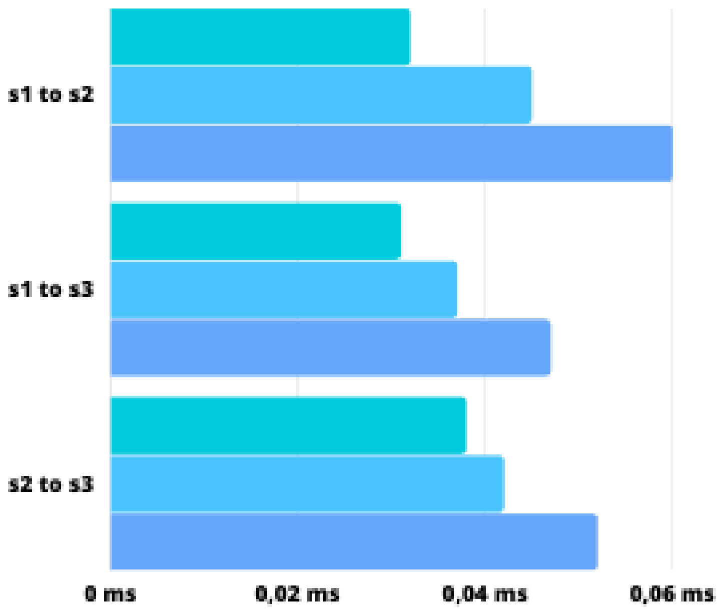

Fig. 4. Round-Trip Time

to measure RTT. The first packet is ignored in our study because in a Mininet environment using an SDN controller, the first ping in a sequence may have a higher RTT due to the way the OpenFlow protocol works. When the first ping is sent, the switch may not have a matching flow entry and sends the packet to the controller for processing. The controller determines the appropriate action, installs a flow entry in the switch, and forwards the original packet. This process can add an additional delay to the first ping. Subsequent pings that match the installed flow entry can be processed by the switch without involving the controller, reducing their RTT [10]. The minimum RTTs of the proposed SDN topology are measured as 0.032 ms, 0.031 ms, 0.038 ms for s1 to s2, s1 to s3 and s2 to s3 respectively. The average RTTs of the proposed topology are measured as 0.045 ms, 0.037 ms, 0.042 ms for s1 to s2, s1 to s3 and s2 to s3 respectively and the maximum RTTs of the proposed topology are 0.060 ms, 0.047 ms, 0.052 ms for s1 to s2, s1 to s3 and s2 to s3 respectively.

IV. Discussion

Based on the data collected, we can notice that the performance of the POX controller varies significantly when implemented on different network topologies. Specifically, when comparing the results of previous tests conducted on a star topology to those obtained on a tree topology, commonly used in IoT environments. In terms of network performance, the star topology appears to be the superior choice between the two architectures in this scenario. This suggests that, when possible, it may be advantageous to opt for a star topology in order to achieve optimal results. Further research is needed to fully understand the underlying mechanisms driving these differences and to determine whether these findings can be generalized to other network topologies and controllers.

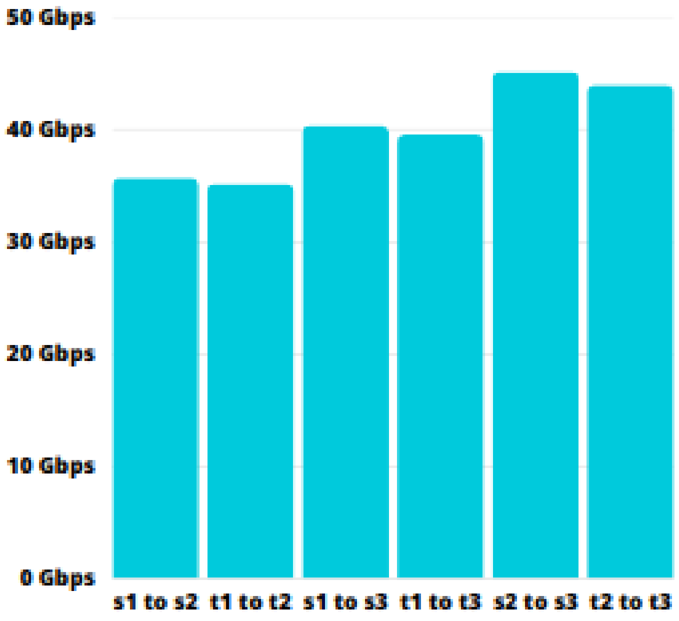

Fig. 5. Bandwidth of Star vs Tree topology

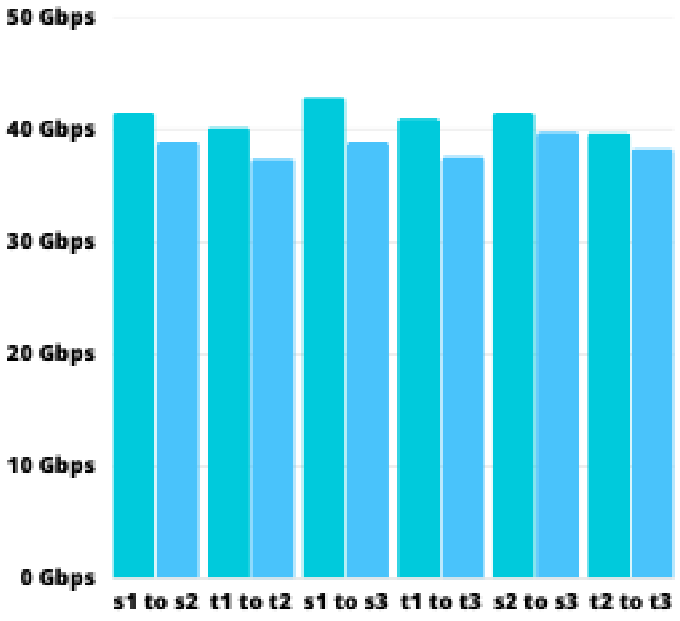

Fig. 6. Throughput of Star vs Tree topology

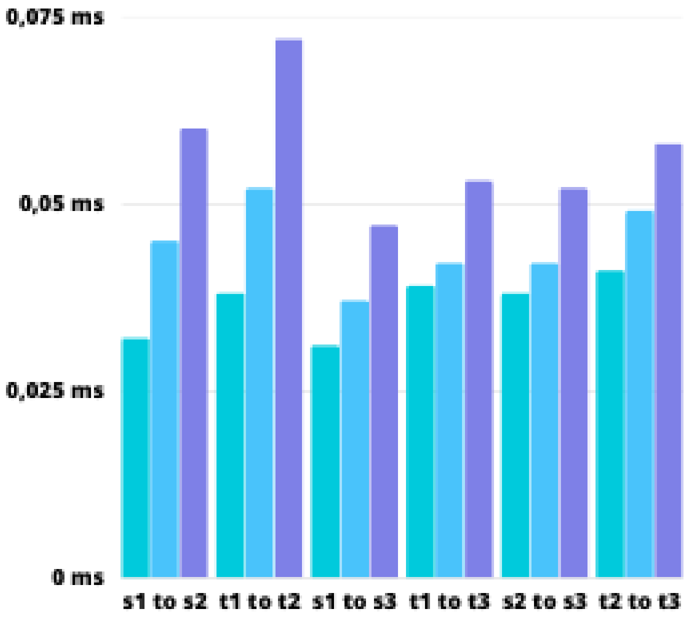

Fig. 7. RTT of Star vs Tree topology

V. Conclusion

The field of Internet of Things (IoT) is in a state of constant evolution, propelled by technological advancements. This is particularly pertinent in the realm of Software-Defined Networking (SDN), where the controller assumes a important role in monitoring and analyzing real-time data traffic. This process is crucial for comprehending the flow of data packets between source and destination hosts, as well as the requirements imposed by IoT on the network.

Assessing and benchmarking the performance of a controller is always a challenge. In this study, we employ an SDN architecture utilizing the open-source POX controller to detail network traffic. Our objective is to enhance traffic routing performance metrics in IoT environments, thereby augmenting the network's capacity to optimally transport data from IoT sensors to their respective applications. Through an evaluation of the POX controller's performance, we illustrate how traffic analysis can result in superior resource utilization, improved network performance, more efficient data traffic management, reduced costs, and increased innovation.

The POX controller is a widely-used open-source SDN controller written in Python. Our findings will be of relevance to researchers investigating SDN and controller evaluation. It is important to note that the majority of controllers proposed in the literature lack available implementations, and the details provided are insufficient for third-party replication. Consequently, apart from theoretical comparisons, it is not feasible to evaluate them. Furthermore, some controllers with publicly available implementations are either unmaintained or rudimentary in nature. For instance, some studies utilize initial (outdated) implementations of it's controllers, which inherit the limitations of older code, thereby affecting the reliability and replicability of their performance.

Nonetheless, our findings will be of interest to researchers examining SDN and controller evaluation for various network topologies in which IoT can be implemented to achieve optimal outcomes in each specific scenario.

REFERENCES

[1] Elif Bozkaya and Berk Canberk. 2020. SDN-enabled deployment and path planning of aerial base stations. Computer Networks 171 (April 2020), 107125.

[2] Nunes, A., Mendonca, M., Nguyen, X.N., Obraczka, K.: A survey of software-defined networking: past, present, and future of programmable networks. Commun. Surv. Tutor. IEEE 16(3), 1617–1634 (2014)

[3] Bannour, F., Souihi, S., and Mellouk, A. (2017). Distributed SDN control: Survey, taxonomy, and challenges,. IEEE Communications Surveys and Tutorials, 20, 333–354

[4] M. Baddeley et al., "Evolving SDN for low-power IoT networks," in IEEE NetSoft, 2018, pp. 71–79.

[5] Zhu, Liehuang, et al. "SDN controllers: A comprehensive analysis and performance evaluation study." ACM Computing Surveys (CSUR) 53.6 (2020): 2-6.

[6] Mininet Team. 2018. Mininet: An Instant Virtual Network on Your Laptop (or other PC) Mininet. Retrieved Jul 13,2023 from http://mininet.org/.

[7] Idris Z. Bholebawa and Upena D. Dalal. 2018. Performance analysis of SDN/OpenFlow controllers: POX versus floodlight. Wireless Personal Communications 98, 2 (2018).

[8] McCauley, M. (2009). POX (online). http://www.noxrepo.org/.

[9] Mamat, H., B. H. Ibrahim, and M. P. Sulong. "Network topology comparison for internet communication and IoT connectivity." 2019 IEEE Conference on Open Systems (ICOS). IEEE, 2019.

[10] Wang, Shie-Yuan. "Comparison of SDN OpenFlow network simulator and emulators: EstiNet vs. Mininet." 2014 IEEE Symposium on Computers and Communications (ISCC). IEEE, 2014.