Investigation Into The Performance of Software-Defined and Conventional Networking

1 Abstract

This report evaluates Software-Defined Networking (SDN) against conventional networking. The report is split into two distinct sections. Section 1 contains a literature review on SDN and its benefits, issues and future directions. Section 2 is an SDN topology performance analysis with comparisons made against the performance of a traditional topology. The performance of TCP and UDP protocols is investigated. Performance metrics include bandwidth, transfer per interval for TCP and included jitter (inter-packet delay variance) for UDP. Comparisons for UDP are made at 10, 100 and 1000 Mbits/sec bandwidth. Results showed that the conventional topology had increased performance using TCP, and the SDN topology performed better using the UDP protocol. Both topologies showed promise for implementation depending on the network requirements.

Word Count: 2938 – 2200 (Minus Titles, Tables, Refrences, Abstract)

Keywords: SDN, Conventional, Bandwidth, Performance, UDP, TCP

2 Introduction

Software Defined Networking (SDN) is an emerging set of concepts that abstract the control from individual network devices and proposes using software to replace much of management and administration (Macedo et al., 2015). The control plane and data plane are concepts that separate forwarding intelligence and forwarding hardware. In SDN, the control element (controller) resides in the logical control plane and makes decisions for forwarding elements in the data plane. Control is based on a logical and global network overview provided by an abstraction protocol that communicates between the planes. The OpenFlow standard is the only standard for communication between the different planes. OpenFlow offers statistical collection and flow modification at the controller's discretion. This report aims to investigate the performance of SDN networks versus conventional networks in terms of TCP and UDP, analysing throughput, transfer, and jitter.

3 Literature Review

In (Farhady et al., 2015), the authors describe the main aims of SDN to separate control of the network and the forwarding of data, improve network operations performance, and encourage innovation. Similarly, the authors of (Jammal et al., 2014) describe SDN as a necessary element to promote innovation and extension of network services. SDN can benefit network administrators by reducing the configuration and maintenance required compared to legacy networks. As the authors of (Jammal et al., 2014) found, reducing administrative workload can create time and resources for developing new services and improving network users' experience.

The authors of (Karakus & Durresi, 2018) performed an analysis solely of the economic benefits of SDN, a topic typically overshadowed by the performance aspects of the technology. In (Karakus & Durresi, 2018), the authors describe the potential environmental benefits of implementing software-defined networking in place of conventional network architecture. They found that SDN is cost and

energy efficient when considered over the OPEX cost of conventional networks from efficient optimisation of the control plane.

In (Karakus & Durresi, 2017b), issues detailing challenges the centralised SDN architecture faces with the scalability of a network. The centralised controller is responsible for the decision-making in place of the network device. With tens, hundreds, or thousands of devices, the SDN controller has a limit on the processing capacity available for making data plane decisions (Karakus & Durresi, 2017b). In (Farhady et al., 2015), in 2015, the authors concluded that SDN would shift from centralised to distributed architectures to improve the scalability of the SDN network through a varied analysis of current (2015) SDN technologies. This claim could be considered accurate with results from proposals in papers (Chen et al., 2017), (Bannour et al.) and (Wu et al., 2020) showing that distributed architecture reduced the per controller load. Reduced load for a controller allows for faster decision-making and increased availability for applications to perform responsibilities.

In (Hakiri et al., 2014), the authors consider that SDN, like (Jammal et al., 2014), promotes innovation and the creation of new services. However, the authors of (Hakiri et al., 2014) are increasingly concerned about the potential security risks of SDN and the unique challenges the technology introduces. The centralisation of control means network operations are reduced to a single point of failure (Hakiri et al., 2014). The authors of (Iranmanesh & Reza Naji, 2021) investigate the impact of DDoS attacks on the centralised controller architecture. They found that if an effective DDoS attack was implemented in a portion of the network, the controller could be saturated with OpenFlow requests, affecting the entire network's performance. After investigating the distributed and centralised architectures, the authors of (Hakiri et al., 2014) found that the former provided redundancy against attacks that generally affect the centralised architecture, with secondary controllers managing what a compromised controller is unable. Their conclusion is similar to the recommendations made by (Iranmanesh & Reza Naji, 2021) after their investigation.

(Karakus & Durresi, 2017) is a paper that reviews the Quality of Service (QoS) benefits and limitations of the SDN architecture. With heterogeneous services requiring special considerations for flow management to provide a complete service for the end-user, conventional networks require many separate solutions to ensure QoS standards. The authors of (Karakus & Durresi, 2017) believe SDN can potentially provide a singular centralised solution to fine-grained flow management through its global network view and statistics collection. The authors conclude that the main challenges for SDN QoS and the lack of implementation are signalling overhead and research focusing on intra-domain over the single domain.

In (Moin et al., 2020), the direction of SDN and future trends are discussed in terms of environmental factors, unlike other papers. The authors suggest that SDN could be used to provide "green networking", a term they use to describe environmentally efficient networking. Green architecture has one or many features, such as energy efficiency or low carbon footprint (Moin et al., 2020). For SDN to become an environmentally sustainable solution in the future, the authors of (Moin et al., 2020) believe it must address its common issues as well as increase innovation for innovative consumption strategies that would, in theory, be deployed as controller applications to manage device power states. As SDN evolves, this may come as a by-product of innovations that the typical SDN architecture is supposed to bring, as (Jammal et al., 2014) suggests.

4 Experimental Setup

4.1 Simulation Tools

The investigation uses Mininet, a python-based network emulator designed for integration with SDN concepts. Mininet can support many general-purpose simulations for both conventional and SDN networks and achieves host volumes of >1000 (Muelas et al., 2018).

4.2 Performance Analysis Tools

Iperf measures the performance of network links between hosts in the network. Network performance statistics are gathered from TCP and UDP (at 10Mb/s) connections between the Iperf Server H1 and clients on H2-H6 (Table 1). Statistics are averaged to provide an average performance metric of the entire network. Further metric analysis is performed when the network is under different loads using UDP at 100Mb/s and 1000 Mb/s between nodes H5-H1 (Table 1).

Table 1 – Commands Used

TCP Server	iperf -s -p 5566 -i 1
TCP Client	iperf -c 10.0.0.1 -p 5566 -t 15
UDP Server	iperf -s -u -p 5566 -i 1
UDP Client	iperf -c 10.0.0.1 -u -b 10M -t 15 -p 5566
UDP 100	iperf -c 10.0.0.1 -u -b 100M -t 15 -p 5566
UDP 1000	iperf -c 10.0.0.1 -u -b 1000M -t 15 -p 5566

4.3 Network

Each scenario consists of 6 hosts and 4 switches. The SDN topology uses Kernel switches, while the conventional topology uses basic switches. The SDN topology contains a singular ONOS controller connected to each of the 4 Kernel switches. In Figure 9 - Conventional Topology and Figure 10 - SDN Topology, link connections are made apparent. IP addresses are automatically assigned.

5 Results

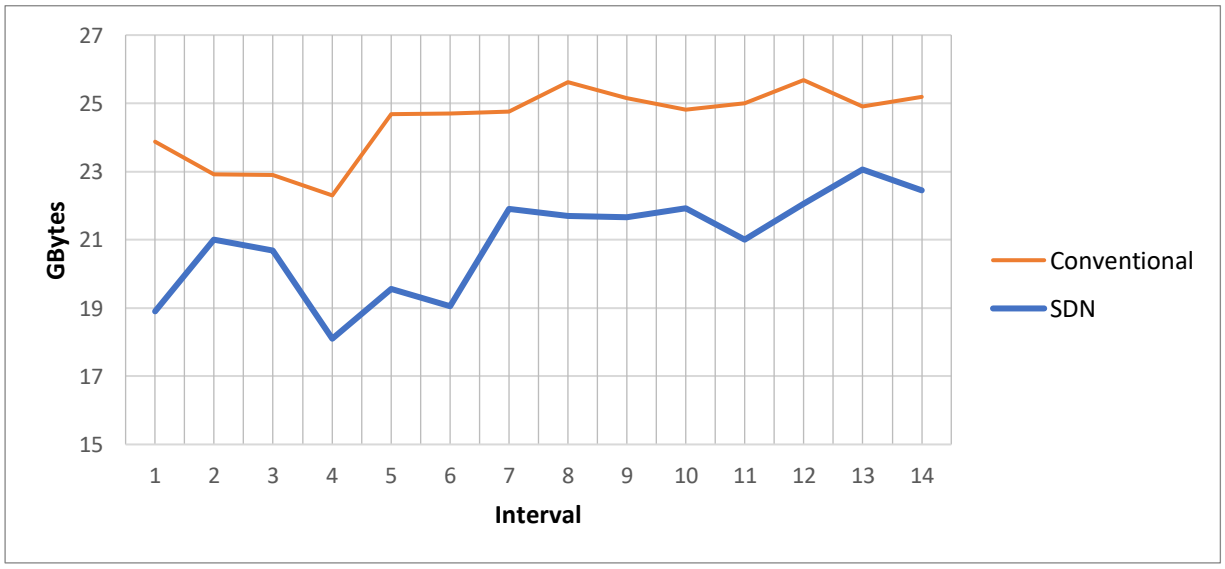


Figure 1 - Average Link Bandwidth (TCP)

Comparison 1 (Figure 1) is the average link bandwidth between hosts in each scenario over 14 intervals. The above graph shows that the conventional scenario's average bandwidth has the lowest

variance and is thus the most consistent. The conventional scenario shows a difference of 3.38 MB between its highest and lowest bandwidth. This is compared to the SDN, which had a difference of 4.96 MB. Less variance suggests greater consistency showing that the conventional scenario has TCP connections that are more stable than the SDN scenario. TCP performance is impacted by bandwidth oscillation and may cause congestion of network links from a backlog of unprocessed packets sent between links of varying bandwidths (Ren et al., 2008). Both scenarios start with lower bandwidths but steadily increase as the test progresses. However, the conventional scenario is quicker to improve the bandwidth of its connections. The SDN scenario may be slower due to a delay caused by the installation of flows from the controller onto the OVS switch. Another reason may be an increased TCP congestion window in the SDN scenario, depending on how ONOS handled the TCP connection. The SDN scenario also showed the lowest minimum (SDN: 18.1, Traditional: 22.3) and maximum (SDN: 23.06, Traditional: 25).

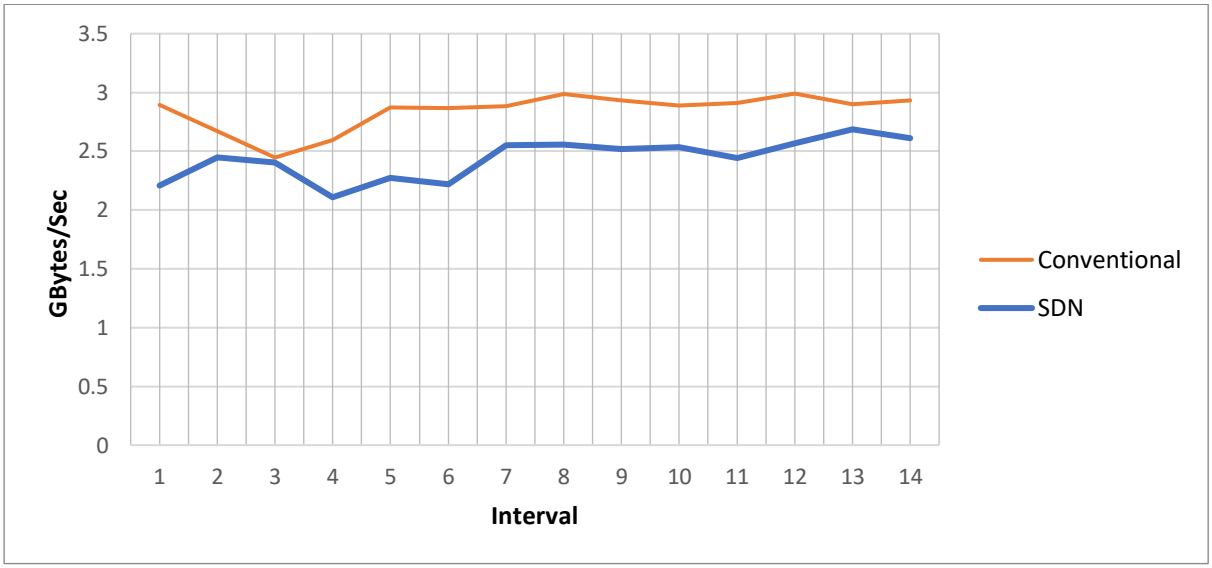


Figure 2 - Average Transfer Per Interval (TCP)

Comparison 2 (Figure 2) analyses the average volume of data transferred between hosts at each interval in a TCP connection. The performance evaluation using transfer and bandwidth shows a clear relationship between each metric, as higher bandwidth results in higher transfer. Therefore, the results in comparison 2 show similar trends to comparison 1. The conventional scenario shows the least variance between minimum and maximum and, thus, greater consistency. The conventional scenario had a variance of 0.544 MB compared to 0.578 MB in the SDN scenario. The results between scenarios are much closer than in comparison 1; however, the results still show that SDN is less consistent. The SDN scenario performed the worst in comparison 2, transferring the least volume of data (SDN: 34.1 {3.sf} Traditional: 39.8 {3.sf}).

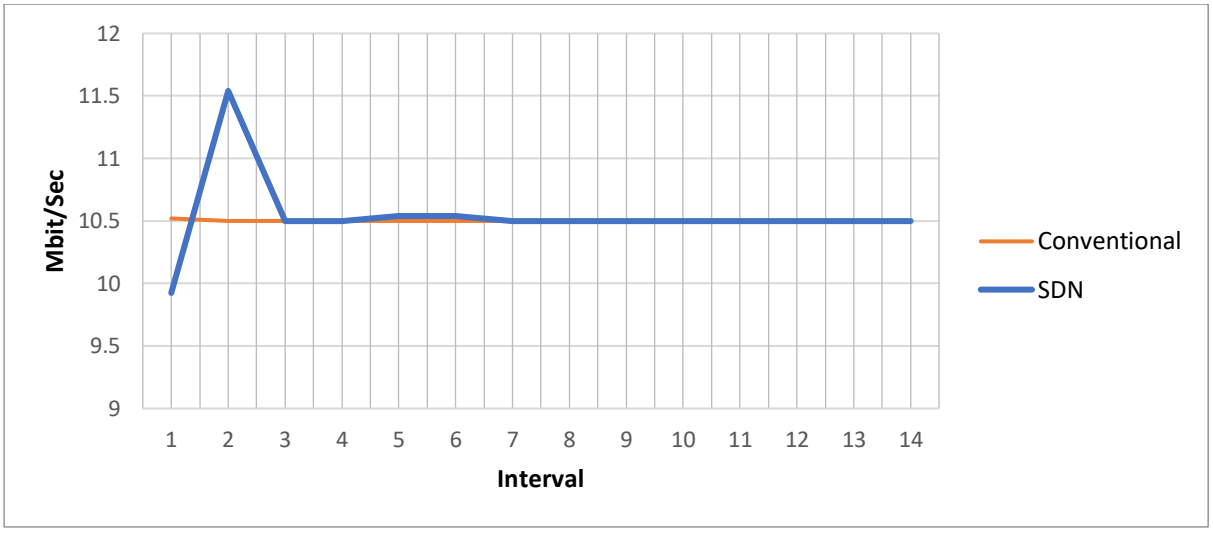


Figure 3 - Average Bandwidth (UDP)

Comparison 3 (Figure 3) compares the average bandwidth of a UDP transmission between hosts at each interval. The conventional scenario shows the most consistency with the least variance between minimum and maximum. The conventional scenario had a variance of 0.02 Mb/s compared to 1.616 Mb/s in the SDN scenario. Increased congestion at the controller at the initialisation of the test may cause more significant variance in the SDN scenario. Congestion is pushed to the controller having to process a large influx of PACKET_IN requests before implementing flow rules on the OVS switch. Congestion may not be as prominent in the TCP tests as TCP utilises a slow start to increase the possibility of a successful connection and reduce link congestion (Wang et al., 2017). However, the SDN scenario had the highest maximum average bandwidth (SDN: 11.5 {3.sf} Traditional: 10.5 {3.sf}) at interval 2. A higher maximum implies that the SDN scenario could theoretically support a higher bandwidth than the conventional scenario. However, this is only when a guaranteed connection such as TCP is not required. It could be said that there is increased congestion in the SDN network compared with the conventional network. While congestion is minimal in both scenarios, congestion is suggested through analysis of the average volume of out-of-order packets. An increase in out-of-order packets suggests issues in the network as some packets arrive after packets that were sent later in the scenario Figure 12 - SDN Data.

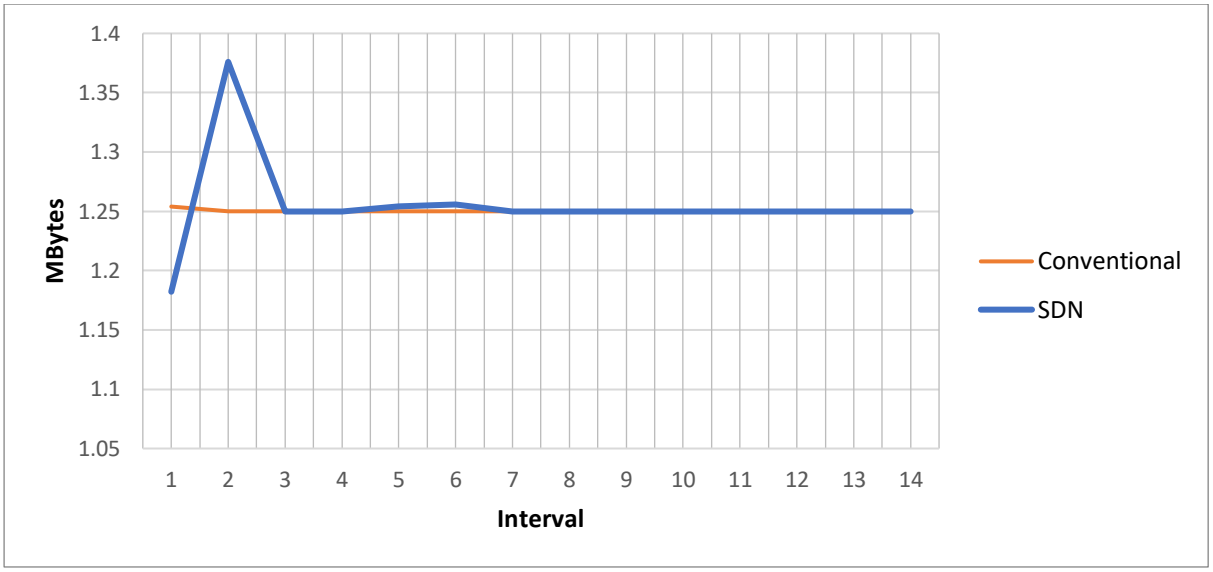


Figure 4 - Average Transfer Per Interval (UDP)

Comparison 4 (Figure 4) analyses the average volume of data transferred between hosts at each interval for a UDP transmission. Trends are similar to comparison 4 and the relations between comparisons 1 and 2. The results show similar trends to the bandwidth in both scenarios, with both scenarios stabilising after interval 3. The SDN scenario produces the most instability between intervals, with large spikes at the start of the scenario. The conventional scenario shows a minimal spike at the first interval but offers the greatest consistency after that. The SDN scenario has the highest maximum (SDN: 1.37 {3.sf}, Traditional: 1.24 {3.sf}), and the conventional scenario has the least variance with 0.008 Mb/s compared to 0.192 Mb/s {3.sf}.

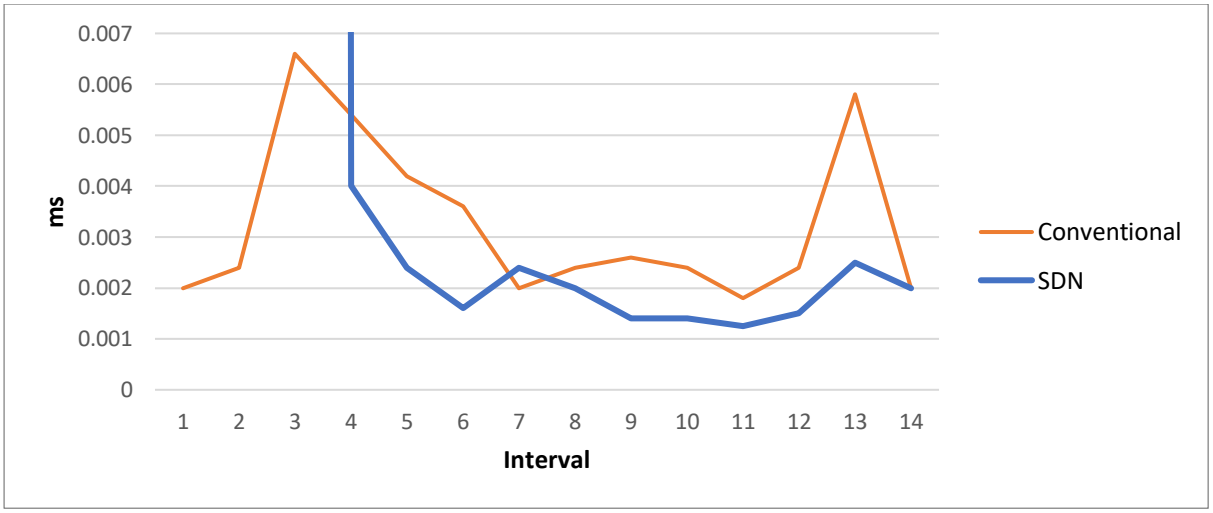


Figure 5 - Average Jitter (UDP)

Comparison 5 (Figure 5) is between scenarios' average jitter (inter-packet delay variance). The SDN scenario had far higher jitter at the initialisation of each test (SDN: 78.9 {3.sf}, Traditional: 0.002). High jitter is caused by the SDN controller requiring the first packets to be sent to the controller for inspection so the correct flow rules can be implemented on the switches (Abdallah et al., 2020). Controller congestion can cause large amounts of delay, which is drastically reduced after rule implementation. After rule implementation, the SDN scenario has, on average lower jitter than the conventional scenario Figure 12 - SDN Data.

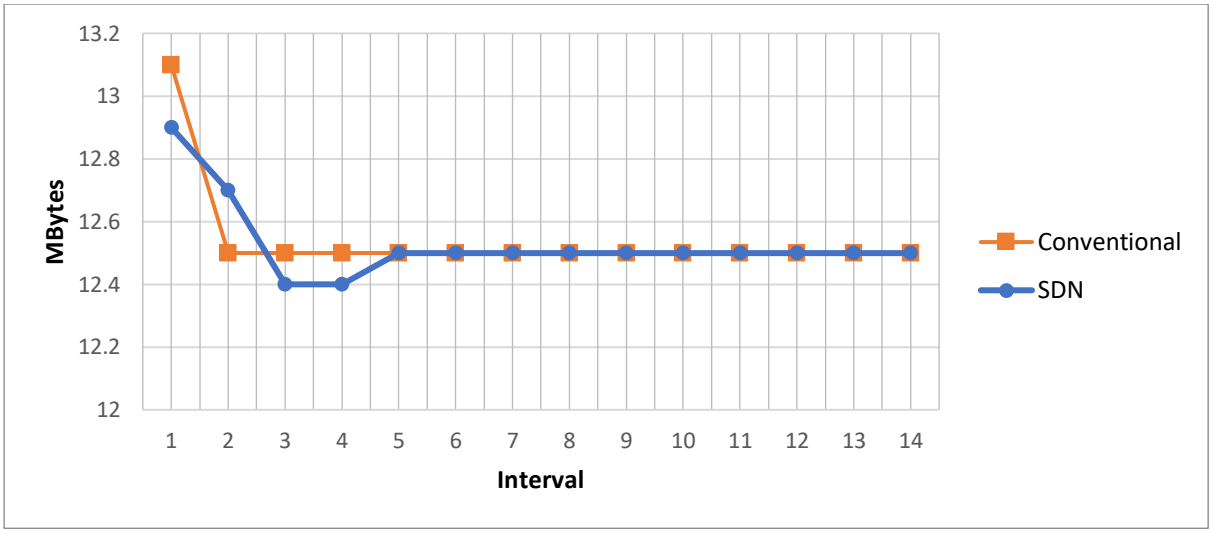


Figure 6 - 100Mb/s Bandwidth

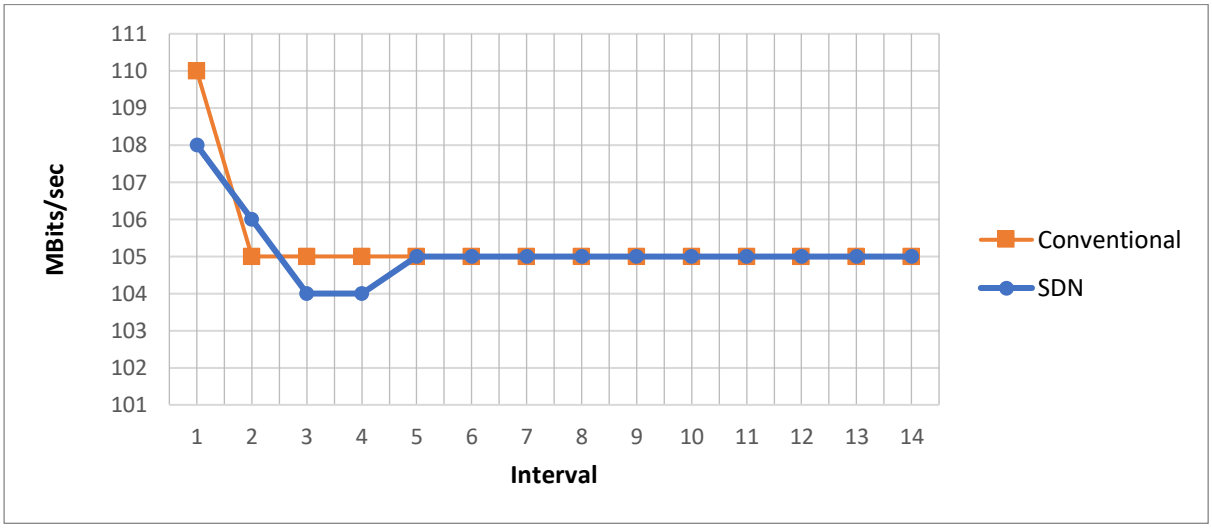


Figure 7 - 100Mb/s Transfer Per Interval

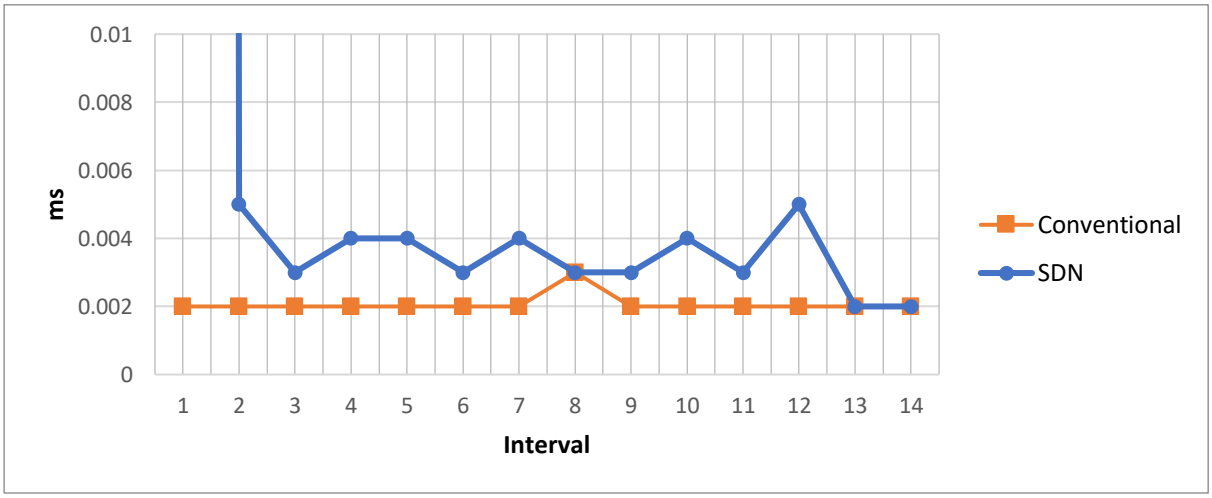
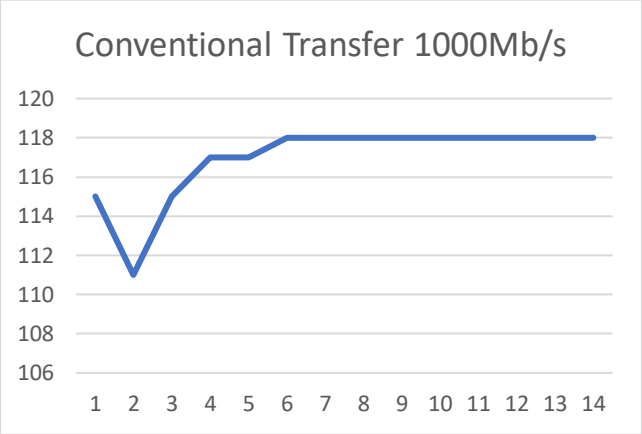
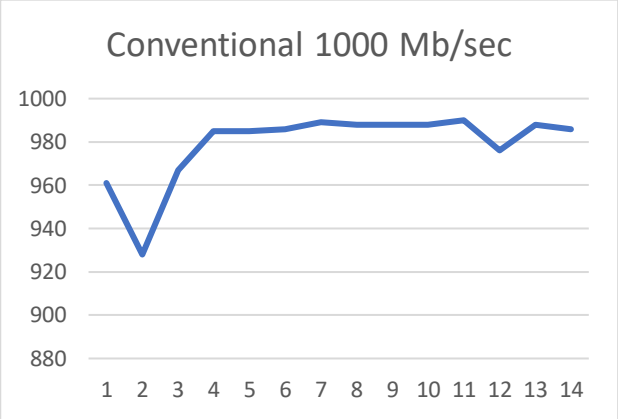


Figure 8 - 100Mb/s Jitter

Comparison 6 (Figure 6, Figure 7, Figure 8) is between scenarios at an increased load of 100Mb/s traffic. At this network load, both scenarios perform well, with similar trends to those previously seen. The conventional scenario has a higher maximum bandwidth (110 Mb/s), but both stabilise at 105 Mb/s. The conventional scenario does stabilise faster in terms of bandwidth. Similar trends are shown by analysing the transfer with near identical patterns to the bandwidth. The jitter of both scenarios also shows similar trends to the previous analysis, with a considerable delay at the initialisation stage for SDN. Increasing the bandwidth to 100 Mb/s caused little to no impact for both scenarios except what was previously discussed.





When the bandwidth was increased to 1000 Mb/s, the conventional scenario showed it struggled to meet demands and could not provide 1000 Mb/s of bandwidth but still showed similar trends to previous test cases. In the SDN scenario, the controller attempted to process the large traffic volume, but this ultimately crashed the simulation tool and, most concerningly, disabled the route between iperf client and server so no traffic could flow (Figure 13 SDN 1000Mb/s). It is speculated that the ONOS controller could not process the large volume of traffic and initialise the flow rules on the switch, or this was a limitation of the simulation software to accept flow rules from the controller.

6 Conclusion

Comparisons suggest that the SDN architecture negatively impacts the performance of the TCP protocol in contrast to the conventional architecture. The SDN architecture could not match the volume of data transferred by the conventional scenario and only managed to have a maximum bandwidth slightly above the minimum bandwidth of the traditional. The UDP comparisons showed that the SDN architecture performed better than the conventional when a connectionless protocol was used. The SDN scenario showed higher throughput and transfer compared to the conventional scenario. Both TCP and UDP test cases showed drastically higher inter-packet delay variance in the SDN scenario; this was suspected to be caused by the controller installing flow rules on the OVS switch. The conventional approach is suitable for networks that rely on guaranteed transmission and delivery (based on performance). On the other hand, it could be recommended that the SDN approach is suitable for multi-media networks (based on performance).

Bibliography

Abdallah, S., Kayssi, A., Elhajj, I. H., & Chehab, A. (2020). Performance analysis of SDN vs OSPF in diverse network environments. *Concurrency and computation*, *32*(21), n/a. https://doi.org/10.1002/cpe.5410

Bannour, F., Souihi, S., & Mellouk, A. (2018). Adaptive state consistency for distributed onos controllers.

Chen, Y., Yang, Y., Zou, X., Li, Q., & Jiang, Y. (2017). Adaptive Distributed Software Defined Networking. *Computer communications*, *102*, 120-129. https://doi.org/10.1016/j.comcom.2016.11.009

Farhady, H., Lee, H., & Nakao, A. (2015). Software-Defined Networking: A survey. *Computer networks (Amsterdam, Netherlands : 1999)*, *81*, 79-95. https://doi.org/10.1016/j.comnet.2015.02.014

Hakiri, A., Gokhale, A., Berthou, P., Schmidt, D. C., & Gayraud, T. (2014). Software-Defined Networking: Challenges and research opportunities for Future Internet. *Computer networks (Amsterdam, Netherlands : 1999)*, *75*, 453-471. https://doi.org/10.1016/j.comnet.2014.10.015

Iranmanesh, A., & Reza Naji, H. (2021). A protocol for cluster confirmations of SDN controllers against DDoS attacks. *Computers & electrical engineering*, *93*, 107265. https://doi.org/10.1016/j.compeleceng.2021.107265

Jammal, M., Singh, T., Shami, A., Asal, R., & Li, Y. (2014). Software defined networking: State of the art and research challenges. *Computer networks (Amsterdam, Netherlands : 1999)*, *72*, 74-98. https://doi.org/10.1016/j.comnet.2014.07.004

Karakus, M., & Durresi, A. (2017a). Quality of Service (QoS) in Software Defined Networking (SDN): A survey. *Journal of network and computer applications*, *80*, 200-218. https://doi.org/10.1016/j.jnca.2016.12.019

Karakus, M., & Durresi, A. (2017b). A survey: Control plane scalability issues and approaches in Software-Defined Networking (SDN). *Computer networks (Amsterdam, Netherlands : 1999)*, *112*, 279-293. https://doi.org/10.1016/j.comnet.2016.11.017

Karakus, M., & Durresi, A. (2018). Economic Viability of Software Defined Networking (SDN). *Computer networks (Amsterdam, Netherlands : 1999)*, *135*, 81-95. https://doi.org/10.1016/j.comnet.2018.02.015

Macedo, D. F., Guedes, D., Vieira, L. F. M., Vieira, M. A. M., & Nogueira, M. (2015). Programmable Networks-From Software-Defined Radio to Software-Defined Networking. *IEEE Communications surveys and tutorials*, *17*(2), 1102-1125. https://doi.org/10.1109/COMST.2015.2402617

Moin, S., Karim, A., Safdar, K., Iqbal, I., Safdar, Z., Vijayakumar, V., Ahmed, K. T., & Abid, S. A. (2020). GREEN SDN — An enhanced paradigm of SDN: Review, taxonomy, and future directions. *Concurrency and computation*, *32*(21), n/a. https://doi.org/10.1002/cpe.5086

Muelas, D., Ramos, J., & Vergara, J. E. L. d. (2018). Assessing the Limits of Mininet-Based Environments for Network Experimentation. *IEEE Network*, *32*(6), 168-176. https://doi.org/10.1109/MNET.2018.1700277

Ren, F., Huang, X., Liu, F., & Lin, C. (2008). Improving TCP Throughput over HSDPA Networks. *IEEE transactions on wireless communications*, *7*(6), 1993-1998. https://doi.org/10.1109/TWC.2008.061007

Wang, M.-H., Chen, L.-W., Chi, P.-W., & Lei, C.-L. (2017). SDUDP: A Reliable UDP-Based Transmission Protocol Over SDN. *IEEE access*, *5*, 5904-5916. https://doi.org/10.1109/ACCESS.2017.2693376

Wu, D., Nie, X., Asmare, E., Arkhipov, D. I., Qin, Z., Li, R., McCann, J. A., & Li, K. (2020). Towards Distributed SDN: Mobility Management and Flow Scheduling in Software Defined Urban IoT. *IEEE transactions on parallel and distributed systems*, *31*(6), 1400-1418. https://doi.org/10.1109/TPDS.2018.2883438

Appendix

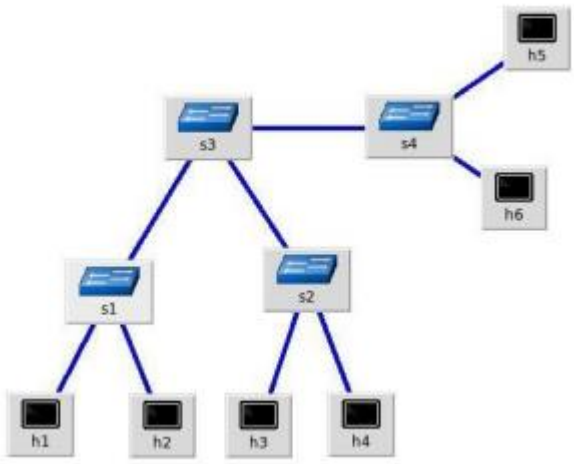


Figure 9 - Conventional Topology

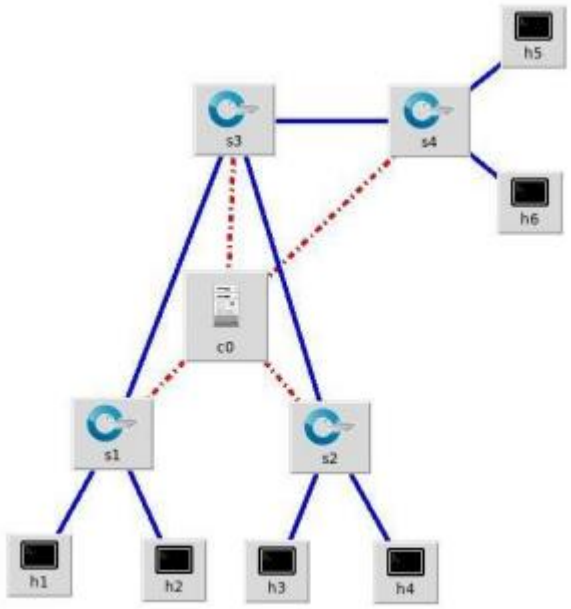


Figure 10 - SDN Topology

TCP	TCP Average Bandwith	TCP Average Transfer Per Interval		UDP	UDP Average Transfer Per Interval	UDP Average Bandwith Per Interval	UDP Average Jitter Per Interval		Packets Out Of Order
	23.88	2.894			1.254	10.52	0.002		0.6
	22.92	2.67			1.25	10.5	0.0024		0
	22.9	2.446			1.25	10.5	0.0066		0
	22.3	2.596			1.25	10.5	0.0054		0
	24.68	2.874			1.25	10.5	0.0042		0
	24.7	2.868			1.25	10.5	0.0036		0
	24.76	2.882			1.25	10.5	0.002		0
	25.62	2.984			1.25	10.5	0.0024		0
	25.16	2.932			1.25	10.5	0.0026		0
	24.82	2.888			1.25	10.5	0.0024		0
	25	2.91			1.25	10.5	0.0018		0
	25.68	2.99			1.25	10.5	0.0024		0
	24.9	2.898			1.25	10.5	0.0058		0
	25.18	2.932			1.25	10.5	0.002		0
Lowest	22.3	2.446			1.25	10.5	0.0018		
Highest	25.68	2.99			1.254	10.52	0.0066		0.003145455
Differenc	3.38	0.544			0.004	0.02	0.0048		

Figure 11 - Conventional Data

TCP	TCP Average Bandv	TCP Average Transfer Per Interval		UDP	UDP Average Transfer Per Inte	UDP Average Bandwith Per Inte	UDP Average Jitter Per Interval	Packets O
	18.894	2.2054			1.1822	9.924	78.8812	45.4
	21	2.446			1.376	11.54	1.203	89.4
	20.68	2.406			1.25	10.5	2.403	0.6
	18.1	2.108			1.25	10.5	0.004	0.6
	19.56	2.274			1.254	10.54	0.0024	2.8
	19.06	2.22			1.256	10.54	0.0016	4
	21.9	2.55			1.25	10.5	0.0024	0
	21.7	2.554			1.25	10.5	0.002	0
	21.66	2.518			1.25	10.5	0.0014	0
	21.92	2.536			1.25	10.5	0.0014	0
	21	2.444			1.25	10.5	0.00125	0
	22.06	2.568			1.25	10.5	0.0015	0
	23.06	2.686			1.25	10.5	0.0025	0
	22.44	2.612			1.25	10.5	0.002	0
Lowest	18.1	2.108			1.1822	9.924	0.00125	0
Highest	23.06	2.686			1.376	11.54	78.8812	0.002041
Differenc	4.96	0.578			0.1938	1.616	78.87995	0.002041

Figure 12 - SDN Data

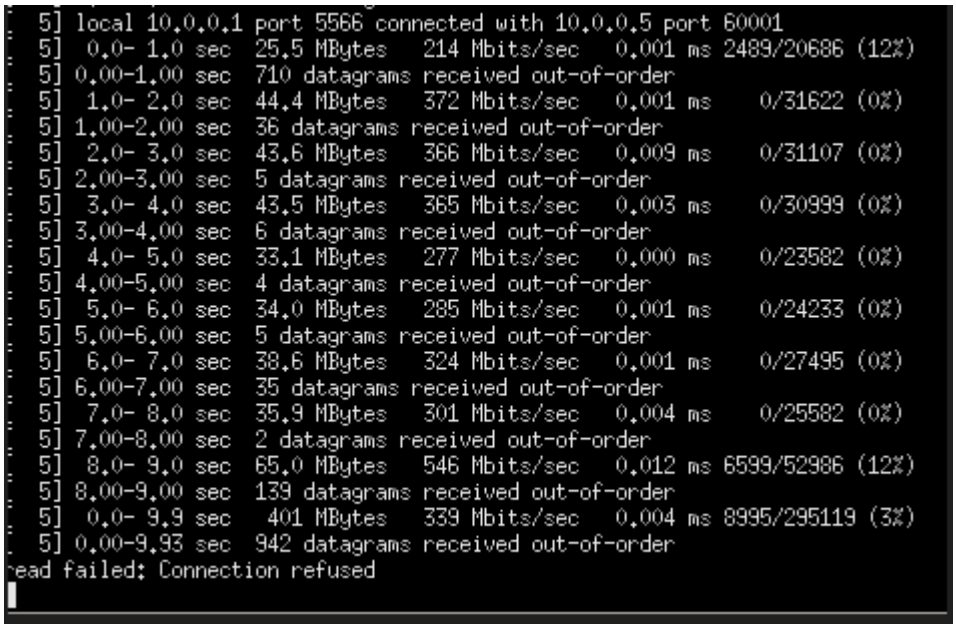


Figure 13 SDN 1000Mb/s