Resiliency Analysis of ONOS and Opendaylight SDN Controllers Against Switch and Link Failures

Mehmet Dagli
NETAS
Istanbul, TURKEY
mdagli@netas.com.tr

Selcuk Keskin
NETAS
Istanbul, TURKEY
selcukk@netas.com.tr

Yagmur Yigit
NETAS
Istanbul, TURKEY
yagmury@netas.com.tr

Adem Kose
NETAS
Istanbul, TURKEY
akose@netas.com.tr

***Abstract*—Video conference applications require a high quality of service (QoS) from the point of latency, jitter, and bandwidth. As the QoS parameters seriously suffer from instant congestions and network failures, resiliency mechanisms utilized at network level have a direct impact on the experienced video quality. In addition to the solutions offered under traditional networking, software defined networking (SDN) paradigm also offers various solutions to keep quality of experience (QoE) at certain levels even in instant network failure situations. Different types of SDN controllers provide different resiliency mechanisms to recover from failures. Recovery time is an important parameter to keep video conference experience at satisfactory levels as longer recovery time causes frozen streams, streaming delays and even drops in video conferencing. This study gives a comparative analysis of the recovery time on ONOS and Opendaylight SDN controllers in case of the link and switch failures.**

***Index Terms*—Software Defined Networking (SDN), Controller, OpenDayLight, ONOS, Resiliency**

I. Introduction

The network devices are made up of the control plane (which also includes a management plane) and the data plane in traditional switching. These are established in advance with the fixed capabilities and functions in the firmware of routers and switches. Orchestration of switching and signaling is enabled by the control plane since the data plane forwards traffic according to the routing table provided by the control plane. The traditional switches are dependent on the abilities of the software sent with hardware which is very costly in case of improvement. To overcome the limitations of new networking development, the approach of Software Defined Networks (SDN) has been proposed.

In the search of supporting a more dynamic and scalable networking environment, SDN has come into the open as a groundbreaking expedience. In addition, the division of the planes (data and control) has provided many advantages, such as application and service flexibility, improving user experience and reducing complexity. The concept of programmability and decoupling in SDN are the main reasons for the growing momentum of this technology [1]. Thanks to the SDN controllers, the network administration tasks are much easier in the world where a new device is added the Internet with different technologies every day [2].

978-1-7281-8818-8/20/$31.00 ©2020 IEEE

Bypassing any requirement to specific control of separate devices, the control plane can be supervised by a centric SDN controller. The central controller enables network control plane direct programming, which forms a nimble network that is centrally controlled. It has information on all available paths and a precise aspect of the network. Therefore, the controller can compute the routes using various factors, such as the class of traffic, source, and the target. The applications can be developed allowing fast and potent optimization of network resources that are manipulable by the specific needs since SDN is not dependent on any private software. Although SDN is a popular research topic in different networking fields, most researchers place reliance on a network emulator called Mininet to evaluate a novel approach or algorithm working on SDN.

Resiliency is an important point of concern in SDN. There are some threats that directly affect network performance. For instance, link and node failures that impact the overall performance of applications running on SDN, and the cyberattack to the SDN controller are some of the important threats to network performance. There are many aspects of recovery of the network against such threats/failures and the type of SDN controller is one of them. In this study, we compare and analyze the performance of different SDN controllers -ONOS and Opendaylight- against the link and switch failures for a video conference scenario.

II. State of the Art

There are many previous studies in the literature comparing and evaluating general SDN performance. [3], [4], [5], and [6] provided a comparative performance investigation and evaluation of SDN controllers. Results [3] show the SDN network model based on the cloud can scale up with more tangled topology and enterprise-scale load. It also provides more reliable performance compared to the controller of local hosted. Throughput tests presented NOX better than the default version of FloodLight and POX results proved to be commonly unsuitable for the production environment, however, this controller may be best suited for experimentation. [5] result found that Floodlight outperformed OpenDaylight where less memory and CPU is utilized. Nevertheless, OpenDaylight outperformed Floodlight in average setup latency but it is not stable when interoperating OFNet. [6] test results showed that in terms of latency and throughput, multi-threaded controllers (Floodlight, OpenMul, Beacon, Maestro, OpenDaylight, and ONOS) performed significantly better than single-threaded controllers like NOX, POX, and Ryu.They also stressed that simulation or emulation-based evaluation can only give one performance indicator at best and may differ significantly from the actual production environment evaluation.

On the other hand, some of the articles have focused on evaluating online video services over the SDN network. [7] proposed a system architecture and an emulation environment based on Mininet, for the purpose of using multi-domain SDN with end-to-end QoS to stream multimedia. They compared their system with IntServ which is of the most known service quality models. According to the results, the proposed system is more efficient and scalable than IntServ in terms of message traffic and guaranteed end to end QoS. [8] proposed a multi-media path selection algorithm for the SDN traffic engineering module. This algorithm works in two options: declined traffic mode and accepted traffic mode. Test results confirm that the proposed algorithm in the declined traffic mode has a higher amount of accepted multimedia requests compared to the rest of the algorithms, and the algorithm in the accepted traffic mode has a lower amount of reduced bandwidth than others.

Resiliency mechanism in SDN impacts the quality-of-experience (QoE) in video applications. Switch and link failures negatively impact QoE and recovery time from the failure is important for uninterrupted video service delivery. [9] proposed a new framework to ensure satisfactory QoE for VOIP and video applications through quick recovery mechanisms in network failure cases. Results showed less packet loss and higher MOS scores in the event of a link failure with the proposed framework. [10] proposed protection strategies against different types of link and node failures in a service chain. Results for online video gaming showed deployment of 107 percent additional virtual network function (VNF) nodes to ensure resiliency against single link and node failures. [11] proposed a high fidelity SDN network simulator developed by authors. The proposed solution analyzes resilience to the impact of controller failure using snapshots of actual switch configurations. It also proposed a backup paths algorithm. They simulated the proposed backup path and guaranteed it is not to cause loops.

This paper will evaluate the performance of SDN controllers - ONOS and Opendaylight - in more specific and lifelike video conference scenarios against link and switch failures.

III. SDN Controllers and Resiliency

Stanford University has developed SDN paradigm with OpenFlow protocol to ensure an architecture enabling for the manage of the network helping a program and it gives a novel perspective to computer networks with a series of items added with virtualization. Flow-based decision are made during the packet transmission by the devices which are transfer elements in SDN architecture. A set of actions creates a criteria for package field values to define the flows [12].

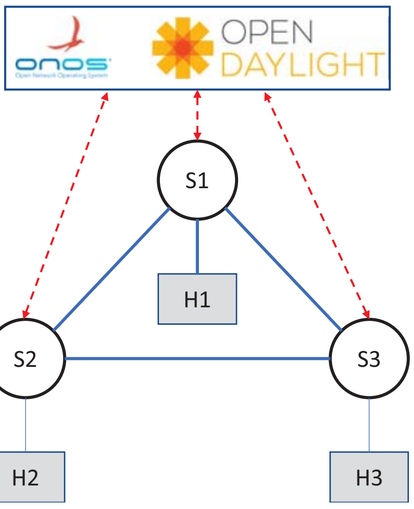

Fig. 1. Simplified SDN architecture

One of the first SDN standards is considered as the OpenFlow specification. It has been started by a society from Stanford University, then, it proceeded under control of the Open Network Foundation (ONF) which is a non-profit operator led consortium [13]. To advocate for the adoption, marketing and distribution of solutions that use open-source software, white box economies and decoupled networks to maximum advantage to change the network industry radically, the organization aims to create an ecosystem. To enable the communication from SDN Controllers to OpenFlow switches and vice versa, the ONF describes the specifications.

In the SDN/OpenFlow framework, a set of packets from a source to a destination defines a flow. All packages' flow of all transfer elements uses the same service protocols. Flow provides the combination of different network devices such as firewalls, switches, routers and intermediate boxes to make decisions about packets. Furthermore,the flow programming gives remarkable flexibility reached through just the flow tables used. The logic of control is isolated from network devices and it can be moved to an external resource. A simplified SDN architecture is depicted in the Figure 1.

A. *Opendaylight*

One of the breakthroughs of ONF (Open Network Foundation) is Opendaylight (ODL) which was announced in 2013. ODL is an open source project with a Java-based controller which reinforces high availability and clustering. It also eliminated numerous flaws found in previous controllers. The project was supported and implemented by many vendors to make a trustworthy and effective controller for large scale networks. It made a huge impact on the commercial SDN industry in terms of a large number of SDN controllers based on ODL. The Raft algorithm has been used in ODL project to send all data changes to the head which is chosen a controller to process the update [14].

B. ONOS

The other SDN controller platform by Linux Foundation is Open Networking Operating System (ONOS). ONOS gives assistance to many distributed SDN control planes providing the maximum availability, high performance and scale-out [15], [16]. It is also used in many ways by service providers, enterprises, research institutions and other organizations around the world. Huawei Agile Controller has been developed based on ONOS and a business prepared SDN item has been built up with utilization of ONOS by Samsung, as well as business level SDN arrangements has been created privately with different administrators. Furthermore, ECI works on an open-source SDN controller and Ciena also works on a financially solidified rendition of ONOS controller programming which aims to help telecommunications companies transform their focus businesses into various distributed data centers.

C. Resiliency

Well defined failure management strategies in SDN contribute to uninterrupted service delivery for online video services. Failures occur in data plane and control plane which main topics need to be taken into consideration in the design of a resilient SDN. [17] discussed data plane resiliency in details and proposed a fault-tolerant multi-controller approach. On the other hand, SDN controller's network discovery after a link or switch failure presents another important problem in the data plane. There are different link failure detection and recovery mechanisms utilized in different SDN controllers as evaluated in [18]. ONOS and Opendaylight SDN controllers also utilize different link and switch failure detection and recovery mechanisms which makes them react against failures in different fashions. As network failures cause serious impacts on online video services like frozen streams, video delay or even call drops in real time services like video conferencing, the amount of time SDN controllers react against failures directly impact QoE of video services. Since ONOS and Opendaylight controllers are among the most popular SDN controllers, their reaction against link and switch failures during the delivery of video conferencing service in different levels of network complexities require an investigation which is the main objective of this study.

IV. Experimental Results

In this section, we provide detailed performance analysis of Opendaylight and ONOS. We run our tests on a virtual machine (VM) was configured with 12GB memory and 4 virtual cores, each running 2.2 GHz. Operating system was Ubuntu 16.04 LTS-64 bit. As SDN controller, we considered Opendaylight Oxygen (version 0.8.4) and ONOS Uguisu (version 2.4.0). As data plane, we used Mininet version 2.2.2 and Open vSwitch 2.5.5 supporting OpenFlow 1.3 to model an *NxN* Torus topology.

We conduct an experiment that uses Mininet to emulate different networks with different number of OpenFlow-enabled switches on the Torus topology. A Torus 4x4 topology can be seen in Figure 2. In the topology, there are 16 switches

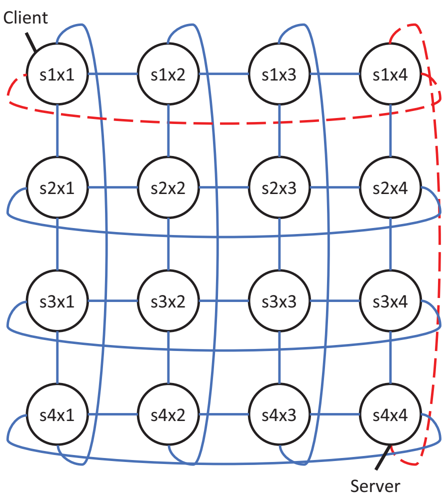

Fig. 2. Network topology on Mininet environment

and each one is connected with other 4 switches and one host machine. First, the controller is activated on the local or any remote machine. Then, the below command, which gets the IP address of the controller as a parameter, is executed to generate the torus network topology that is monitored and controlled by the controller: *sudo mn --topo torus,4,4 --controller remote,ip=127.0.0.1, port=6633 --switch ovsk,protocols=OpenFlow13*

We have used iperf3 which is a software to generate dummy traffic to measure TCP and UDP bandwidth performance of the network. During the traffic flow, the software provides measurement according to bandwidth, jitter, delay, etc. [19].s

Two different experiments were conducted to compare resilience mechanism of SDN controllers. The first is to stop the connection between the two switches where the flow occurs while the communication between the two users continues. Let's assume that first user is the person in a video conference that uses the host *H1x1* in the Torus topology and the second one is the server of the video conference that uses the host *H4x4*. The flow from the user to the server is shown with long dashed lines in the Figure 2. The host *H1x1* is connected to the switch *S1x1* in the Torus topology, so the packages are directly sent to the switch *S1x1* from the user. Then the packages are forwarded to the switches *S1x4* and *S4x4*, respectively. Finally, the server fetches the packages from *S4x4* which is directly connected the host *H4x4*. The path is assigned by the SDN controller at the initialization time of the topology. The link between between *S1x4* and *S4x4* is stopped by a Mininet command in the first experiment. When the connection stops, the SDN controller assigns a new path for the flow. During the re-assignment, some packages of the communication do not reach their desired destination.

The second experiment is to stop the switch on the path of the communication. While the communication continues between the user and the server, the switch *S1x4* is stopped by a Mininet command. In this situation, the SDN controller

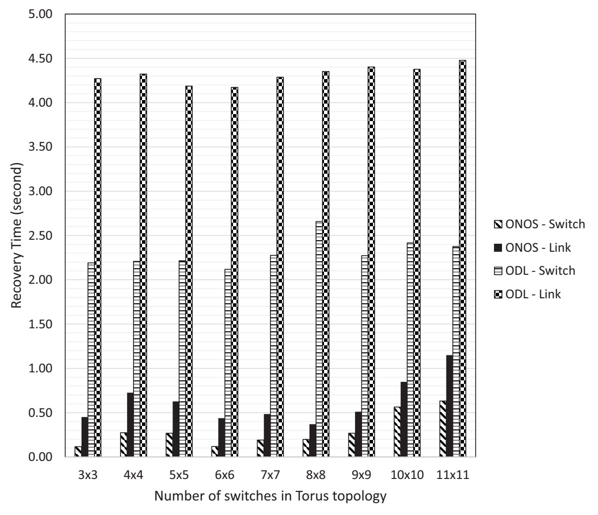

Fig. 3. Recovery time comparison for link and switch failure

builds a new end-to-end path after detecting the switch failure.

In order to know how fast is the SDN controllers to re-route the path, we execute the *iperf3* command on the network, from host *H1x1* to host *H4x4*, and use that value as reference:

On the server host: *iperf3 -s -p 9999*
On the user host: *iperf3 -c 10.0.0.16 -p 9999 -u -t 30*

To make a good comparison, each measurement lasts 30 seconds and each scenario is repeated 10 times and computed the average. First, in each measurement, we stopped the link between *S1x4* and *S4x4* after 15 seconds and monitored the traffic on the server host *H4x4*. During the failure, ONOS controller immediately forwards the traffic over another alternative path. However, ODL controller takes much longer to recover the links to the destination host. In the second experiment, we stopped the switch *S4x4* after 15 seconds. The both SDN controllers show the similar behaviour that is ONOS reacts faster than ODL. Figure 3 illustrates the performance results of the SDN controllers in consideration of both link and switch failures under varying number of switches. According to result obtained, ONOS is on average 10 times faster than ODL in both failure types. On the other hand, the results show that SDN controllers are able to build a new path in case of switch failures faster than link failures.

Furthermore, we compared the resiliency mechanisms of SDN controllers for the video communications platform. VIO [20], developed by Turkish Telecommunications company Netas, was utilized during the test as multi-point video communications platform. VIO server works as selective forwarding unit (SFU) where only selected video streams are forwarded to participant clients. VIO client running on web browsers utilizes web based real time communication (WebRTC) technology to connect other clients through conference server. VIO server instance installed into mininet environment configured as one of the hosts(*H1x1*) during the tests. WebRTC clients running on supported browsers initialized from other hosts in mininet environment(*H4x4* and *H4x3*). When we performed the same experiments in this scenario, the users experienced frozen frames for a while because of packet loss until forwarding packets to the new path.

V. Conclusion and Future Work

A. Conclusion

In this paper, the open source SDN controllers ONOS and ODL has been compared in terms of their resiliency mechanism under link and switch failures. The ONOS controller beats out the ODL controller in terms of recovering a new path to avoid degrading service quality. The results show that in the event of a possible failure, SDN controllers need to reduce the recovery time as much as possible so that the quality of the experience of users participating in the video conference does not decrease.

B. Future Work

This study covered link and switch failures to compare the resiliency behaviour of ONOS and Opendaylight SDN controllers, future works may add other SDN controllers (POX, Ryu, etc...) in the comparison list to have a wider view. Moreover future works may also take SDN controller failure scenarios into consideration in addition to link and switch failures.

Acknowledgment

This work has been performed in the frame work of European Celtic-NEXT RELIANCE project (C2017/3-8) and funded by the Scientific and Technological Research council of Turkey TUBITAK.

References

[1] B. A. A. Nunes, M. Mendonca, X. Nguyen, K. Obraczka, and T. Turletti, "A survey of software-defined networking: Past, present, and future of programmable networks," *IEEE Communications Surveys Tutorials*, vol. 16, no. 3, pp. 1617–1634, 2014.

[2] S. Asadollahi, B. Goswami, A. S. Raoufy, and H. G. J. Domingos, "Scalability of software defined network on floodlight controller using OFNet," in *2017 International Conference on Electrical, Electronics, Communication, Computer, and Optimization Techniques (ICEECCOT)*, Dec. 2017, pp. 1–5.

[3] K. Basu, M. Younas, A. W. Wan Tow, and F. Ball, "Performance comparison of a sdn network between cloud-based and locally hosted sdn controllers," in *2018 IEEE Fourth International Conference on Big Data Computing Service and Applications (BigDataService)*, 2018, pp. 49–55.

[4] A. K. Arahunashi, S. Neethu, and H. V. Ravish Aradhya, "Performance analysis of various sdn controllers in mininet emulator," in *2019 4th International Conference on Recent Trends on Electronics, Information, Communication Technology (RTEICT)*, 2019, pp. 752–756.

[5] A. H. M. Hassan, A. M. Alhassan, and F. Izzeldean, "Performance evaluation of sdn controllers in ofnet emulation environment," in *2019 International Conference on Computer, Control, Electrical, and Electronics Engineering (ICCCEEE)*, 2019, pp. 1–6.

[6] L. Zhu, M. M. Karim, K. Sharif, F. Li, X. Du, and M. Guizani, "Sdn controllers: Benchmarking & performance evaluation," *ArXiv*, vol. abs/1902.04491, 2019.

[7] S. Yilmaz, K. T. Bagci, and A. M. Tekalp, "System architecture and emulator for multimedia streaming over multi-domain sdn," in *2015 23nd Signal Processing and Communications Applications Conference (SIU)*, 2015, pp. 715–718.

[8] A. Chooprateep and Y. Somchit, “Video path selection for traffic engineering in sdn,” in *2019 11th International Conference on Information Technology and Electrical Engineering (ICITEE)*, 2019, pp. 1–6.

[9] M.-E. Xezonaki, E. Liotou, N. Passas, and L. Merakos, “An SDN QoE Monitoring Framework for VoIP and Video Applications,” in *2018 IEEE 19th International Symposium on "A World of Wireless, Mobile and Multimedia Networks" (WoWMoM)*, June 2018, pp. 1–6.

[10] A. Hmaity, M. Savi, F. Musumeci, M. Tornatore, and A. Pattavina, “Virtual network function placement for resilient service chain provisioning,” in *2016 8th International Workshop on Resilient Networks Design and Modeling (RNDM)*, 2016, pp. 245–252.

[11] D. M. Nicol and R. Kumar, “Sdn resiliency to controller failure in mobile contexts,” in *2019 Winter Simulation Conference (WSC)*, 2019, pp. 2831–2842.

[12] D. Kreutz, F. M. V. Ramos, P. E. Veríssimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, “Software-defined networking: A comprehensive survey,” *Proceedings of the IEEE*, vol. 103, no. 1, pp. 14–76, 2015.

[13] J. Tourrilhes, P. Sharma, S. Banerjee, and J. Pettit, “Sdn and openflow evolution: A standards perspective,” *Computer*, vol. 47, no. 11, pp. 22–29, 2014.

[14] T. Kim, S. Choi, J. Myung, and C. Lim, “Load balancing on distributed datastore in opendaylight sdn controller cluster,” in *2017 IEEE Conference on Network Softwarization (NetSoft)*, 2017, pp. 1–3.

[15] Y. Zhang, L. Cui, W. Wang, and Y. Zhang, “A survey on software defined networking with multiple controllers,” *Journal of Network and Computer Applications*, vol. 103, no. C, p. 101–118, 2018.

[16] P. Berde, M. Gerola, J. Hart, Y. Higuchi, M. Kobayashi, T. Koide, B. Lantz, B. O’Connor, P. Radoslavov, W. Snow, and G. Parulkar, “ONOS: Towards an Open, Distributed SDN OS,” in *Proceedings of the Third Workshop on Hot Topics in Software Defined Networking*, 2014, p. 1–6.

[17] E. S. Spalla, D. R. Mafioletti, A. B. Liberato, G. Ewald, C. E. Rothenberg, L. Camargos, R. S. Villaca, and M. Martinello, “Ar2c2: Actively replicated controllers for sdn resilient control plane,” in *NOMS 2016 - 2016 IEEE/IFIP Network Operations and Management Symposium*, 2016, pp. 189–196.

[18] Ali, Jehad and Lee, Gyu-min and Roh, Byeong-hee and Ryu, Dong Kuk and Park, Gyudong, “Software-Defined Networking Approaches for Link Failure Recovery: A Survey,” *Sustainability*, vol. 12, no. 10, pp. 1–28, 2020.

[19] iperf3, tool for active measurements of the maximum achievable bandwidth on ip networks. [Accessed 14 October 2020]. [Online]. Available: https://github.com/esnet/iperf

[20] Netas VIO Video Communications Platform. [Online]. Available: http://www.netas.com.tr/en/innovation-productization/vio-video-communication-platfrom/