Towards a Software Defined Multi-Domain Architecture for the Internet of Things

Leonel Piscalho Junior
Lisboa, Portugal
email: leopiscalho@gmail.com

José Moura
Instituto de Telecomunicações
ISCTE - Instituto Universitário de Lisboa
Lisboa, Portugal
email: jose.moura@iscte-iul.pt

Rui Neto Marinheiro
Instituto de Telecomunicações
ISCTE - Instituto Universitário de Lisboa
Lisboa, Portugal
email: rui.marinheiro@iscte-iul.pt

Abstract — **The emerging communication networks tend to aggregate heterogeneous networking infrastructures as well as data flows with very distinct requisites. This implies that the complete satisfaction of Quality of Service (QoS) metrics is very difficult to achieve, using the legacy management solutions. Alternatively, the Software Defined Networking (SDN) paradigm offers a logical centralized management of the necessary network resources for data flows, namely the ones originated in sensor devices. Therefore, this work investigates a solution that meets the QoS requirements of traffic from remote Internet of Thing (IoT) devices. To achieve this goal, we have designed a SDN-based solution that manages a network topology formed by several domains. We assume each network domain is controlled by its own SDN controller. In addition, our solution assumes that the several SDN controllers need to be orchestrated among them to maximize the management efficiency of the available end-to-end network resources. This orchestration is done via an SDN transit domain ruled by the ONOS SDN-IP application. We have emulated network topologies with IoT devices to evaluate the proposed solution in terms of its functionality, robustness against network failures, and QoS support. Analyzing the obtained results, our solution can support a cross-controller SDN domain communication. It is also capable of reacting automatically to topology failures. In addition, it can prioritize the traffic within the network infrastructure, providing to the end users strong guarantees on the desired quality for the exchange of data associated to the applications they aim to use.**

Keywords-Multi-domain; SDN; IoT; QoS

I. INTRODUCTION

The exponential data traffic growth and the network heterogeneity are challenging the legacy networks. This occurs due to the high-level of complexity to interconnect several services and smart devices, both related to the emerging paradigm of IoT. They exchange real-time information through the networking infrastructure to be processed by intelligent applications. This implies not only various types of traffic, but also the ability to offer QoS guarantees across the network [1]. With the advent of SDN, it offers new ways to design more flexible networks.

SDN stands out for its flexibility, programmability and centralized logical management, which separates the data layer from the control layer, allowing the passage of logical operations from the data plane devices to a centralized software controller, which operates over those devices [2].

Due to the size, heterogeneity, and complexity of current networks, approaches based on multiple domains are very scalable. This domain multiplicity consists in the network division in different administrative domains, each managing its network subset and optimizing both the domain performance and the fulfilment of QoS requisites.

Previous research [3] tries to improve the IP domain routing management and provide end-to-end QoS paths [4]. Nevertheless, the available work is mostly based on a centralized controller approach that handles routing within a single administrative domain, offering very limited results. In this way, the SDN configuration of inter-domain scenarios is very pertinent. The orchestration among all the SDN controllers is also vital to ensure reliable end-to-end services, such as routing, and QoS deployment.

The interaction between the different SDN domains depends on an inter domain routing protocol, and BGP is a very popular protocol for this. ONOS [5] and ODL [6] are SDN controllers that support distributed scenarios. They are also most commonly used in wide area networks (WANs). Nevertheless, these two SDN controllers have slight performance differences as shown in [10], where ONOS seems to be a better choice for our current WAN scenario.

The authors of [7] suggest a solution designated by Inter Cluster ONOS Network application (ICONA). This solution manages a large networking scenario under the same administrative domain (i.e., GEANT) with geographically distributed controllers. Another contribution [8] proposes a gradual implementation of SDN-based solutions over different administrative domains that interoperates with other non-SDN based domains. They study a peering application among distinct Autonomous Systems (ASs) called SDN-IP, which runs at the top of the ONOS SDN controller.

Due to the low number of literature contributions supporting end-to-end QoS in IoT networks with scarce resources, the SDN-IP application is very important to achieve our goal for ensuring QoS support in distributed systems with multiple SDN controllers. Therefore, the research question that motivated our work is "How to Provide the necessary resources to meet QoS and robustness requirements for traffic from heterogeneous IoT devices in a distributed system with multiple SDN controllers?".

The main contributions of the this paper are the deployment of a SDN solution that manages resources from ASs to meet QoS and robustness requirements for routing heterogeneous traffic across those ASs. The routed traffic is from heterogeneous devices, including IoT ones, located at the network edge.

The remaining part of the current paper is following described. Section II presents the literature review in the related research areas. Section III discusses the design of the proposed solution. Section IV is about the deployment of the proposed contribution. Section V discusses the performed tests and their results. Finally, Section VI presents some general conclusions about the current contribution and some promising future work.

II. LITERATURE REVIEW

This section briefly revises the literature in the next topics: SDN architecture, inter-domain communication, and IoT.

A. SDN Architecture

The SDN is a new emerging network paradigm to simplify networking management, where the data and control layers are separated. In addition, SDN enables the programming of the network operation [2]. This programming can be made with distinct levels of hardware abstraction. In this way, the SDN controller can program the network devices at the data plane but the former needs to know in advance some specifics from the latter ones, such as the number and characteristics of ports at each device. In a distinct way, a networking application at the top-most layer of the SDN architecture, as shown in Fig. 1, can program the network topology without knowing any detail about the network data plane.

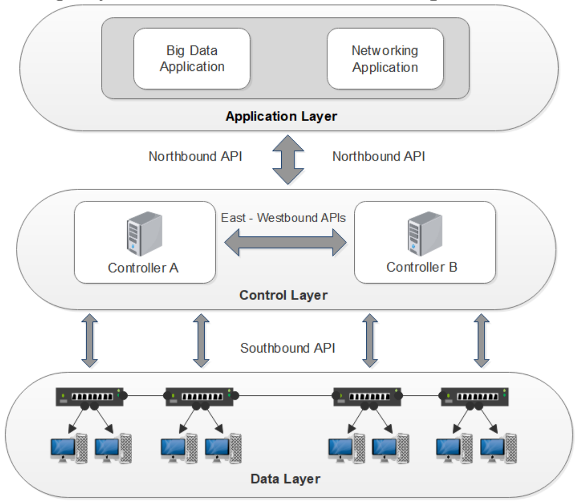


Figure 1. SDN architecture.

Fig. 1 shows an SDN architecture with three layers. The bottom layer is the data layer, which consists of compatible SDN devices, like routers and switches. The intermediate layer is the control layer. It is formed by the controllers that have the global vision of the network. The control layer communicates with the devices at the data layer through a Southbound protocol (e.g., OpenFlow). The application layer is the top-most layer. It communicates with controllers via Northbound APIs (e.g., Restful); this layer has several running applications that deploy many relevant management services.

Separating these layers, there are two vertical communication channels to connect each pair with Northbound/Southbound APIs, as well as East/Westbound APIs to provide horizontal communication between controllers, aiming the federation between domains.

B. Inter-Domain Communication

Initially, SDN was based on a single controller's approach to manage an entire network. Despite its simplicity in terms of both development and operation, it faces some limitations when deployed in large networks, regarding reliability and scalability. An SDN design with a single controller can become unreliable due to the issue of a single point of failure. Moreover, a single SDN controller can become overwhelmed when working with multiple simultaneous requests from the data plane [9]. Alternatively, a multiple controllers approach, provides solutions to mitigate the problems just discussed, such as the single point of failure, and low scalability [2][9]. The authors of [9] discusses some challenges imposed to SDN-based solutions with multiple controllers for managing large networks, such as, complexity, scalability, consistency, reliability and load balancing.

There are several distributed architectures formed by multiple SDN controllers namely horizontal or hierarchical [2]. They also discuss several methods to establish communications among SDN controllers. In [1], a comparative study of the most currently used SDN controllers is presented. From these, we highlight ODL and ONOS. Both support a fully distributed architecture and an SDN implementation across diverse networking domains [2]. Although these two options are similar, there are some differences [10], which justifies ONOS as a more suitable controller than ODL to explore the full potential of SDN in carrier-grade scenarios, as the one of our paper.

A multi-domain SDN architecture refers to a set of different administrative SDN domains or ASs that exchange information regarding network status, configuration, or other relevant network services, such as packet routing to a destination. In addition, Border Gateway Protocol (BGP) [11] is the most commonly used protocol to provide the end-to-end IP routing services over administrative domains. Then, each SDN controller needs to process an external learned BGP route to a destination prefix and translate it to local routing rules, which are only valid within the network domain the controller is responsible for. It is expected that summing up the individual routing contributions from the diverse SDN controllers results in a final aggregated outcome that fulfils the end-to-end BGP route.

C. IoT Overview

An Internet of Things (IoT) domain is a network of physical devices and sensors with embedded technology that interacts with the local environment. The IoT network not only collects data but it also exchanges the data to some servers located at remote clouds or even to some fog servers located at the network periphery. There are many IoT scenarios, such as health, home automation, smart transportation, environmental monitoring, or smart grids.

Recent work [12] has highlighted the relevance of SDN-based systems for controlling network domains formed by IoT devices and surveyed previous related contributions. However, SDN solutions for wireless networks and, more specifically, in wireless sensor (and actuators) networks do not abound [13]. Delivering end-to-end service orchestration chains, across multiple SDN domains, for an IoT infrastructure deployment, including data collection at the cloud, edge processing, and publishing services with quality differentiation is still at its infancy [14].

The present paper provides some novel contributions regarding the line of research discussed in the current section.

III. PROPOSAL DESIGN

As previously mentioned, the main goal of the current paper is to investigate a solution that meets the QoS requirements of data traffic originated at IoT devices in a heterogeneous network with multiple domains ruled by SDN controllers. We next discuss the design of our proposal.

Fig. 2 shows the design of a network topology formed by multiple administrative domains.

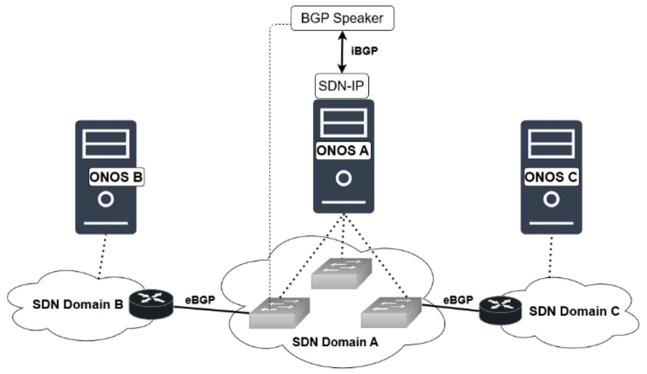


Figure 2. System design.

Each administrative domain is controlled by an SDN controller located at the intermediate level of the proposed architecture. In this way, SDN domain A works as a transit AS, which interconnects different externals SDN domains (B and C) that interface with the domain A, through BGP routers.

The data plane is formed by switches, BGP routers, and end host devices. At the application plane, there are BGP speakers that behave like BGP route reflectors, learning from the BGP routers IP destination prefixes and passing them to the SDN-IP application. Then, this application interacts with the SDN controller. From the previous interaction, the BGP learned paths are mapped to compatible data flow rules, which are transferred via Southbound protocol to the switches.

At the top layer are running applications that define how the network operates. In the transit domain A, the SDN-IP application allows, as already explained, the routing of packets among BGP ASs. The previous routing implies the forwarding of packets among the diverse switches belonging to the SDN Domain A. In addition, some auxiliary applications in the SDN controller of Domain A are also required (e.g., Configs and ProxyARP).

One of the most important QoS concept is that the traffic should not be treated equally, e.g., we need to prioritize the usage of communication link resources. Therefore, in our proposal we also prioritize the traffic in a network that is a mixture of IoT and legacy flows. The traffic prioritization is based on creating distinct virtual output queues offered at the data plane switches. In addition, some flow rules are installed in the data plane switches. These flow rules allow traffic to be served by different queues according to the traffic priority. In our work, we assume that the traffic priority is unrelated with the priority field normally used in OpenFlow flow rules. An interesting future prospect could be to use the OpenFlow priority field for controlling the traffic quality.

IV. PROPOSAL DEPLOYMENT

This section discusses the testbed topology and the deployment of our proposal to manage that topology. It aims to satisfy QoS requirements in the presence of heterogeneous flows, some originated from remote IoT devices. The network infrastructure is formed by several administrative domains.

A. Multi-Domain Topology

Table I lists all software and tools used to deploy and validate our proposal.

TABLE I. SOFTWARE USED IN THE DEPLOYMENT

Category	Software / Technology
Northbound Application	SDN-IP
SDN Controller	ONOS 1.15.0
Software Switch	OpenvSwitch 2.9.2
Southbound Communication	OpenFlow
Interdomain Protocol	BGP
Network Emulator	Mininet
BGP Software	Quagga
Traffic Analyser	Wireshark, Tcpdump
Virtual Hypervisor	Oracle Virtual Box
VM Operating System	Ubuntu 16.04
Traffic Generator and Measurement	Iperf
Video transmitter Application	VLC

Firstly, the general idea is to deploy a scenario that provides end-to-end communication among diverse SDN domains. A virtual network topology was built to meet these conditions and is presented in Fig. 3. The proposed system consists of three administrative SDN domains, each managed by its own ONOS SDN controller. In the top-most layer of the current architecture the SDN-IP application is running that enables the communication between SDN domains using BGP. At the data path layer there are terminal hosts and software switches (i.e., OpenvSwitch) interacting to the associated SDN controller via Southbound (i.e., OpenFlow).

Therefore, we have configured the entire network topology using the Mininet emulator. The topology has three SDN domains, each managed by its controller. The central domain (A) works as a transit AS, responsible for interconnecting the remaining external networks. Each external network, in this case B, C is considered a different AS, which interfaces with the central domain (A) through routers, running Quagga, a well-known software emulator for routing packets. In the central domain (A), there is an SDN controller with an SDN-IP application running on its top that learns BGP routes to destination prefixes previously announced by the BGP routers of the network topology.

After the learning phase, the SDN controller of domain A translates each learned BGP route to SDN intents. Then, the same SDN controller converts each intent in to several flow rules, which are then transferred from the SDN controller to the data plane switches, using the OpenFlow protocol. These switches are the ones previously selected by the SDN controller to support an AS transit ingress/egress routing intent associated to a previously announced BGP IPv4 prefix.

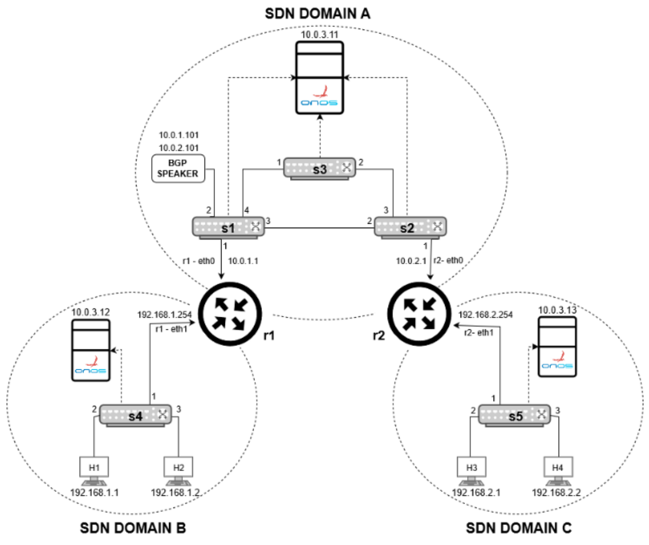


Figure 3. Multi-domain topology.

B. *QoS Deployment*

This scenario considers a security system monitorization, installed on the public road. This consists of vigilance cameras equipped with motion sensors transmitting RTP video flow by VLC and generic user computers generating UDP traffic. Motion sensor cameras were simulated using network devices. The testbed topology for this is shown in Fig. 4.

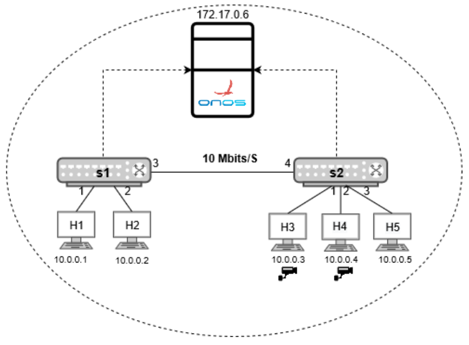


Figure 4. QoS testbed topology.

We deployed in one SDN domain to test quality of service (QoS_topology.py), but the same logic can be extended to larger scenarios implementing multiple domains. We limited all network links to 10 Mbit/s, using the Traffic Control (TC).

Initially all the traffic is going through the same path and if the motion sensor detects movement, the vigilance cameras should have a higher priority than the other non-video traffic. This implies the video traffic is transferred to a new queue and consequently can transmit the video with the highest quality without the competition of another non-video traffic. The queues are configured in OVS switch s1 using *ovs-vsctl* within the Mininet script that builds up the topology used in the current scenario.

As a conclusion of this sub-section, we assume that the traffic exchanged through the testing network should not be treated equally, e.g., we need to prioritize the usage of communication link resources. Therefore, in our proposal we will effectively prioritize the traffic in a network that is used by a mixture of IoT and legacy traffic. The traffic prioritization is based on creating distinct virtual output queues offered at the network switches. In addition, we have used a script that via Northbound API (e.g., HTTP POST request) forces the installation of adequate flow rules on the data plane switches. These flow rules allow traffic to be routed to different queues according to each traffic priority.

V. PROPOSAL EVALUATION

This section evaluates the solution in terms of its main functionality, the automatic reaction to a network failure, and the differentiated support of QoS for concurrent flows.

A. *System Validation*

The ONOS GUI on Fig. 5 shows the SDN ONOS controlled topology and summary information at the top.

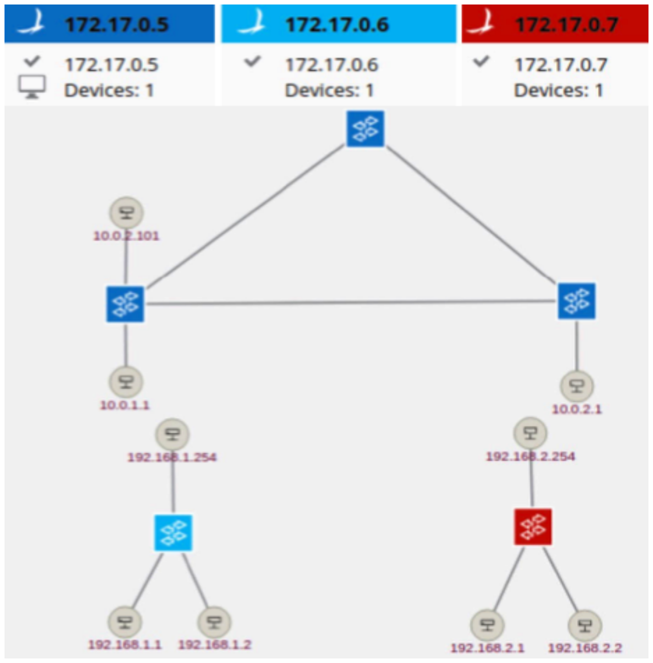
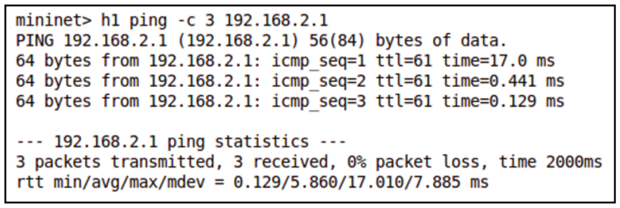


Figure 5. Topology at ONOS web GUI.

There are three SDN controllers, each one represented by a colour to evidence the network devices controlled by that controller. The first SDN controller (172.17.0.5) controls the transit domain, which contains three central switches. The second SDN controller (172.17.0.6), represented by the light blue colour, manages the left domain, which contains a single switch, interconnecting two terminal hosts (for example, h1 with IP address 192.168.1.1/24). The same happen with the SDN domain (172.17.0.7) represented by red colour on the right, which contains a switch with two hosts (h3 and h4). Hence, we have a physically distributed system with multiple controllers, each managing its own domain autonomously, but the central domain is managed by the ONOS SDN-IP Application. We have validated our system using ICMP traffic originated at host h1 (192.168.1.1) with destination at host h3

(192.168.2.1). Analyzing Fig. 6, the first ICMP attempt has a larger Round Trip Time than remaining ones because the SDN controller after deciding about the message routing path of the first attempt (reactive mode), it installs in the switches the path rules for next ping attempts (proactive mode).



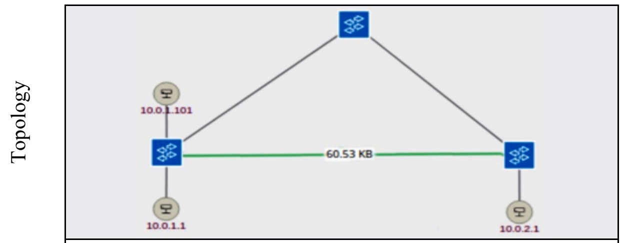
```
mininet> h1 ping -c 3 192.168.2.1
PING 192.168.2.1 (192.168.2.1) 56(84) bytes of data.
64 bytes from 192.168.2.1: icmp_seq=1 ttl=61 time=17.0 ms
64 bytes from 192.168.2.1: icmp_seq=2 ttl=61 time=0.441 ms
64 bytes from 192.168.2.1: icmp_seq=3 ttl=61 time=0.129 ms

--- 192.168.2.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2000ms
rtt min/avg/max/mdev = 0.129/5.860/17.010/7.885 ms
```

Fig 6. Successful connectivity test using ping from h1 to h3.

B. *Link Failure Test*

System failure detection is a very important aspect for ensuring fault tolerance in large scale distributed systems. In our case, if the SDN controller detects a link failure, it should quickly and effectively divert traffic to an alternate path to ensure the continuation of the communication service until the primary link is again operational. The goal is to reduce the time required to detect a failure and mitigate its negative impact on the traffic network routing.

Fig. 7 shows selected messages from several traffic captures made by Tcpdump. At the beginning of the test, the topology was operating without any failure and the used routing path between h1 and h3 was through switches s1 and s2 of the transit Domain A (s1-eth3, s2-eth2). One can also note that the initial ICMP Request TTL is 64 (h1-eth0) and then it is decremented down to 61 (h2-eth0), meaning that message has traversed three routers (i.e., r1, BGP speaker, r2) on its way to the destination node. Through the shortcut “A” in the ONOS GUI, which is shown in the first row of Fig. 7, one can see the traffic path being used and its speed.

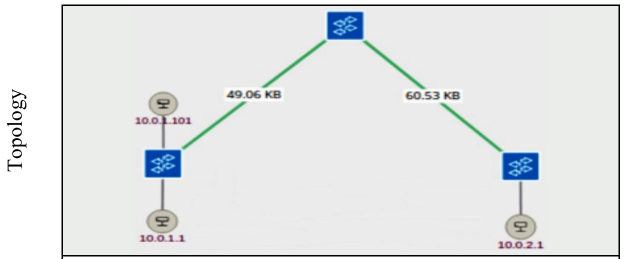


H1-eth0	10:58:36.709985 IP (tos 0x0, ttl 64, id 13117, offset 0, flags [DF], proto ICMP (1), length 84) 192.168.2.1 > 192.168.2.1: ICMP echo request, id 12153, seq 1, length 64
S2-eth2	10:58:36.741345 IP (tos 0x0, ttl 62, id 13117, offset 0, flags [DF], proto ICMP (1), length 84) 192.168.2.1 > 192.168.2.1: ICMP echo replay, id 13117, seq 1, length 64
S3-eth1	
H3-eth0	10:58:36.756439 IP (tos 0x0, ttl 61, id 3307, offset 0, flag [DF], proto ICMP (1), length 84) 192.168.2.1 > 192.168.2.1: ICMP echo replay, id 12153, seq 1, length 64

Figure 7. ICMP request from H1 to H3, S1-eth3 UP

Then, we turned off the link between s1 and s2, forcing that link to fail. This implied an event communication failure associated to a specific ONOS intent. This intent is like a routing path through the transit domain that incorporates the failed link. Consequently, after the failure occurrence, the ONOS analyzes the topology of the transit domain to find out an alternative path, which it should also interconnect the same ingress/egress points of the transit domain that were being used before that failure. In the current experiment, as indicated in Fig. 8, the alternative path through the transit Domain A was as follows s1-eth4, s3-eth1, s3-eth2, and s2-eth3.

We have validated the SDN-IP/BGP integration proposal, using a scenario where a failure in a specific routing path was mitigated by the functional robustness of ONOS intents. For future work, we aim to measure the time required to detect a link failure and to successfully detour traffic from that failure.



H1-eth0	11:05:34.305256 IP (tos 0x0, ttl 64, id 61729, offset 0, flags [DF], proto ICMP (1), length 84) 192.168.2.1 > 192.168.2.1: ICMP echo request, id 11341, seq 1, length 64
S2-eth2	
S3-eth1	11:05:34.327286 IP (tos 0x0, ttl 62, id 61729, offset 0, flags [DF], proto ICMP (1), length 84) 192.168.2.1 > 192.168.2.1: ICMP echo replay, id 12361, seq 1, length 64
H3-eth0	11:05:34.329246 IP (tos 0x0, ttl 61, id 61729, offset 0, flags [DF], proto ICMP (1), length 84) 192.168.2.1 > 192.168.2.1: ICMP echo request, id 11341, seq 1, length 64

Figure 8. ICMP request from H1 to H2, S1-eth3 DOWN

C. *Qos Test Validation*

Here, the QoS deployment topology is validated. When the topology is started, three devices will be enabled, two of them are VLC terminals and another is an RTP video server. Each VLC terminal receives a video from a simulated remote vigilance camera. In the device with the video server, the streaming of the video was started, which is consumed by two distinct VLC clients, simulating videos from two remote webcams. As mentioned, one of the videos is on the switch priority queue and the other is on the non-priority queue, sharing the available network resources with other flows.

Fig. 9 shows the rate trend of three flows used in the current test. It shows the system reaction after the video on the non-priority queue suffers the interference from UDP traffic, which tends to starve all the available network resources. Interference may be accessed using quality monitors [15] placed at strategic network point.

The trend of Fig. 9 is basically divided into three time intervals. The first one (between 8s and 24s) is when there is no interference in the video transmission of camera 2, because we still have no interference from UDP traffic over the RTP video traffic that uses the switch non-priority queue. We can see that when the video transmission starts, the blue line (camera 1) is transmitting the video at the same rate (1 Mbps) of the red line (camera 2). In addition, the camera 1 is in the high priority queue and camera 2 is in the low priority queue.

The second interval begins around 24s, when UDP traffic is injected for the purpose to cause interference with the camera 2 video transmission. Therefore, we can see that UDP (black line) traffic uses practically all the link bandwidth (i.e., around 8 Mbps) and the camera 2 rate significantly decreases (temporarily below 80 Kbps), degrading the quality of the received video from that camera (see Fig. 10, right side). This occurs because the UDP traffic is competing with camera 2 traffic at the same output queue. At this moment, we do not yet observe any corrective action from the SDN system to protect the quality of camera 2 video.

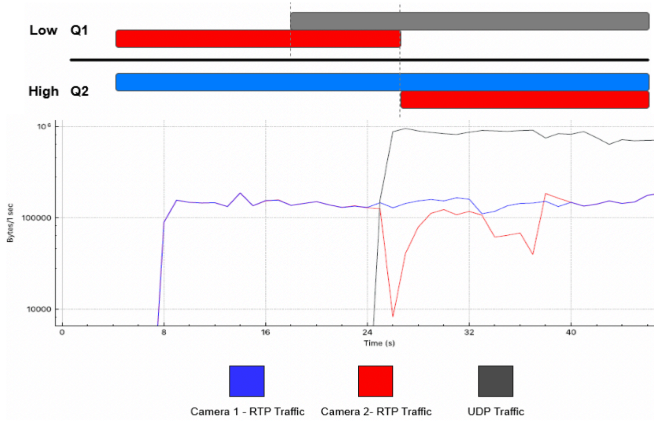


Figure 9. Rate trend of the three flows in our QoS test.

In the last time interval of current test, starting around 26s, the QoS mechanism is applied to improve transmission quality for camera 2. In this way, a flow rule is dynamically set to change the video to the switch high priority queue. In this way, we can see that the video transmission of camera 2 return to its normal rate and consequently enhance the perceived quality at the receiver. We can conclude that at that moment the UDP traffic is no longer interfering with the transmission quality from camera 2.

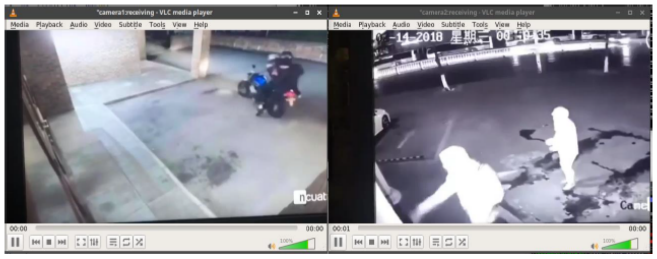

*camera1:receiving - VLC media player
*camera2:receiving - VLC media player

Figure 10. Remote vigilance videos with UDP traffic competition.

VI. CONCLUSIONS AND FUTURE WORK

The current work main goal was to understand how to deploy and manage a network infrastructure formed by several administrative domains, with multiple SDN controllers, satisfying QoS and robustness requirements of heterogeneous flows, some originated from IoT devices.

Our experimental results have shown that the proposed SDN-based solution can ensure communication between physically distributed SDN domains via the BGP protocol through a transit SDN system with the SDN-IP application running on the ONOS controller. We also demonstrate that our contribution is sensitive to link failures by redirecting traffic directly to another available path and ensuring the normal network operation.

Referring to quality of service, we have also validated within a network domain ruled by an SDN controller that traffic prioritization can be deployed. For that, some OpenFlow rules were installed in the data plane switches, which have output queues differentiated by the level of quality of service they aim to serve. In this way, we have shown that video from remote surveillance cameras, despite the presence of UDP traffic that normally starves all the available resources, can be transmitted with an optimum quality, thus meeting pertinent safety concerns in public environments. Further work is envisioned for testing the QoS scenario with IoT IPv6-compatible devices across ASs.

REFERENCES

[1] F. X. A. Wibowo, M. A. Gregory, K. Ahmed, and K. M. Gomez, “Multi-domain Software Defined Networking: Research status and challenges,” *J. Netw. Comput. Appl.*, vol. 87, pp. 32–45, Jun. 2017.

[2] Y. Zhang, L. Cui, W. Wang, and Y. Zhang, “A survey on software defined networking with multiple controllers,” *J. Netw. Comput. Appl.*, vol. 103, pp. 101–118, Feb. 2018.

[3] A. Gupta *et al.*, “SDX: A software defined internet exchange,” *Comput. Commun. Rev.*, vol. 44, no. 4, pp. 551–562, 2015.

[4] V. Kotronis, X. Dimitropoulos, R. Kloti, B. Ager, P. Georgopoulos, and S. Schmid, “Control Exchange Points: Providing QoS-enabled End-to-End Services via SDN-based Inter-domain Routing Orchestration,” pp. 3–4, 2016.

[5] P. Berde *et al.*, “ONOS : Towards an Open , Distributed SDN OS,” pp. 1–6.

[6] S. Badotra, “Open Daylight as a Controller for Software Defined Networking,” no. May 2017, 2018.

[7] M. Gerola *et al.*, “ICONA: Inter Cluster ONOS Network Application,” 2015.

[8] D. Gupta and R. Jahan, “Inter-SDN Controller Communication: Using Border Gateway Protocol,” no. April, pp. 1–16, 2014.

[9] T. Hu, Z. Guo, P. Yi, T. Baker, and J. Lan, “Multi-controller Based Software-Defined Networking: A Survey,” *IEEE Access*, vol. 6, pp. 15980–15996, 2018.

[10] F. Bannour, S. Souihi, and A. Mellouk, “Distributed SDN Control: Survey, Taxonomy, and Challenges,” *IEEE Commun. Surv. Tutorials*, vol. 20, no. 1, pp. 333–354, 2018.

[11] V. Kotronis, A. Gämperli, and X. Dimitropoulos, “Routing centralization across domains via SDN: A model and emulation framework for BGP evolution,” *Comput. Networks*, vol. 92, pp. 227–239, 2015.

[12] M. Ndiaye, G. P. Hancke, and A. M. Abu-mahfouz, “Software Defined Networking for Improved Wireless Sensor Network Management : A Survey,” pp. 1–32, 2017.

[13] A. C. Anadiotis, L. Galluccio, S. Milardo, G. Morabito, and S. Palazzo, “SD-WISE: A Software-Defined WIreless SEnsor network,” *Comput. Networks*, vol. 159, pp. 84–95, 2019.

[14] W. Cerroni *et al.*, “Intent-based management and orchestration of heterogeneous openflow/IoT SDN domains,” *2017 IEEE Conf. Netw. Softwarization Softwarization Sustain. a Hyper-Connected World en Route to 5G, NetSoft 2017*, 2017.

[15] J. R. S. Soares, L. A. Da Silva Cruz, P. Assuncao, and R. Marinheiro, “No-reference lightweight estimation of 3D video objective quality,” in *2014 IEEE International Conference on Image Processing, ICIP 2014*, 2014, pp. 763–767.