Delay-based Priority Queueing for VoIP over Software Defined Networks

Cristian Olariu, Martin Zuber, Christina Thorpe
UCD, School of Computer Science, Belfield, Dublin 4, Ireland
[firstname.lastname]@ucd.ie

***Abstract*—Software Defined Networking and Network Function Virtualisation provide significantly more flexibility and control when provisioning QoS for delay sensitive services. Network delay can have a detrimental impact on VoIP quality and, therefore, minimising delay can help maintain the quality and prevent call dropping. This paper proposes a queuing scheme based on packet delay for prioritising VoIP calls made over SDN. An OpenFlow testbed was developed to validate our proposal, and results show that delay-based prioritisation of VoIP packets in SDN networks ensures that only a small number of users will be affected by network congestion.**

I. Introduction

The recent advent of Software Defined Networking (SDN) and Network Function Virtualisation (NFV) has lead to a paradigm shift for network and service management. Both SDN and NFV have gained significant momentum in the research community and in industry; they support a flexible and rapid orchestration of networks and the services that traverse them [1]. The increased control offered by these new paradigms have boosted the capabilities of network administrators to perform the advanced monitoring and QoS provisioning needed to ensure high quality, delay-sensitive services [2].

VoIP is extremely sensitive to network conditions [3][4]; the quality of a call can degrade if there is loss, delay, or jitter on the network links between two end users. Additionally, some VoIP packets and calls are more important than others [5]. Hence, there has been some research activity in the area of VoIP call prioritisation [6][7].

Our paper presents an SDN-based approach for intra-VoIP packet prioritisation, where the VoIP packets are prioritised based on the length of time spent in the network. It effectively implements prioritisation as a virtualised network function; it is based on our previous work in wireless mesh networks [6] .

An SDN testbed was developed using OpenFlow [8], mininet [9], and Floodlight. We leveraged on our previous experience with developing OpenFlow SDN testbeds: for service assurance in IPTV [10], [11], and for advanced VoIP quality monitoring [12]. We use the iMOS metric, first presented in [13] and then in an SDN context [12], to measure the intermediate VoIP quality as the packets traverse the network.

The remainder of this paper is organised as follows: Section II discusses the related work published in the literature. Section III describes the mechanism we developed in this work. Section IV details the experimental setup configured for the simulations carried out, including the topology, experiments, and improvements. Section V presents the results and analysis. Finally, Section VI concludes the paper.

II. Related Work

Afaq et.al. [14] propose a framework to identify delay sensitive traffic flows in a data centre network. Xu et. al. [15] propose a QoS-enabled management framework to create an end-to-end communication service over SDN. PolicyCop [16] is an autonomic QoS policy enforcement framework for SDN, which provides an interface for specifying QoS requirements in SLAs and enforcement via the OpenFlow API. Qazi et. al. [17] proposed Atlas, which incorporates application-awareness into SDN. Kumar et. al. [18] looks at improving the QoE of services in home networks using SDN. Kwon et. al. [19] propose the adaptive Mobile Voice over Internet Protocol (mVoIP) service architecture in SDN networks to provide the best quality of mVoIP service to end-users.

Our proposal is different in that we define multiple queues with different priority classes, which are used to prioritise VoIP packet based on delay. We specifically target VoIP over SDN and leverage the visibility afforded to the controller when recording delay values and allocating queues.

III. Mechanism description

In our previous work [6], we showed how VoIP packets can be re-prioritised in the same queue. This solution brought improvements in terms of capacity and increased fairness of the network with regards to delay. However, such a solution involves using a PIFO (Push-In-First-Out) queue, which does not exist in SDN nodes. For this reason, we propose to use more than one queue to serve VoIP packets; the difference among these queues being the priority with which they are served by the queue scheduler. We propose to use 5 different VoIP queues to allow for a more granular differentiation between packets affected by various levels of delay. For example, more delayed packets will be served by a high priority VoIP queue, whereas other less delayed packets will be placed in lower priority queues.

In order to demonstrate the effects of our proposed scheme, we built an SDN application that processes every single VoIP packet that traverses the SDN defined in our experimental scenario. No flow entries are added on the SDN switches; this forces the switches to forward every VoIP packet to the controller via 'Packet In' messages. Our controller application decides in which queue to place the incoming packet. As decision thresholds, 5 different thresholds have been chosen: (1) packets delayed by less than 5 milliseconds are placed in the lowest priority queue, (2) between 5 and 10 milliseconds in the second lowest priority queue, and so on ((3) (4)), and lastly, (5) packets delayed by more the 20 milliseconds will be placed in the highest priority queue.

The SDN application starts tracking VoIP packets when they first enter the SDN network. Using the information contained in the RTP header, each packet can be uniquely identified and time-stamped upon arrival at the ingress port of each SDN node along the path. Hence, the SDN application is aware of each VoIP packet's delay and will place the packet in a queue serving similarly delayed packets. We realise that tracking every single VoIP packet in this manner is unfeasible in a production network, future work will include the design and implementation of a more realistic solution that is more practical.

IV. Experimental setup

This section details the experimental setup designed for this work, investigating two ideas:

1) the VoIP transmission performance on a bandwidth-limited network.
2) how to manipulate transmissions using a priority queue based mechanism utilising the QoS support introduced in OpenFlow version 1.3.0 protocol [8].

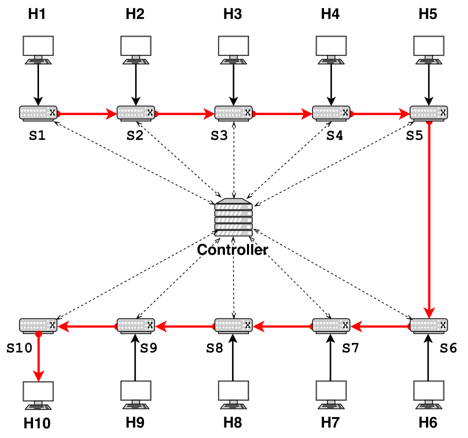

Fig. 1. Experiment Topology

We used a powerful server to simulate an SDN network using Mininet, and we implemented our proposal in an application running on top of the SDN controller.

A. Topology

The experiment topology consists of a line topology composed of 10 switches (Figure 1), which is the network's backbone (red links). There are 10 hosts in the topology, and each host is connected to an OpenFlow switch (black links). Each switch has also an out-of-band connection to the SDN controller (grey dotted links). Each backbone link between two switches has a delay and bandwidth limit configured. This was accomplished using Linux *tc*, set up on each switch's egress interface.

B. Technology

The technical configuration, with regards to the software and hardware, for the test-bed is detailed in the following. We used a machine running Ubuntu Server 14.04 as operating system, and installed Mininet 2.2.1, OVSwitch 2.3.1, and the latest stable version of the Floodlight SDN Controller.

Mininet and OVSwitch are part of the Mininet installation package [9]. The installation works with Ubuntu box 14.04 LTS, and the specific versions detailed above are important for experiment reproducibility and compatibility purposes. For example, OVSwitch and Floodlight must both support OpenFlow version 1.3. The experiment utilises OpenFlow 1.3 QoS support for new *set_queue* action command, which allows the controller to instruct an OVSwitch to enqueue a packet into a specific queue on a network interface.

The hardware specification of the machine is as follows: CPU – Intel Xeon E5630 with 8 Cores @ 2.53GHz and L2 Cache of 12288KB, Memory – 24GB of RAM with 24GB of swap, and storage – SCSI ATA HD of 894GB.

C. Simulating VoIP calls

In order to create a congestion scenario, we generate one-way VoIP UDP traffic into the backbone network, with cumulative bit-rate slightly higher that the backbone's bandwidth capacity. Conversations (300s long between two people) recorded in *wav* files, encoded with the *opus* codec, are simultaneously transmitted from nine hosts (H1 - H9) towards the destination host H10.

D. VoIP congestion

We limited the bandwidth of each link to support slightly less than 9 VoIP calls; this is used to create congestion on switch S9's egress interface. The result being that the backlog of packets waiting for bandwidth is created on switch S9's egress interface txqueue (transmit queue). Since the txqueue is capable of holding a maximum of *1000* packets in a waiting state, packet loss does not occur.

V. Results

The proposed solution favours longer delayed packets over less delayed packets by means of different priority VoIP packet queues. As a first step, we investigated the distribution of the delay that affects packets that reach the congested switch (S9 in Figure 1).

For the default single queue scenario, the distribution of delay ranges from 0 to 1300 milliseconds, which affects every VoIP call placed in the network. In the multi-queue scenario, 90% of the packets are in the 0 to 100 millisecond range of delay, while 10% of the packets are scattered around 3500 milliseconds. Basically, as there is one call more than what the network can support, the equivalent amount of packets belonging to a VoIP call have been pushed to the back by our solution. This ensures timely passage of the other packets. This has an impact of the expected user experience, as seen in Figures 2 and 3.

The extended queueing delay affects the MOS and we observe a decrease of MOS in time for each conversation (Figure 2). In other words, switch S9's egress interface has enough bandwidth for 8 conversations to pass but not enough for 9 conversations that are actually transmitted. At this point, a smart VoIP service would adapt itself and employ a call admission control or other similar schemes to limit the number of calls that go through the network.

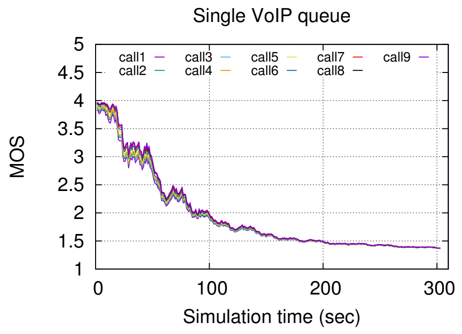

Fig. 2. Conversations MOS over time, when a single VoIP queue is employed, i.e. baseline case.

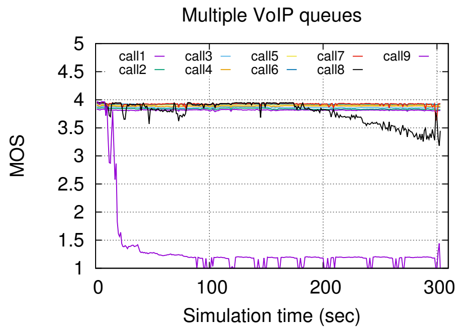

Fig. 3. Conversations MOS over time, when 5 different priority VoIP queues are employed.

From figure 3, it is apparent that the conversation that originated on H9, which is closest to the congested node, suffers the greatest MOS degradation. The packets from that conversation are getting assigned to the lowest priority queue. We can also observe that the conversation that originated on H8 starts to be affected by the scheme around 170 seconds into the experiment when the MOS value starts to degrade slowly. This behaviour is caused by a much slower delay build-up. In other words, our proposed solution allows the network to employ a delay-based call admission control, by systematically de-prioritising VoIP packets that have suffered low network delay, which in most cases is proportional to the resources used thus far until they reached a congested node.

VI. Conclusion

The focus of this paper was to propose a delay-based queue prioritisation scheme for VoIP service management over SDN. The fundamental idea of the work is that packets that have already consumed network resources, and thus have a higher delay, should be prioritised in order to prevent resource wastage. Results show that employing our scheme significantly increased the number of good quality calls supported by the network. Future work will involve developing a fairer algorithm, which may use selective packet drops to maintain a 'sufficient' call quality for all calls and prevent call dropping.

Acknowledgment

Supported, in part, by Science Foundation Ireland grant 10/CE/I1855 and by Science Foundation Ireland grant 13/RC/2094 and by Enterprise Ireland Grant IP20140344.

References

[1] H. Kim and N. Feamster, "Improving network management with software defined networking," *IEEE Communications Magazine*, vol. 51, no. 2, pp. 114–119, 2013.

[2] C. Monsanto, J. Reich, N. Foster, J. Rexford, D. Walker *et al.*, "Composing Software Defined Networks," in *NSDI*, 2013, pp. 1–13.

[3] A. P. Markopoulou, F. Tobagi, M. J. Karam *et al.*, "Assessment of voip quality over internet backbones," in *INFOCOM 2002*, vol. 1. IEEE, 2002, pp. 150–159.

[4] L. Zheng, L. Zhang, and D. Xu, "Characteristics of network delay and delay jitter and its effect on voice over ip (voip)," in *Communications, 2001. ICC 2001. IEEE International Conference on*, vol. 1. IEEE, 2001, pp. 122–126.

[5] C. Hoene, B. Rathke, and A. Wolisz, "On the importance of a voip packet," in *ISCA Tutorial and Research Workshop on Auditory Quality of Systems*, 2003.

[6] C. Olariu and A. Hava, "Dapp: A delay-aware packet prioritisation scheme for voip in wireless multi-hop networks," in *2016 9th IFIP Wireless and Mobile Networking Conference (WMNC)*. IEEE, 2016, pp. 49–56.

[7] P. O. Flaithearta, H. Melvin, and M. Schukat, "Optimising qos of voip over wireless lans via synchronized time," in *World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2013 IEEE 14th International Symposium and Workshops on a*. IEEE, 2013, pp. 1–3.

[8] Open Network Foundation, "OpenFlow Switch Specification Version 1.3.0."

[9] M. Team, "Native installation from source." [Online]. Available: http://mininet.org/download

[10] P. McDonagh, C. Olariu, A. Hava, and C. Thorpe, "Enabling iptv service assurance using openflow," in *Advanced Information Networking and Applications Workshops (WAINA), 2013 27th International Conference on*. IEEE, 2013, pp. 1456–1460.

[11] C. Thorpe, C. Olariu, A. Hava, and P. McDonagh, "Experience of developing an openflow sdn prototype for managing iptv networks," in *Integrated Network Management (IM), 2015 IFIP/IEEE International Symposium on*. IEEE, 2015, pp. 966–971.

[12] C. Thorpe, A. Hava, J. Langlois, A. Dumas, and C. Olariu, "imos: Enabling voip qos monitoring at intermediate nodes in an openflow sdn," in *2016 IEEE International Conference on Cloud Engineering Workshop (IC2EW)*, April 2016, pp. 76–81.

[13] C. Olariu, J. Fitzpatrick, P. Perry, and L. Murphy, "A QoS based call admission control and resource allocation mechanism for LTE femtocell deployment," in *Consumer Communications and Networking Conference (CCNC), 2012 IEEE*. IEEE, 2012, pp. 884–888.

[14] M. Afaq, S. U. Rehman, and W. C. Song, "Visualization of elephant flows and qos provisioning in sdn-based networks," in *Network Operations and Management Symposium (APNOMS), 2015 17th Asia-Pacific*, Aug 2015, pp. 444–447.

[15] C. Xu, B. Chen, and H. Qian, "Quality of service guaranteed resource management dynamically in software defined network," *Journal of Communications*, vol. 10, no. 11, 2015.

[16] M. F. Bari, S. R. Chowdhury, R. Ahmed, and R. Boutaba, "Policycop: an autonomic qos policy enforcement framework for software defined networks," in *Future Networks and Services (SDN4FNS), 2013 IEEE SDN for*. IEEE, 2013, pp. 1–7.

[17] Z. A. Qazi, J. Lee, T. Jin, G. Bellala, M. Arndt, and G. Noubir, "Application-awareness in sdn," in *ACM SIGCOMM Computer Communication Review*, vol. 43, no. 4. ACM, 2013, pp. 487–488.

[18] H. Kumar, H. H. Gharakheili, and V. Sivaraman, "User control of quality of experience in home networks using sdn," in *Advanced Networks and Telecommuncations Systems (ANTS), 2013 IEEE International Conference on*. IEEE, 2013, pp. 1–6.

[19] D. Kwon, R. Thay, H. Kim, and H. Ju, "Qoe-based adaptive mvoip service architecture in sdn networks," *ICN 2014*, p. 73, 2014.