On QoS Support to Ofelia and OpenFlow

Balázs Sonkoly*[†], András Gulyás*[‡], Felicián Németh*[‡],
János Czentye*, Krisztián Kurucz*, Barnabás Novák*, Gábor Vaszkun*
*Department of Telecommunications and Media Informatics
Budapest University of Technology and Economics
Email: {sonkoly,gulyas,nemethf}@tmit.bme.hu
[†]MTA-BME Future Internet Research Group
[‡]Hungarian Academy of Science (MTA) Information system research group

***Abstract*—OpenFlow is the most promising architecture for future Software Defined Networks (SDNs). However, from the aspects of large-scale or carrier-grade networks, it still lacks some key components. For example, QoS (Quality of Service) provisioning is an indispensable part of such production networks. During the evolution of the OpenFlow standard, some QoS capabilities have been added to the protocol, however, even the latest version has only a limited and not well-defined QoS framework. Hence, integrated QoS support is missing in current OpenFlow experimental testbeds including Ofelia. This paper describes a possible architectural extension to Ofelia in order to make it capable of running QoS related experiments. We summarize the initial tasks regarding the survey of QoS features and limitations of OpenFlow switches deployed in Ofelia islands and the performance tests needed to characterize these devices. For extending the feature palette of Ofelia, we propose a QoS management platform with full integration into the existing management framework. By means of this envisioned extension, QoS settings of the whole Ofelia testbed can be adjusted easily, in a user friendly fashion. Moreover, we walk through the main steps needed not only towards an integrated OpenFlow testbed with QoS support but towards a QoS architecture to OpenFlow.**

***Index Terms*—OpenFlow, QoS, configuration, network management**

I. Introduction

The increasing diversity of emerging networking applications generates flows with more and more diverse characteristics to be carried over the Internet (e.g., http, p2p, audio and video streaming, e-mail, ftp, etc.). Moreover, these various traffic types require different treatment from the carrier network to finally meet the Quality of Experience (QoE) requirements of the end users. The dire need to accommodate such level of diversity and the ever increasing expectations of the users keeps architectures that can provide traffic differentiation and Quality of Service (QoS) guarantees for the carried traffic in the centre of networking-related research and development activities worldwide. As a result of these efforts, some QoS capabilities became available also in OpenFlow since version 1.0. Although the possibility for QoS is available there, the heterogeneity of hardware and software implementations of QoS in the products of different vendors renders the setup of large-scale OpenFlow testbeds with QoS support as a highly non trivial task. Due to these issues, integrated QoS support is missing in the currently available OpenFlow testbeds including Ofelia.

QoS provisioning in packet switched networks has a very long history. In the overview of Campbell et al. we can read the following about the state of QoS in 1994 [1]:

"Recent technological developments in high speed networks and multimedia workstations are making possible entirely new classes of distributed application such as distance learning, desktop video conferencing and remote multimedia database access. In these applications, communication requirements are extremely diverse and demand varying levels of latency, bandwidth and jitter, etc. Furthermore, for continuous media such as video and audio it is often a requirement that levels of service are guaranteed. Other time critical distributed applications such as distributed real-time control systems are also growing in prominence. These applications have stringent quality of service (QoS) requirements for both reliability and guaranteed bounds on message latency ... however existing architectures are based on a best effort performance model and were never designed to support quantitative QoS. In the Internet the support of reliable data transfer was a primary design goal and performance QoS was only a marginal consideration." (Campbell, A. and Coulson, G. and Hutchison, D. "A quality of service architecture" ACM SIGCOMM Computer Communication Review 1994)

Since the appearance of this paper a plethora of studies considered QoS issues in packet networks and proposed scores of mechanisms and architectures to elaborate satisfactory solutions. In the light of volume of the QoS-related literature generated in the last two decades, one can be surprised that the above statements still hold in today's packet networks. The reasons behind the phenomenon that QoS became such a chronic issue are definitely complex. However, we believe that the closed interfaces of the networking equipments significantly contributed to turning QoS provisioning into a permanent and classic problem.

Because OpenFlow can support traffic differentiation and therefore QoS, its open architecture can activate and incentivize the research community to revisit QoS from a practical perspective and work out functioning systems, similarly as BitTorrent managed to implement multicast in the application layer. We believe that enabling QoS in large-scale OpenFlow testbeds like Ofelia would promote such QoS related research and development activities and significantly contribute to find solutions to this classical problem.

The main goal of the paper is to give a potential architectural extension to Ofelia enhancing the feature palette with integrated and manageable QoS support. To achieve this goal, we identify the main steps including in depth study on Ofelia's OF switch arsenal digging for available (maybe vendor specific) QoS capabilities; performance measurements of the heterogeneous devices under diverse QoS settings from realistic QoS scenarios and use-cases; identification of future enhancements in the QoS configuration of the OpenFlow switches (e.g., support for different queuing logic, specification of various QoS parameters); implementation of novel QoS functions in software switches and NetFPGA cards; and finally the extension of the current management system of Ofelia to be able to manage QoS in a unified and integrated manner.

The rest of the paper is organized as follows. In Section II a state of the art about the QoS support in OpenFlow is given and the OpenFlow related network management approaches are summarized. In Section III the main QoS related issues are highlighted for Ofelia testbeds. Section IV is devoted to our architectural extensions and the key tasks needed to strengthen Ofelia and OpenFlow with QoS support. Section V concludes the paper and gives future plans.

II. Background

A. *Quality of Service in OpenFlow*

The OpenFlow community realized the importance of Quality of Service support by enriching the 1.0 version of protocol with some QoS capabilities [2]. However, even the latest, 1.3 version has only a limited QoS support through a simple queuing mechanism [3]. In version 1.0, packets can be forwarded to queues of output ports by the optional enqueue action, which was transformed into the optional set-queue action in version 1.2. Controllers can query various queue statistic and queue configuration parameters through the OpenFlow protocol, but the protocol has no support to create queues or alter the behavior of existing queues. Therefore queue configuration needs to take place outside of the OpenFlow realm.

Although OpenFlow protocol does not support creating or modifying queues, an OpenFlow capable switch can be queried to report the queues attached to a specific port, and the guaranteed minimum rate associated with a queue. Additionally, after version 1.2, it can report to controllers the maximum rate as well. The minimum and the maximum rates are, of course, not enough to build a moderately complicated QoS scenario, but traffic queues can have other characteristics that cannot be queried. For example, an OpenFlow controller is unable to distinguish a small sized RED queue with zero minimum threshold from an enormous FIFO queue causing bufferbloat.

The read-only nature of the queue properties greatly limits the QoS capabilities of OpenFlow. To mitigate this limitation, for example, HP defined their rate-limiter vendor extensions. Using these extensions the controller is able to remotely create a rate limiter object that drops packets exceeding a preset limit and to add actions to flow tables that send matching packets through rate limiter objects. HP's own OpenFlow switches obviously supports the rate-limiter extensions [4] and HP proposed an extension to previous OpenFlow version, as well. Currently, OpenFlow v1.3 supports a similar rate-limiting functionality with the help of *meter tables.*

Instead of adding switch configuration instructions to the OpenFlow protocol, the Open Networking Foundation created an auxiliary protocol called OF-Config [5]. Its first version supports configuring an OpenFlow v1.2 capable switch.[1] From QoS point of view, it can be used only to remotely set minimum guaranteed and maximum rates of queues. Interestingly, OF-Config should be used to configure queues, but OpenFlow should be used to configure meter tables.

B. *Network management in OpenFlow*

Network management is an indispensable component of large-scale networks, however, OpenFlow has lacked a well-defined management framework. Recently, a new protocol has been proposed, namely OF-Config protocol [5], in order to take the first step toward a standard management framework. OF-Config is the configuration and management protocol for OpenFlow capable devices. Currently it is based on the Netconf [6], [7] network management protocol since it is more suitable for configuration management than traditional SNMP. The proposal also gives data models of OpenFlow components using the Yang [8] data definition language.

In spite of lacking standard management tools in OpenFlow, several OpenFlow capable testbeds were established in the recent years in the US and Europe, as well. These facilities provide specific control and management interfaces to experimenters focusing on the configuration of data plane and FlowVisor [9]. For example, Expedient [10] is the configuration tool used currently in Ofelia, and it was used by GENI previously. Now the network configuration in GENI is based on FOAM [11], [12]. Another important feature of OpenFlow is the capability for provisioning network virtualization. Currently, FlowVisor is the widely used component providing light-weight network virtualization, or network slicing. Naturally, the management tools have to be capable of configuring and managing the operation of this component, as well. A widely used companion tool of Expedient is Opt-In Manager [13] managing information on flowspaces owned by users and on running experiments. Slicing of network resources is solicited by pushing down the flowspace information to FlowVisor.

In our previous work [14], we have extended an open-source network management software, namely OpenNMS [15], with FlowVisor management capabilities. This extension provides not only configuration management but some performance metrics of flowspaces can also be queried. FlowVisor is an efficient tool, however, it limits the functionality of the OpenFlow switches as it is rigidly bound to the version of OpenFlow

[1]The latest version supports OpenFlow v1.3.

protocol. This is a serious drawback since FlowVisor has to be updated for every protocol update. In order to eliminate this drawback, we have recently proposed an integrated OpenFlow virtualization framework [16], which is capable of running and managing multiple instances and versions of OpenFlow switches with different forwarding capabilities, running and configuring controllers or network applications under the management of the proposed framework. We have also prepared a proof of concept prototype making use of open source components focusing on key elements of the architecture. Our management components are integrated with OpenNMS [15] and the components of the OpenFlow network (running OF switches and controllers) are extended with management interfaces. Our management interfaces are based on Netconf/Yang [6], [7], [8] and it was implemented making use of open-source tools (Yuma [17]).

III. QoS ISSUES IN OFELIA

For sophisticated QoS support, we have to set up queues with diverse settings and map the flows to these queues. The switches have to support this mapping through the OpenFlow protocol specification. Moreover the switches should support this without significant degradation of forwarding performance.

The practical first step of QoS provisioning is queue configuration of routing / switching devices. Completing this step is a daunting task since uniform queue configuration is not possible with OpenFlow. Each switch type needs to handled differently in the diverse set of equipments deployed in Ofelia. For example, command sequence to create a queue in NEC switch model IP8800/S3640 is completely different from that in HP 3500 switch, and both are conceptually different from that in software switches.

In a virtualized networking environment, the second step should be the resource allocation for an experiment. Ofelia's control framework, as Figure 1 shows, is designed around Expedient and Opt-In tools. However, Expedient lacks support to create or to configure queues. Even if an operator creates queues required for a QoS experiment, it is not possible to assign queue resources to an experiment. The flowspace is set in FlowVisor with the help of the Opt-in Manager, but FlowVisor in itself cannot eliminate the interference among experiments conducted in parallel, which might result in unintended QoS degradation.

Finally, even after having set up the slice and the flowspace, the experimenter's OpenFlow controller has to overcome the different QoS support implemented in switches. For example, HP switches do not implement the optional enqueue action, instead they map packets to priority queues using VLAN priority or IP Type-of-Service, DiffServ Code-Point fields [4], whereas NEC switches forward packets to queues by the standard enqueue action [18].

Currently, we see a large diversity in the available QoS support in proprietary switches. Therefore a comprehensive survey of the QoS features available in Ofelia switches or today's

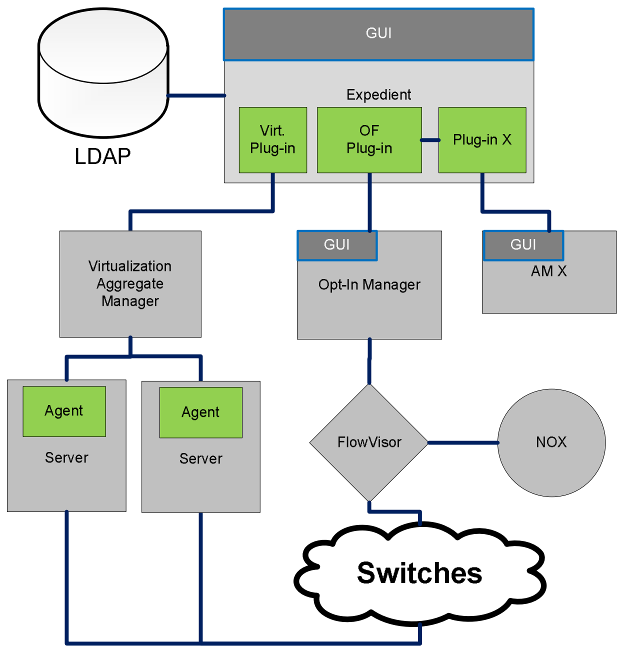

Fig. 1. General Architecture of Ofelia Control Framework.

OpenFlow switches is an important contribution towards the deployment of future applications with QoS requirements.

Another challenging issue is forwarding performance while QoS is at use. Therefore, in depth measurements regarding the available forwarding speed when large number of QoS flows are present, would be an expedient input for future Ofelia experiments.

Currently, we do not see any QoS support from the management system. However, for running QoS applications, an integrated management framework that can enable standard queue configuration and management in Ofelia switches would be highly beneficial.

IV. TOWARDS QoS SUPPORT TO OFELIA AND OPENFLOW

In order to enhance Ofelia testbed (and later OpenFlow protocol itself) with QoS support, some main steps are needed. This section is devoted to summarize these building blocks and to describe our envisioned QoS framework.

On the one hand, we have to define and run experiments on Ofelia facilities in order to evaluate and investigate current QoS capabilities of the devices in Ofelia islands. These specific measurements and evaluations on the Ofelia experimental *facility* need *extensions* of the already available infrastructure as QoS configuration is not supported in the current stage. On the other hand, functional enhancements to Ofelia experimental facility are also required to make it capable of QoS related experiments, as well. This involves the *extension* of current Ofelia *control framework* with QoS and queue configuration capabilities and *extensions* beyond current versions of *OpenFlow* specification to provide more sophisticated QoS support.

A. Comprehensive study on capabilities and performance

As Ofelia islands are equipped with a diverse set of OpenFlow capable devices, it is an important goal to investigate the capabilities of these switches regarding QoS configuration and QoS performance.

To achieve this goal, as a first step, the current possibilities in different versions of OpenFlow protocol and the recent OF-Config protocol have to be studied and summarized. The QoS capabilities of devices from different vendors (mainly focusing on devices included by Ofelia islands) can be revealed as a second step.

Based on the results of the investigation, a comprehensive QoS performance analysis on Ofelia testbeds is also inevitable. This includes the definition and running of different experiments regarding QoS capabilities of the devices and also includes the implementation of additional software components, such as controller applications. Currently available infrastructure of Ofelia experimental facility needs to be extended, as well, in order to support the QoS configuration in the network devices.

On the one hand, this analysis can provide a verification of the QoS configuration capabilities of different switches with different tools. On the other hand, we gain detailed information on the QoS performance of these devices and the testbed networks.

B. Extensions to Ofelia control framework

Figure 2 shows the control framework architecture with our proposed QoS extensions. The Queue Manager Plug-in empowers Expedient with uniform queue configuration capabilities; it hides equipment heterogeneity with consistent user interface to set-up and to manage queues of switches in the testbed, and to configure their properties. The Queue Manager translates users' request to vendor specific configuration command sequences and executes them through different configuration / management interfaces as shown in Figure 3. On the one hand, available SNMP interfaces can be used for hardware switches. On the other hand, for software switches, we have recently designed and developed an open source Netconf based interface with the accompanying QoS extensions [19] heavily relying on the traffic control (tc) linux kernel module. Currently OF-Config protocol exists only as a protocol specification document, but OF-Config support can easily be added into Queue Manager when an implementation to soft-switches or a new hardware firmware appears. In line with the findings in the QoS survey of OpenFlow devices deployed in Ofelia, Queue Manager can be connected with other devices as well, for example, the Netconf based extensions might easily be ported to OpenWrt based wireless routers.

Similarly to the current port assignment process, users will be able to select pre-configured queue into their slice with QoS enhanced Expedient. Likewise, the necessary extensions and modifications to FlowVisor and Opt-in Manager have to be added as well, so that the user defined flowspace could contain queues restricted from general access.

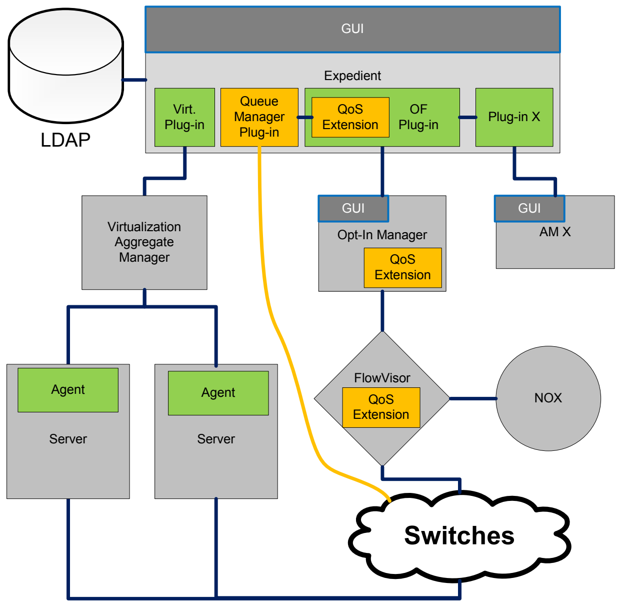

Fig. 2. General Architecture of OFELIA Control Framework extended with QoS support. Orange background denotes required extensions.

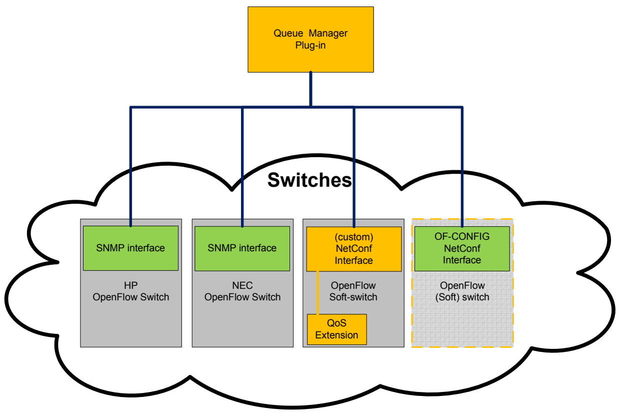

Fig. 3. Communication paths between the Queue Manager and OpenFlow capable switches. Orange background denotes required extensions.

C. OpenFlow extensions

In the envisioned QoS framework, the queue configuration is not restricted to minimum and maximum rates defined in the latest OpenFlow specification. As straightforward next steps, one should identify possible enhancements in QoS/queue configuration of OpenFlow switches; extend the OpenFlow protocol by defining experimenter (vendor specific) queue properties to selected queue types / settings; and implement support for these new queue properties in software or NetFPGA switches.

In order to demonstrate the benefits of the enhancements, we need different use cases to be defined; novel applications to be implemented; and measurements to be conducted in a wide range of network scenarios.

D. Summary

To sum up, our proposed QoS extensions to Ofelia's control framework architecture

- provides QoS enhancements of the facility enabling the development and testing of QoS sensitive OpenFlow applications,
- it also enables conducting experiments with our vendor specific queue properties extensions to go beyond OpenFlow v1.0,
- Queue Manager plug-in allows to uniformly configure QoS capabilities of devices hiding the differences in individual hardware components, which results in improved usability of the testbed.

V. Conclusion and Future Work

In this paper we have proposed architectural extensions to the main European OpenFlow experimental testbed, i.e., Ofelia. The goal of these enhancements is to make Ofelia the first available federated OpenFlow testbed which can provide manageable and integrated QoS support towards its experimenters and which is capable of running QoS related experiments. In this respect, Ofelia can become the main driver and host of QoS innovations in OpenFlow.

We have summarized the main steps and building blocks needed to strengthen Ofelia and OpenFlow with QoS support. This involves the initial tasks regarding the survey of QoS features and limitations of currently deployed OpenFlow switches and QoS performance tests needed to characterize these devices. Moreover, we have proposed a QoS management platform fully integrated with the existing management framework of Ofelia in order to extend the feature palette of the testbeds.

In general, a QoS architecture for OpenFlow with sophisticated QoS functions and flexible configuration management is also a challenging research direction. We have recently designed and prototyped a novel management framework for OpenFlow which is presented in a companion paper [19]. It is based on OpenNMS [15], an open-source network management system. The basic QoS/queue configuration functions have been integrated into this framework and we are going to do some of the previously presented steps in this environment to get a (currently only small-scale) OpenFlow testbed with enhanced QoS support.

Acknowledgements

This work was performed in the High Speed Networks Laboratory at BME-TMIT. The research was supported by OTKA-KTIA grant CNK77802.

References

[1] A. Campbell, G. Coulson, and D. Hutchison, "A quality of service architecture," *SIGCOMM Comput. Commun. Rev.*, vol. 24, no. 2, pp. 6–27, 1994.

[2] OpenFlow Consortium, "OpenFlow switch specification, version 1.0.0," Dec. 2009. [Online]. Available: http://www.openflow.org/documents/openflow-spec-v1.1.0.pdf

[3] Open Networking Foundation, "OpenFlow switch specification, version 1.3," Apr. 2012. [Online]. Available: https://www.opennetworking.org/images/stories/downloads/specification/openflow-spec-v1.3.0.pdf

[4] "HP switch software OpenFlow supplement," User's Manual, Software version K.15.05.5001, Nov. 2011.

[5] Open Networking Foundation, "OpenFlow configuration and management protocol 1.0 (OF-Config 1.0)." [Online]. Available: https://www.opennetworking.org/standards/of-config-10

[6] R. Enns (Ed), "NETCONF configuration protocol," Internet Engineering Task Force, RFC 4741, Dec. 2006.

[7] R. Enns, M. Bjorklund, J. Schoenwaelder, and A. Bierman (Eds), "Network configuration protocol (NETCONF)," Internet Engineering Task Force, RFC 6241, Jun. 2011.

[8] M. Bjorklund (Ed), "YANG - a data modeling language for the network configuration protocol (NETCONF)," Internet Engineering Task Force, RFC 6020, Oct. 2010.

[9] R. Sherwood, G. Gibb, K. Yap, G. Appenzeller, N. McKeown, and G. Parulkar, "FlowVisor: A network virtualization layer," OpenFlow Switch Consortium, Tech. Rep. OPENFLOW-TR-2009-1, 2009. [Online]. Available: http://www.openflow.org/wk/index.php/FlowVisor

[10] J. Naous, "Expedient: A pluggable platform for geni control framework." [Online]. Available: http://yuba.stanford.edu/~jnaous/expedient/

[11] GENI project contributors, "FOAM." [Online]. Available: http://groups.geni.net/geni/wiki/OpenFlow/FOAM

[12] J. Smift, "GENI, openflow update – FOAM," presentation slides, Nov. 2011, Kansas City, Missouri.

[13] OpenFlow Consortium contributors, "Opt-in manager." [Online]. Available: http://www.openflow.org/wk/index.php/OptIn_Manager

[14] J. Czentye and B. Sonkoly, "FlowVisor module for OpenNMS." [Online]. Available: http://sb.tmit.bme.hu/mediawiki/index.php/OpenNMS_FlowVisor

[15] "OpenNMS." [Online]. Available: http://www.opennms.org/

[16] B. Sonkoly, A. Gulyás, J. Czentye, K. Kurucz, G. Vaszkun, A. Kern, D. Jocha, and A. Takács, "Integrated OpenFlow virtualization framework with flexible data, control and management functions," in *Proceedings of IEEE INFOCOM 2012 (Demo)*, Orlando, Florida, USA, 2012.

[17] "Yuma: a YANG-based unified modular automation toolkit." [Online]. Available: http://www.Netconfcentral.org/yuma

[18] NEC, "Ip8800/s3640 software manual, openflow feature guide (version 11.1 compatible)," NEC, Tech. Rep. NWD-105490-001, May 2010.

[19] B. Sonkoly, A. Gulyás, F. Németh, J. Czentye, K. Kurucz, B. Novák, and G. Vaszkun, "OpenFlow virtualization framework with advanced capabilities," in *Proceedings of European Workshop on Software Defined Networks (EWSDN)*, Darmstadt, Germany, 2012.