OFLOPS-Turbo: Testing the Next-Generation OpenFlow switch

Charalampos Rotsos*¶, Gianni Antichi*, Marc Bruyere†‡§, Philippe Owezarski†‡, Andrew W. Moore*
¶ *School of Computing and Communications, Infolab 21, Lancaster University*
**Computer Laboratory, University of Cambridge*
†*Université de Toulouse* ‡*CNRS, LAAS, France* §*DELL Inc.*

***Abstract*—The heterogeneity barrier breakthrough achieved by the OpenFlow protocol is currently paced by the variability in performance semantics among network devices, which reduces the ability of applications to take complete advantage of programmable control. As a result, control applications remain conservative on performance requirements in order to be generalizable and trade performance for explicit state consistency in order to support varying performance behaviours. In this paper we argue that network control must be optimized towards network device capabilities and network managers and application developers must perform informed design decision using accurate switch performance profiles. This becomes highly critical for modern OpenFlow-enabled 10 GbE optical switches which significantly elevate switch performance requirements. We present OFLOPS-Turbo, the integration of the OFLOPS switch evaluation platform, with the OSNT platform, a hardware-accelerated traffic generation and capture system supporting lossless 10 GbE functionality. Using OFLOPS-Turbo, we conduct an evaluation of flow table manipulation capabilities in a representative collection of 10 GbE production OpenFlow switch devices and interpret the evolution of OpenFlow support by comparison with historical data.**

***Keywords*-SDN; OpenFlow; Open–Source; High Performance; Testing; NetFPGA.**

I. Introduction

Research on SDN technologies and primarily its predominant realisation, the OpenFlow protocol, has developed a wide range of applications to improve network functionality. OpenFlow control applications can improve network management, monitoring and performance, while being backwards-compatible with data plane protocols and end-host network stacks. As a result, within only a few years since the definition of the first version of the protocol, many vendors have introduced production-level support in an effort to transfer innovative research output to the market.

Nonetheless, such continuous network innovation introduces a dilemma for network testing, as relevant evaluation platforms remain closed and proprietary and provide limited flexibility. To achieve compliant and functional equipment, effort must be put into all parts of the network-equipment life-cycle, from design to production. The problem of network testing is further augmented in the OpenFlow protocol context. The protocol philosophy introduces new performance challenges in network device design, dissimilar to the challenges of traditional network equipment, and sets testing flexibility as a primary requirement. OpenFlow introduces a reduction in the network control timescales which closely approximate flow control timescales. As a result, OpenFlow switch implementation influences the control architecture of the network and its overall performance. A switch which provides poor performance in the support of a protocol functionality can become a bottleneck for architectures that rely heavily on the specific functionality; *e.g.,* high latency in `flow_stats_reply` functionality of a switch can be critical for traffic monitoring control applications. Protocol support variability is equally critical for the consistency of the control plane. Most switches do not provide any guarantees on the installation order and latency of a sequence of flow table updates; as a result, such updates need to be carefully sequenced to not violate policy [8], effectively increasing the insertion latency.

The OpenFlow protocol, currently defining its 1.5 version, has greatly transformed its design over the years, reflecting the deployment experience and requirements of a constantly widening range of network environments. As a result, the OpenFlow community *requires* a performance testing platform capable to co-evolve with the protocol and to support rapid prototyping of experimentation scenarios which highlight the impact of new protocols' features. Furthermore, the increase in link capacity augments the precision requirement for meaningful packet-level measurements. For example, 10 GbE links have become the de-facto solution for the aggregation layer of modern datacenter networks, a first-class citizen of the SDN ecosystem, and require high measurement precision, on the order of sub-μsec, for certain network application classes. An estimation error of 100μsec in the policy enforcement of a security application may translate to unauthorized transmission of multiple KBs of sensitive information. This paper claims that in order to fully exploit OpenFlow protocol capabilities in production environments, we require a flexible and high–precision open–source measurement platform. The openness and flexibility is important in order to establish an evolvable community–based tool.

This paper presents an effort to enhance the measurement capabilities of the OFLOPS [15] switch evaluation framework with support for the emerging protocol requirement. Specifically, we present the OFLOPS integration with the NetFPGA-10G platform[1] and the Open Source

[1] http://www.netfpga.org

Network Tester (OSNT)[2] platform [1]. OSNT enhances OFLOPS with sub-μsec precision and 20 Gbps full bi-directional traffic generation and capturing (when traffic thinning techniques are being used), providing a highly flexible and open platform for OpenFlow experimentation at high data rates.

For the rest of this paper, we initially motivate the OFLOPS-Turbo design with a set of use cases (§ II), followed by a detailed presentation of the system design (§ III). Furthermore, we present an evaluation of the flow table manipulation capabilities for a representative set of 10 GbE OpenFlow switches (§ IV), discuss related work (§ V) and conclude our work (§ VI).

II. Use Cases

Programmable control is considered the holy grail for a number of common network research problems. In order to motivate the discussion for switch performance characterisation, this section presents a set of SDN applications and discusses their performance requirements.

A. White-Box dilemma

White-box Ethernet switches [7] evolve into a compelling network device solution for a wide range of network environments, primarily due to their enhanced flexibility. White-box networking refers to the ability to use generic, off-the-shelf switches and routers in the forwarding plane of a network. Such devices are open to different OSes, while OpenFlow protocol, implemented through OF-agents, is a popular control abstraction among them. The majority of white-box switch OSes are Linux-based, primarily because of their support for a wide range of CPU architectures and free tools. Nonetheless, different realisations of the same standard commonly optimise different performance aspects. In the case of white-box switches, although vendors use the same chipset (most 48x10Gbps + 4x40Gbps 1RU switch use the Broadcom Trident chipset [3]), they employ a diverse device design approach (CPUs, memory, implementation). Technology adopters require flexible and easily configurable mechanisms to develop comparison studies between combinations of white-box switches (*i.e.*, OSes and OF-agents). This is helpful to acquire a deep understanding of the resulting switch capabilities and achieve a better fit with the requirement of their network architecture and environment.

B. Towards an Iperf-like OpenFlow test platform

The compelling simplicity and generality of the OpenFlow abstraction pushes its adoption across all devices of the network, in an effort to converge network control across the network. For example, the ALIEN project [13] develops a version-oblivious Hardware Abstraction Layer for OpenFlow support, targeting a wide and diverse range of hardware platforms, such as user-space x86-based routers, Broadcom chipsets, EZ-CHIP and Octeon network processors and NetFPGA 10G FPGAs. In order to ensure a common performance comparison "vocabulary", the SDN community requires a high-precision performance testing platform, which will establish a common ground between different implementations through standardised evaluation galleries, similar to the OFTest [12] compatibility testing platform.

[2]http://www.osnt.org

C. IXP: Internet eXchange Point

Internet eXchange Points (IXP) are a special network infrastructure, providing inter-AS peering in a single location. IXPs must support high data plane performance and in parallel provide programmability, advanced traffic monitoring capabilities and scalability for hundreds of peers. The introduction of the SDN paradigm provides unprecedented opportunities to improve the performance of such network environments using commodity network devices [9]. IXP service providers, interested to invest in infrastructural SDN/OpenFlow support, require tools which allow them to identify the ultimate performer switch for their needs. Furthermore, to facilitate the migration of the IXP to a programmable control paradigm, network device vendors design switches with hybrid capabilities and concurrent support for legacy control on a subset of the Ethernet interfaces and OpenFlow control on the rest. Nonetheless, hybrid switches have limitations, primarily deriving by the CAM configuration in order to abstract under a common flow abstraction multiple forwarding tables with variable matching capabilities. For example, a hybrid switch must provide matching table support for the complete OpenFlow tuple, along with support for FIB and ACL lookup tables. Research on SDN-based IXP control architectures focuses on the network controller and demonstrates fast processing of sophisticated routing policies and scalability for hundreds of IXP members. Nonetheless it remains equally important to understand how an OpenFlow-enabled switching fabric behave during, for example, large bursts of flow table modifications.

III. OFLOPS-Turbo design

Measuring OpenFlow switch implementations is a challenging task in terms of characterisation accuracy and precision. Predominantly, production-level OpenFlow-switch devices provide proprietary OpenFlow drivers with minimum logging and instrumentation capabilities. Characterising the performance of an OpenFlow-switch requires a black box approach to the problem, where multiple input measurements must be monitored concurrently in order to characterise the behaviour of the switch.

A schematic of the OFLOPS-Turbo design is depicted in Figure 1. The OFLOPS-Turbo architecture consists of a software and a hardware subsystem. The software subsystem runs the core OFLOPS functionality, along with the test of the user. A test contains both the control and data plane logic of the experiment. The hardware subsystem consists of a series of NetFPGA-10G cards running the OSNT design and is responsible to fulfil the data plane requirements of the experiment. The user can interconnect the OFLOPS-Turbo host with one or more

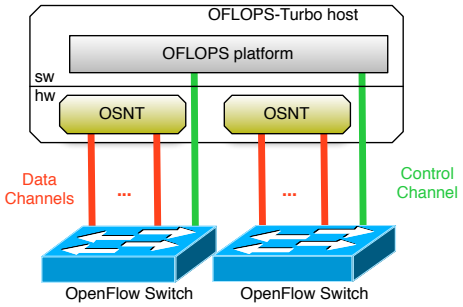

Figure 1. OFLOPS-Turbo design

switches in arbitrary topologies and measure with high precision specific aspects of the network architecture, both on the data and control plane. For the rest of the section, we describe the design of the OSNT hardware design and the OFLOPS software architecture.

A. OSNT: Open Source Network Tester

The OSNT is a fully open–source traffic generation and capturing system. Its architecture is motivated by limitations in existing network testing solutions: closed-source/proprietary, high costs, inflexibility, and lack of important features such as high-precision timestamping and packet transmission. Primarily designed for the research and teaching community, its key design goals were low cost, high-precision time-stamping and packet transmission, as well as, scalability. In addition, the open–source nature of the system gives the flexibility to allow new protocol tests to be added to the system. The prototype implementation builds upon the NetFPGA-10G platform – an open-source hardware platform designed to support full line-rate programmable packet forwarding. The combination of Traffic Generator and Traffic Monitor subsystems into a single FPGA-equipped device allows a per–flow characterisation of a networking system (*i.e.*, OpenFlow-enabled switch) within a single card. Using one or more synchronised OSNT cards, the architecture enables a user to perform measurements throughout the network, characterising aspects such as end-to-end latency and jitter, packet-loss, congestion events and more.

The OSNT *Traffic Capture* subsystem is intended to provide high–precision inbound timestamping with a loss-limited path that gets (a subset of) captured packets into the host for further processing. The design associates packets with a 64-bit timestamp on receipt by the MAC module, thus minimising queueing noise. The timestamp resolution is 6.25 nsec with clock drift and phase coordination maintained by a GPS input. The OSNT *Traffic Generation* subsystem provide a PCAP replay function with a tunable per–packet inter–departure time. The traffic generator has an accurate timestamping mechanism, located just before the transmit 10GbE MAC. The mechanism, identical to the one used in the traffic monitoring unit, is used for timing-related measurements of the network, permitting characterisation of measurements such as latency and jitter. When enabled, the timestamp is embedded within the packet at a preconfigured location and can be extracted at the receiver as required.

B. OFLOPS: Open Framework for OpenFlow Switch Evaluation

OFLOPS is an holistic measurement platform which enables the development of custom OpenFlow-based experiments. The platform provides a unified API that allows developers to control and extract information from the data and control channels, as well as, SNMP switch-state information. Experimenters can develop measurement modules on top of OFLOPS, implementing custom OpenFlow applications and measure their performance through the data plane. Currently OFLOPS provides a wide range of elementary testing modules evaluating the performance of flow actions, flow table management, flow counter extraction, OpenFlow-based packet injection and capturing and detecting potential performance penalties due to OpenFlow operation co-interaction.

OSNT support for 10 GbE traffic generation and capturing and GPS-corrected hardware timestamps enhances significantly the precision of the switch performance characterisation task and extend the ability to evaluate SDN deployment scenarios in network environments with high traffic rates. In comparison to the earlier 1G NetFPGA hardware design, the OSNT platform improves significantly the packet generation and capturing capabilities. The earlier OFLOPS NetFPGA 1G hardware design exhibited limited traffic generation capabilities, restricted primarily by the DRAM memory module read speed, while the packet capture capability was limited by the design choice to integrate the driver with the Linux network stack. As a result, the 1G traffic generation is measured to achieve up to 800 Mbps traffic, when using 100-byte packets, while lossless traffic capture was limited to 100 Mbps. The OSNT platform has been tested and can generate up to 20 Gbps traffic using small packets, while the traffic capturing subsystem provides applications with direct access to the DMA engine of the card and thus can achieve lossless 20 Gbps traffic capturing using traffic thinning techniques (*i.e.*, record a fixed-length part of each packet) [1]. In addition, OSNT provides an enhanced timestamp accuracy using a clock drift and phase coordination module maintained by a GPS input.

The integration between OFLOPS and OSNT is achieved through a C library[3], which exposes a traffic generation and capture interface, as well as access to card statistics (*e.g.,* counter for packet captures and drops). The library is designed to exploit the zero copy DMA engine of the OSNT driver and optimizes the required CPU operations to transfer a packet from the kernel to the process address space, in order to support high throughput measurements in 10 GbE links.

IV. Performance Evaluation

In this section we present our experimental setup and the ensemble of tested switch (§ IV-A), preceded by

[3]https://github.com/OFLOPS-Turbo/nf-pktgencap-lib

an analysis of the measurement improvement achieved by OFLOPS-Turbo (§ IV-B) and an evaluation of the flow table manipulation subsystem in 10 GbE production OpenFlow switches (§ IV-C).

A. Experimental setup

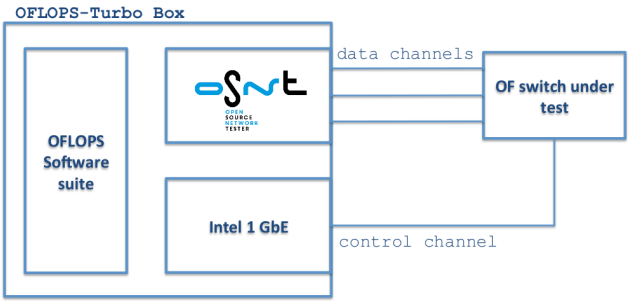

Figure 2. The proposed testbed.

In this paper we use OFLOPS to evaluate two representative 10 GbE switches: the *Pica8 P3922*; and the *Dell Force10 S4810*. The Pica8 P3922 switch [14] is a 48 ports 10 GbE SFP+ and 4 ports 40 GbE QSPF+ port switch. The switch provides two operational modes: *L2/L3* and *OVS* mode. L2/L3 mode provides hybrid OpenFlow support along with other standardised data link and network control protocols, like OSPF and RIP. OVS mode functions as a standalone Open VSwitch switch with a forwarding table mapped to the TCAM table of the switch using a custom device driver. Vendor specifications do not define the maximum flow table size supported by the silicon, but we have successfully tested fast-path data plane support for up to 1000 unique flows in OVS mode. The Pica8 L2/L3 mode allows users to define the flow table size through the switch user interface and can be configured to support up to 500 flows. The Dell Force10 S4810 [6] switch is a 48 10 GbE QSFP+ ports. Its firmware supports hybrid OpenFlow functionality and exposes three hardware tables: a 512 wildcard entry table supporting full OpenFlow tuple matching, a 20000 entry table supporting only destination MAC address matching and a 6000 entry table support IP source and destination matching. Both switches use a similar StrataSGX trident silicon version [3]. The two switches reflect two popular approaches in OpenFlow support: mixing protocol functionality with legacy protocols; or developing a single-purpose OpenFlow-optimised firmware.

We setup OFLOPS-Turbo in a dual socket 8-core Intel Xeon E5 server with 128 Gb RAM memory, equipped with a NetFPGA 10G card running the OSNT design and an Intel 1 GbE card for the control channel. Three ports of the NetFPGA card are connected directly to the measured switch, while the switch control channel uses the switch management port and connects directly to the 1 GbE card, similarly to the topology in Figure 2. During experiments we connect the NetFPGA card with a GPS receiver, to correct its time reference and provide nanosecond measurement precision. In addition, in order to establish a common time reference between the control and data plane traffic of the measurement, the control channel is replicated to the fourth port of the NetFPGA card through an optical wiretap with negligible processing delay.

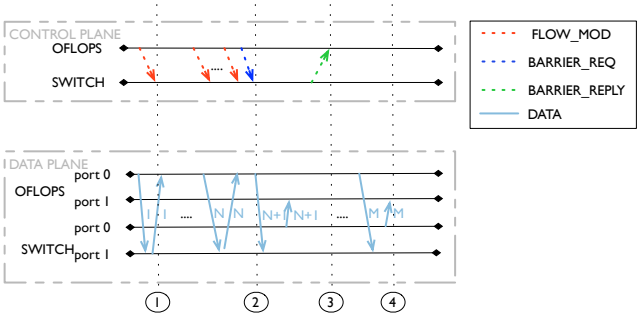

Figure 3. Flow insertion measurement scenario sequence diagram.

B. Flow table measurement precision

In order to evaluate switch flow manipulation capabilities we use the respective test from the OFLOPS code base, depicted in Figure 3. During each experimental run, the measurement module initialises the switch flow table by removing all entries and inserting a batch of flow entries which redirect the data plane measurement probe to a specific port, matching the packet destination IP addresses in the measurement probe. The module sets only required fields in the flow modification message and does not enable any option. In parallel, the test configures the OSNT design to generate a measurement probe from the first port (switch port 0) of the NetFPGA card at a controlled rate, targeting the measured flows in a round-robin manner. Flow table initialisation differs depending on whether the experiment measures flow addition or flow modification support. Flow addition experiments insert a single wildcard entry, while flow modification experiments insert unique flow entries with distinct destination IP addresses. In both cases, the inserted flows forward all incoming packet from port 0 of the switch to `IN_PORT` with a fixed priority (10) (Step 1). After a fixed warmup time, the module sends a parametrized batch of distinct `flow_mod` messages, equal to the number of flows generated by the traffic generator and matching the respective destination IPs, with higher priority (11), which forward matching packets to switch port 1, together with a `barrier_req` message (step 2). The module measures the delay to receive the barrier reply (step 3), as well as the delay to receive at least one packet from each flow on the second port of the NetFPGA card (step 4). Without loss of generality, the module uses version 1.0 of the OpenFlow protocol to perform switch flow table manipulations.

As described, the measurement scenario effectively uses data plane packets as a sampling process to evaluate when a flow becomes active in the flow table of the switch. A change in the output port between two consecutive packets of the same flow signifies that a new flow has become active on the switch flow table during the time period

Throughput (Gbps)	0.1	1	5	8.8
Inter-packet gap (μs)	120	12	2.4	0
Latency (μs)	91689	75524	44886	40301

Table I
EVALUATION OF THE INSERTION DELAY FOR 100 FLOWS USING DIFFERENT PACKET PROBE RATES.

between the transmission of the two packets. Nonetheless, because we cannot identify the exact modification time within this period, the measurement is subject to an over-estimation of the table manipulation latency. The support of OSNT for high rate traffic generation and capturing allows OFLOPS to improve the granularity of the measurement by reducing the latency between consecutive packet with the same destination IP and thus improve estimation error. For example, to measure the insertion delay of 100 flows using a 100 Mbps probe requires an IPG of 0.12 msec. The precision though of this experiment is lower bound by the delay between packet with the same destination IP address, which for the specific example is 1.2 msec IPG, since traffic is send in a round-robin manner with respect to the inserted flows. If the same experiment is contacted using a faster measurement probe at 1 Gbps, then the delay between consecutive packets of the same flow is reduced by an order of magnitude, improving effectively the precision of the latency measurement.

Table I presents the estimated median flow insertion delay of 100 flows for different traffic generation rates, using the OVS mode of the Pica8 switch. The median is calculated across 20 runs of the experiment. The table highlights the improvement in the measurement of the flow table modification latency using probes with higher packet rates. We need to point out, that probes using small packets achieve data rates which are noticeably lower that the link capacity (*e.g.,* a measurement probe using packet of 100 bytes can achieve a maximum throughput of 8.82 Gbps on a 10 GbE link). This is due to the per-packet synchronisation overheads (*e.g.,* preamble) imposed by the Ethernet MAC layer.

C. Switch Evaluation

Using the table manipulation experiment scenario we evaluate the performance of the two aforementioned switches. For the Pica8 switch we consider both operational modes; OVS and L2/L3. Each measurement uses an 8.82 Gbps measurement probe of small packet (150 bytes) and variates the number of installed flows. Figure 4 presents the scalability of flow insertions, while Figure 5 presents the scalability of flow modifications. For the Force10 switch and the Pica8 L2/L3 mode we restrict the measurement up to 500 flows, the maximum supported number of wildcard flows, while for the Pica8 OVS mode we extend the measurement up to 1000 flows.

From the results we highlight three observations. Firstly, flow modifications exhibit a significantly lower latency in comparison to flow additions for flow batch sizes larger than 200 flows. This performance difference can be explained by the potential lower number of required

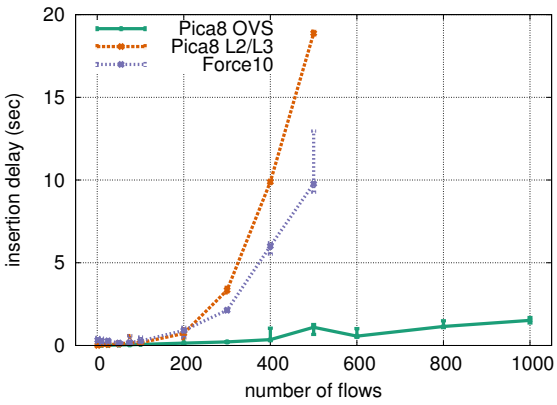

Figure 4. Evaluation of flow table insertions using OFLOPS.

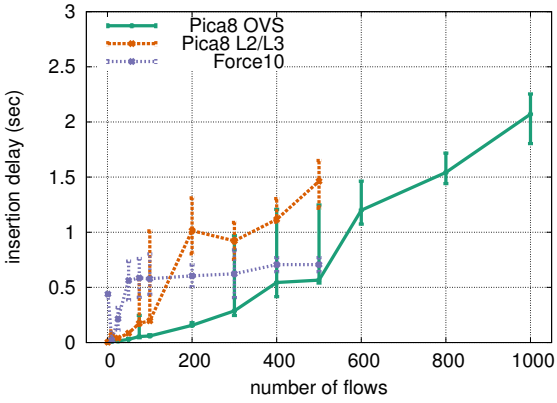

Figure 5. Evaluation of flow table modifications using OFLOPS.

memory manipulations between the silicon and the OF-agent (*i.e.*, each flow modification requires an update of the action part of the flow only). Secondly, our measurement results highlight a significant performance difference between the hybrid firmwares and the single-purpose OVS mode, for more than 100 flows. From the analysis of the per-flow delay distribution for the hybrid firmwares, we observed that beyond 100 flows the switch exhibits a pacing behaviour, limiting per-second flow insertions. The Pica8 OVS mode exhibits a dissimilar behaviour achieving the best performance among tested switches. Finally, in comparison to the results obtained in [15], previous generation 1 GbE switches exhibit a performance lying between the hybrid switches and the Pica8 OVS mode. For the insertion of a single flow a 1 GbE switch requires 5 msec, while Pica8 OVS requires 80 msec and Force10 requires 300 msec. For the insertion of 500 flows a 1 GbE switch requires 1.5 sec, Pica8 OVS mode requires 1.1 sec, while Force10 requires 9.8 sec.

V. RELATED WORK

OpenFlow is a popular standard. It is increasingly adopted by vendors and the research community as it allows direct access and manipulation of the forwarding plane of network devices such as switches and routers, both physical and virtual (*i.e.*, hypervisor-based). To the best of our knowledge, OFLOPS [15] was the first attempt for an OpenFlow switch evaluation platform.

Bianco *et al.* [2] focus on the data plane performance advantage of software OpenFlow over the Linux Ethernet switch, while Curtis *et al.* [5] discuss the switch specific limitations to support OpenFlow functionality in large network deployments. Jarschel *et al.* [11] discuss a model for both forwarding speed and blocking probability of OpenFlow devices. None of the previous solutions provide a general purpose programmable benchmarking platform, similar to OFLOPS. Canini *et al.* [4] propose NICE, an efficient, systematic mechanism for control application testing. While NICE focuses primarily on the OpenFlow application perspective, we are more interested in benchmarking the actual OpenFlow switch implementation.

Finally, Huang *et al.* [10] benchmark OpenFlow-enabled switches and illustrate how their implementation can dramatically impact data plane latency and throughput. They also present a measurement methodology and emulator extension to reproduce these control-path performance characteristics, restoring the fidelity of emulation. While the authors focus primarily on vendor-specific variations in the control plane, we try to benchmark both control and data plane taking advantage of the OSNT system.

VI. Conclusions

In this paper we have presented OFLOPS-Turbo, an open and flexible OpenFlow testing framework for the next-generation of OpenFlow switches. OFLOPS-Turbo takes advantage of the OSNT hardware design for the NetFPGA-10G platform and provides support for 10 GbE traffic generation and capture, coupled with a high precision timestamping functionality. Using OFLOPS-Turbo we presented the enhanced precision capabilities provided by the OSNT platform and evaluated the flow table manipulations for two production 10 GbE switches, highlighting the limitations of hybrid switch firmwares, as well as, the low control plane performance improvement in comparison to 1 GbE switches. We believe that OFLOPS-Turbo will provide the tool for the OpenFlow community to better understand the performance impact of OpenFlow implementations and motivate a community of rigorous OpenFlow testing.

Acknowledgements

This work was jointly supported by the EPSRC INTERNET Project EP/H040536/1, the EPSRC TOUCAN Project EP/L020009/1 and the Defense Advanced Research Projects Agency (DARPA) and the Air Force Research Laboratory (AFRL), under contract FA8750-11-C-0249. The views, opinions, and/or findings contained in this article/presentation are those of the author/ presenter and should not be interpreted as representing the official views or policies, either expressed or implied, of the Defense Advanced Research Projects Agency or the Department of Defense.

References

[1] G. Antichi, M. Shahbaz, Y. Geng, N. Zilberman, A. Covington, M. Bruyere, N. McKeown, N. Feamster, B. Felderman, M. Blott, A. Moore, and P. Owezarski. OSNT: Open Source Network Tester. *IEEE Network*, Special issue on Open Source for Networking: Tools and Applications, 2014.

[2] A. Bianco, R. Birke, L. Giraudo, and M. Palacin. OpenFlow switching: Data plane perfommance. In *Internation Conference on Communications (ICC)*. IEEE, 2010.

[3] High-Capacity StrataXGS Trident Ethernet Switch Series with Integrated 10G Serial PHY - BCM56840 Series. http://www.broadcom.com/products/Switching/Carrier-and-Service-Provider/BCM56840-Series, 2014.

[4] M. Canini, D. Venzano, P. Perešíni, D. Kostić, and J. Rexford. A NICE way to test OpenFlow applications. In *Conference on Networked Systems Design and Implementation (NSDI)*. USENIX, 2012.

[5] A. R. Curtis, J. C. Mogul, J. Tourrilhes, P. Yalagandula, P. Sharma, and S. Banerjee. DevoFlow: Scaling flow management for high-performance networks. In *ACM SIGCOMM*. ACM, 2011.

[6] Dell networking s4810. http://i.dell.com/sites/doccontent/shared-content/data-sheets/en/Documents/Dell_Force10_S4810_Spec_sheet.pdf, 2014.

[7] S. Garrison and S. Crehan. The rise of white-box switches. http://www.infoworld.com/t/sdn/the-rise-of-white-box-switches-231246, Nov. 2013.

[8] A. Guha, M. Reitblatt, and N. Foster. Machine-verified network controllers. *SIGPLAN Not.*, 48(6), June 2013.

[9] A. Gupta, L. Vanbever, M. Shahbaz, S. P. Donovan, B. Schlinker, N. Feamster, J. Rexford, S. Shenker, R. Clark, and E. Katz-Bassett. SDX: A software defined internet exchange. In *ACM SIGCOMM*. ACM, 2014.

[10] D. Y. Huang, K. Yocum, and A. C. Snoeren. High-fidelity switch models for software-defined network emulation. In *ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking (HotSDN)*. ACM, 2013.

[11] M. Jarschel, S. Oechsner, D. Schlosser, R. Pries, S. Goll, and P. Tran-Gia. Modeling and performance evaluation of an OpenFlow architecture. In *International Teletraffic Congress (ITC)*. International Teletraffic Congress, 2011.

[12] OFTest. *http://www.projectfloodlight.org/oftest. Last visited: 18-05-2014.*

[13] D. Parniewicz, R. Doriguzzi Corin, L. Ogrodowczyk, M. Rashidi Fard, J. Matias, M. Gerola, V. Fuentes, U. Toseef, A. Zaalouk, B. Belter, E. Jacob, and K. Pentikousis. Design and implementation of an OpenFlow hardware abstraction layer. In *ACM SIGCOMM Workshop on Distributed Cloud Computing (DCC)*. ACM, 2014.

[14] Pica8 p3922 specifications. http://www.pica8.com/documents/pica8-datasheet-64x10gbe-p3922-p3930.pdf, 2014.

[15] C. Rotsos, N. Sarrar, S. Uhlig, R. Sherwood, and A. Moore. Oflops: An open framework for OpenFlow switch evaluation. In *Passive and Active Measurement (PAM)*, 2012.