Noname manuscript No.
(will be inserted by the editor)

OpenFlow Data Planes Performance Evaluation

Leonardo C. Costa · Alex B. Vieira · Erik de Britto e Silva · Daniel F. Macedo · Luiz F. M. Vieira · Marcos A. M. Vieira · Manoel da Rocha Miranda Junior · Gabriel Fanelli Batista · Augusto Henrique Polizer · André Vinícius Gomes Santos Gonçalves · Geraldo Gomes · Luiz H. A. Correia

Received: date / Accepted: date

Abstract Software-Defined Networks, in its essence, is the separation of the data and control planes of switching devices. The OpenFlow (OF) protocol is the most popular SDN protocol today, being available in many switches. This is due to the low implementation cost as well as the potential for innovative solutions in the network. Although OF is being used in many research papers and production networks, as far as we know, no work on the literature performs an extensive evaluation of the OpenFlow data planes. This evaluation helps network administrators choose which switch to use in their networks. Meanwhile, researchers are made aware of the limitations of existing OF switches. This article evaluates the performance and maturity of OF 1.0 on eleven hardware and software switches using POX, and also of the newer OF 1.3 on five switches using the ONOS controller. Our findings indicate that the performance variations among OF switches are significant. Packet delays vary by one order of magnitude in the evaluated equipment. Meanwhile, there is no performance impact when changing the packet size. Hence, the results highlight that researchers must be aware of issues such as variable jitter, the number of matches in tables as well as missing OF match fields.

Leonardo C. Costa, Alex Borges Vieira
Computer Science Department
Universidade Federal de Juiz de Fora, Juiz de Fora, Brazil
E-mail: alex.borges@ufjf.edu.br

Erik de Britto e Silva, Daniel F. Macedo, Luiz F. M. Vieira, Marcos A. M. Vieira, Manoel da Rocha Miranda Junior, Gabriel Fanelli Batista, Augusto Henrique Polizer
Computer Science Department
Universidade Federal de Minas Gerais, Belo Horizonte, Brazil

Augusto H. Polizer, André V. G. S. Gonçalves, Geraldo Gomes, Luiz H. A. Correia
Computer Science Department
Universidade Federal de Lavras, Lavras, Brazil

Keywords OpenFlow, Software-Defined Networks, data plane, performance evaluation

1 Introduction

Software-Defined Networking (SDN) is a technology that is reshaping the way networks are operated and developed [12]. SDN simplified the creation of new networking solutions, mainly because of the separation of the data and control planes. The key benefits of this are the fast deployment of new services, which drive down the costs of network operators. Because of that, SDN is becoming a key architecture component on 5G systems [10] and Wireless Enterprise Networks [24]. SDN can even be used to support matching rules with domain names [36]. Furthermore, researchers are now able to perform tests in real environments, without affecting the traffic or the availability of production networks [12]. In this context, there is no doubt that the SDN paradigm attracted attention from both academia and industry.

There are several sophisticated SDN implementations, such as P4 (Programming Protocol-independent Packet Processors) and POF (Protocol oblivious Forwarding) [5, 37]. However, OpenFlow is the most popular standard in production networks and research. This occurs because vendors can implement OpenFlow without significant difficulties.

Despite being widely employed in industry and academia, as far as we know, no work has evaluated of the existing OpenFlow data planes. Such a study is important for operators because it provides an assessment that may guide the deployment of OpenFlow on their network. For researchers, this is useful to grasp the limitations of OpenFlow in current hardware. With this information in mind, better emulators and simulators can be built, by representing the performance and features of those switches more realistically . Many papers evaluate the performance of the OpenFlow Controllers [1, 4, 16, 38], however, most of them do not measure how the implementation of the data plane affects the network performance. Only [1, 4, 38] evaluate the data plane, however, their work does not measure the data plane separately from the control plane. It is important to properly investigate the data plane because the performance of the controller only affects the first packet in a flow. The data plane, however, will define the performance for the entire flow.

Given this context, this article evaluates the performance and maturity of the main features of OpenFlow 1.0 and 1.3 on both hardware and software switches. We consider a wide number of switches, varying from various off-the-shelf equipment to open source implementations of software switches running on PCs or embedded platforms (e.g. home routers). In a controlled environment, we systematically evaluate the implementation of the OpenFlow data planes by: (i) measuring the performance of the switch in terms of latency and jitter, (ii) assessing how the OpenFlow mode compares to the legacy L2 switching mode, and (iii) evaluating how OpenFlow operations (such as rule

matches, packet rewrites, flowstats, and packetstats reports) perform. This article extends our previous work [8][1] by:

- Adding OpenFlow 1.3 performance evaluation experiments;
- Adding one more commercial switch to the comparison of OpenFlow 1.0 switches (Zodiac FX);
- Testing a highly optimized version of Open vSwitch using the Intel DPDK suite of Linux kernel modifications, to understand how OS optimizations may improve the performance of software switches;
- Adding high-performance switch evaluation;

Our results show that OpenFlow's performance varies significantly. For example, packet delays vary by one order of magnitude among the evaluated switches, while the performance is not affected by the packet size. Moreover, we also note that some hardware switches perform as well as software switches, which may indicate a software implementation within the hardware. In sum, our article presents a glimpse into how OpenFlow is currently implemented on real devices. This is important for network administrators and academics to understand how OF switches work, allowing them to pick the best switch for their network.

The remainder of this article is structured as follows. Section 2 presents related work. Section 3 describes the evaluation methodology. Sections 4 and 5 discuss the results for OpenFlow 1.0 and OpenFlow 1.3, respectively. Section 6 concludes the article and presents future work.

2 Related Work

OpenFlow [23] is an open protocol in which a central element can inspect and modify the flow tables of the switching elements. This is possible via a standard API, where a computer (the control plane) sends queries and configuration messages to the network equipment (data plane) [12, 35]. OpenFlow is the most popular SDN platform in academia and industry [21]. Companies such as HP, NEC, Pronto, Extreme, Cisco, Brocade, Juniper, and Huawei already sell OF-capable devices [12]. Thus, it is important to measure the performance of OpenFlow-enabled switches.

This section presents the state of the art on OpenFlow evaluation, which can be divided into categories. First, we highlight the studies that performed evaluation campaigns. Then, we tackle the works defining tools and frameworks for performance evaluation, followed by analytical models of OF performance. Finally, we map how the data plane performance influences the design of new OpenFlow features.

[1] Paper available in `http://netlab.ice.ufjf.br/index.php/OFperformance/`

2.1 Empirical Evaluation of OpenFlow Performance

Bianco et al. [4] analyzed one virtual switch working in bridge mode. The authors compare the performance of this switch in OpenFlow mode with the performance of the same switch operating in legacy L2 switch mode, IP routing, and Ethernet switching use cases. Despite that, no hardware-based switch was used in the experiments. Further, the performance results do not isolate the switch's performance from the controller performance.

Sünnen et al [38] also evaluate the performance of a single OpenFlow hardware switch. Some of the evaluation metrics used depend heavily on the controller's performance, which is not the focus of our work. In the present work, the data plane performance is independent of the controller employed because all the rules are written before the beginning of the experiment flows.

In turn, Pfaff et al. [31] perform a comparison between three different OpenFlow switches, consisting of two hardware switches and a software Open vSwitch version. Again, the evaluation metrics also considered the influence of the controller.

Osman et al. [29] evaluated how a lossy control link (e.g. wireless networks) affects the performance of OF switches. They have shown that the performance of OF switches is fairly resilient to a high packet loss by using a hybrid system in which the control can be delegated to local controllers when the link reliability to the central controller is degraded. Anew, no hardware-based switch was evaluated and authors did not evaluate the data plane performance. Authors only show that data plane can achieve reasonable performance, even under a high error rate control link.

Mallon et al. [22] evaluated the capacity of OF controllers to handle a large number of flows, identifying which system design aspects contribute to the performance of the control plane. The authors show that performance bottlenecks arise due to the object-oriented nature of controllers, the use of global data structures that are changed by many threads/cores, as well as the inefficient use of caching and memory copies.

Silva et al. [9] used existing SDN switches to evaluate and compare five load balancing policies. Their results showed that the txbytes(transmitted bytes) and cpuq-load(CPU usage and number of open connections) policies outperformed the others. Instead of evaluating load balancing policies, in this work we evaluate the switches.

Kuźniar et al. [18] evaluate the performance of five hardware switches during table update operations. The authors show that a batch of updates significantly delays the processing of incoming data packets, so the controller should only send a few requests at a time. Further, many of the evaluated switches show performance degradations when the number of flow rules increases. Their work is very similar to ours, however, in this article, we focus on the performance of the flows mostly after the rules have been installed.

2.2 Evaluation methodologies and frameworks

Frameworks and methodologies to measure the performance of switches, in special OF switches, have already been proposed. Rotsos et al. [34] propose an open and generic framework to test OpenFlow devices. Their tool evaluates how the controller interacts with the OF switches. However, the results do not separate the contribution of each plane to the final performance.

OFBench is an automatic test suite that evaluates the performance of OF data planes [19]. It measures the performance of user flows passing through the switch (e.g. the processing delay of match/modify actions), as well as the performance of OF flowmod operations on the switch (the size of the packet in a buffer in the switches and the precision of the hard and soft timeouts in OpenFlow). The tool was evaluated using a software switch (Open vSwitch) and four hardware switches but it evaluates how the controller actions affect performance. This related work is the most similar to ours and focuses on the design of the tool, instead of performing an extensive measurement effort. Their evaluation employs similar performance metrics like those in this article, such as the performance of switches with multiple table flows and the time to perform rule modifications. However, the present work presents additional measurements, such as the performance of OF statistics operations, a comparison with L2 legacy switching, the maximum number of rules, and performance for single and multiple flows.

Spirent provides hardware and software tools for evaluating and testing networks [15]. Their white paper comments on how to evaluate an OpenFlow switch properly, together with some pitfalls. Further, they present a methodology. However, the company does not release data comparing different switches. Our proposal has a similar methodology, however, we measure different OpenFlow primitives. Further, unlike Spirent, we also highlight the limitations of the switches.

Nguyen-Ngoc et al. [26] evaluated the Spirent hardware against a software module on commodity hardware on a Linux machine [26]. The evaluation employed three commercial switches and indicated that both methods achieve similar levels of accuracy. Their main limitation is that the presented measurements are limited to the performance of FlowMod operations only.

Finally, RFC 2544 presents a methodology for the performance measurement of switching equipment [6]. This RFC defines the test topology, the frame sizes as well as how to report the results. This paper follows the guidelines provided by RFC 2544 and uses only precomputed OpenFlow rules so that the controller does not affect the final result. RFC 2544 was superseded by RFC 6815 [7], which focuses on a testing methodology for production networks, in order to measure the performance of production network services. Since this article focuses on the benchmarking of switching hardware and not the services deployed over the hardware, RFC 2544 is still the most adequate for the tests being performed.

2.3 Analytical modeling of OF performance

An analytical model of interaction between SDN switches and SDN controller was proposed in [2]. The authors proposed a network calculus approach based on queuing theory to evaluate the performance of the SDN network, by analyzing scaling, delay-limiting, and buffer-scaling metrics. The purpose of this model is to avoid the need to spend time with extensive simulations or expensive experimental setup. Although the results can help the developers to estimate the buffer requirements of the controllers and the packet delay processing, no comparison with a hardware-based switch (COTS) was carried out to validate the model.

Analytical modeling can be used to improve simulators or to test new algorithms on a large scale. To that end, Xiong et al. [40] provide a model of the performance of OF controllers using queuing models. Their model is compared against an emulated network using CBench (Controller Benchmarker) emulation[2]. Koohanestani et al. [17] use network calculus to devise a performance model of an SDN network, considering the delays of the switch as well as those of the controller [17]. However, both articles present only analytical models and are not validated against simulations nor practical implementations.

2.4 New features for OF dataplanes

Performance modeling is also useful to extend the existing features or to check whether the switches can operate within the requirements of a certain application.

Wang et al. [39] focused their attention on the performance of OpenFlow counters [39]. They identified deficiencies in their implementation, which limits the number of counters in OF switches. Then, they proposed and evaluated a new model of OF counters in software.

OpenFlow has also potential uses in automation. Heise et al. [14] presented the case for OF in avionics, in which the delays must be deterministic. They proposed a technique using meter commands to achieve a deterministic forwarding delay.

2.5 Comparison with the state of the art

Our evaluation differs from previous papers in the following aspects: (*i*) it evaluates many hardware and software OpenFlow switches. (*ii*) our methodology only measures the performance of the data plane, eliminating the effect of the controller from the results. (*iii*) we compare how the switches perform when using legacy L2 switching ad well as OpenFlow. (*iv*) the evaluation indicates whether OpenFlow runs in hardware or software.

[2] CBench tests OpenFlow controllers by emulating packet-in messages from a large number of OpenFlow switches.

3 Scientific Methodology and Evaluation Scenario

This section is divided into two, describing the evaluation methodology for OpenFlow 1.0, and 1.3, respectively. The methodology is different due to the following reason. The controller used for the evaluation of OF 1.0 does not support OF 1.3. Hence, we would have to rerun the tests for OF 1.0 performed in [8] using a new controller. However, some of the tested switches were not available anymore, because they are currently in use in production networks. To grow the list of switches, instead of shortening it, we decided to employ a different methodology for OF 1.3 and keep the original methodology presented in reference [8] for the evaluation of OF 1.0.

As we previously stated, the main objective of the evaluation is to inspect various in the market switches in two modes, OpenFlow and Legacy, and characterize each switches' performance in OpenFlow mode. Despite the existence of newer versions of the OF standard (i.e. version 1.5.1 as of September 2018) OpenFlow versions 1.0 and 1.3 are still the most widely supported. Furthermore, OpenFlow newer versions are rarely supported by most stable firmware.

We made publicity available all data we use in this work to plot graphs as well the traces we collected during our experiments. For further details, we refer readers for: `http://netlab.ice.ufjf.br/index.php/OFperformance/`

3.1 Methodology for OpenFlow 1.0

We deploy the same methodology as proposed in our previous work [8] in respect of OF 1.0 compliant switches. However, we do extend our results to new hardware OF switch (Zodiac FX) and add an optimized version of Open vSwitch using the DPDK suite.

Evaluation Topology: Figure 1 shows the topology of our experiment. It consists of three machines connected to an OpenFlow switch. One of the machines acts as a client, another as a server, and the third as the controller. POX was employed as the SDN controller [33]. All rules installed during the experiment have an unlimited hard timeout so that packets never need to be forward to the controller for processing.

Observe the fact that, in all scenarios, the switches, as well as the testing machines vary, since different teams performed each test. However, the test software is the same in every test. Moreover, we ensured that all machines have enough computational resources to perform the tests. In other words, the machines were not saturated in any experiment.

Evaluated switches: This paper compares OpenFlow implementations for both commercial hardware and popular software switches. Because of the high cost of OpenFlow devices, the tests are limited to equipment purchased by the partner universities and research labs, and no switches with 10 Gbps were considered in this evaluation. Table 1 lists all OF 1.0 compliant switches and summarizes their main characteristics.

Table 1 List of evaluated switches

Switch	Model	OS/Firmware	Ports	CPU	OF version
Extreme x460	Summit x460-24p G2	ExtremeXOS 15.4.2.8	28	Single Core CPU 500 MHz	1.0, 1.3
Extreme x440	Summit x440-48p	ExtremeXOS 15.4.2.8	52	Single Core CPU 500 MHz	1.0, 1.3
NetFPGA	Xilinx Virtex-II Pro 50 FPGA	CentOS 5.11 + openvswitch.org OF	4	AMD Athlon II X4 800 MHz	1.0
Datacom	DM4001 ETH24GX+2x10GX	Datacom Flash 2.22	26	PowerPC e500 990MHz	1.0
DPDK	v16.11.1	Linux Ubuntu 17.04 with OvS 2.6.1	4	Intel Core i7 CPU 2.8 GHz	1.0, 1.3
HP	HP2920-24G	Firmware K 15.5 i	24	Tri Core ARM1176 625 MHz	1.0, 1.3
Mikrotik	Atheros AR9344	RouterOS 6.34.2	24	Single Core CPU 600 MHz	1.0
Pica8	P-3297	PicOS Version 2.1.5	48	P2020 Triumph2	1.0, 1.3, 1.4
Open vSwitch	OvS 2.3.0	Linux Ubuntu 14.04	-	Intel Core i7 CPU 2.8 GHz	1.0
Linksys-WRT	WRT54GL	OpenWRT + Pantou OF	4	Broadcom BCM5352 200 MHz	1.0, 1.3
Zodiac	Northbound ZodiacFX Rev A	Firmware 0.83	4	Atmel SAM4E8CA 120 MHz	1.0, 1.3

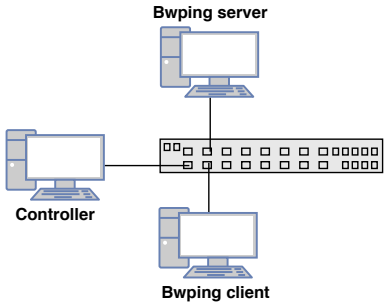

Fig. 1 The topology of the evaluation setup for OF 1.0

It is possible to group the switches into categories, which help to understand the results that will be shown in this article:

– **Commercial versus non-commercial OF switches:** Some switches are sold by established vendors as OpenFlow compliant, and as such one would expect software that is fine-tuned to the underlying hardware. Using this terminology, the switches from Extreme Networks, Datacom, Pica8, HP, and Zodiac are commercial OF switches. Meanwhile, the Linksys switch runs an open OF stack (Pantou [30]), and the Mikrotik switch runs a beta OF package from Mikrotik. Because of that, both run unsupported code, which may not be as optimized as the vendor-supported ones.
Another aspect is the use of non-commercial hardware, such as in NetFPGA. NetFPGA [20] was developed for research in high-speed networking, and because of that, it may not have the same reliability as commercial products.

- **Hardware versus software switching:** This classification deals with the possible use of hardware structures to speed up packet matches. Switches implemented in hardware usually have Ternary Content-Addressable Memories (TCAMs) to execute parallel lookups among all the flow rules. Meanwhile, software switches must perform all their matches using the software. Since the commercial switches are evaluated as black boxes, one cannot affirm that they use hardware structures for rule matching.
- **Use of OS acceleration:** Another aspect that is worth taking into consideration is the use of more efficient OS to speed up the processing. This is relevant when comparing the performance of Open vSwitch [28] with DPDK [13] since the latter is a highly optimized version of Open vSwitch.

Performance evaluation methodology: To reduce the effect of the controller on the performance, we use precomputed rules that are installed at the beginning of the experiment, with an unlimited hard timeout, and then the user flows are started. As a consequence, there is no "packet in" events during the experiments, and for that reason, the performance is not affected by the controller actions or performance.

Succeeding the setup of rules, the client emits 10,000 UDP packets, which the server sends back to the client. The number of packets was empirically chosen and great enough to bypass possible sudden unstable behaviors of the switches. After that, we compute the time each request takes to return, the Round Trip Time (RTT),with microsecond precision. We run the experiments varying the UDP packet sizes from 64, 128 to 256 bytes. The packet sizes used follow RFC 2544 and other works in the literature. Lastly, we present results with confidence intervals of 99%, unless stated differently.

3.2 Methodology for OpenFlow 1.3

OpenFlow 1.3 presents new features when compared to OpenFlow 1.0, such as a higher number of match fields and support to multiple flow tables [27]. Those new features should require more computation to perform the new operations and more memory to store the improved flow tables. Because of that, we extend our work to OF 1.3 compliant switches.

Evaluation of Topology. Figure 2 shows the evaluation topology of our experiment for OF 1.3. Server and client machines generate the UDP traffic that evaluates the data plane performance of the OF switch. Load generator machines generate extra traffic to stress out the switch. This also extends our previous work to identify whether or not additional packet flow processing affects the overall performance of OF switches. Both the testing machines and the testing applications are the same from one test to another. We have assured that all machines have enough computational resources to perform the tests, none of them having reached saturation during any test.

The controller machine runs ONOS as the SDN controller [3]. Note that the SDN controller is different from the OF 1.0 evaluation. The change is be-

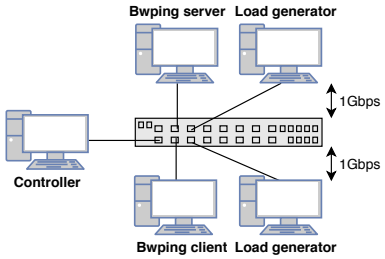

Fig. 2 The topology of the evaluation setup for OF 1.3

Table 2 List of evaluated switches

Switch	Model	OS/ Firmware	Ports	CPU	OF version
Extreme x440	Summit x440-48p	ExtremeXOS 16.1.2.14	52	Single Core CPU 500 MHz	1.0, 1.3
Extreme x460	Summit x460-24p G2	ExtremeXOS 22.3.1	28	Single Core CPU 500 MHz	1.0, 1.3
DPDK	v16.11.1	Ubuntu 17.04 with OvS 2.6.1	5	Intel i7-860 2.8 GHz x 8, 16 GB RAM	1.0, 1.3
HP	HP 2920-24G	WB 16.04.0008	24	Tri Core ARM1176 625 MHz	1.0, 1.3
Pica8	P-3297	PicOS 2.10.1	48	P2020 Triumph2	1.0, 1.3, 1.4

cause the current version of the POX lacks support for the OF 1.3 standard. It is important to highlight that our work decouples the data plane evaluation from the control plane. Therefore, using different SDN controllers does not disqualify the comparison between OF 1.0 and OF 1.3 performance results.

Evaluated switches. We compare the data plane performance of different hardware and software OF 1.3 compliant switches. Similarly to OF 1.0 evaluation, the high cost of OF switches limits our tests to the existing devices in our universities and laboratories. Furthermore, the compliance with OF 1.3 and the need of at least 5 Ethernet ports for testing also limited the set of evaluated switches, remaining only 5 out of 11 switches in respect of table 1 (Both Linksys and Zodiac do not have enough ports for our tests). Those are listed in table 2, which also presents the installed version of the firmware used during this set of experiments. For the DPDK setup, we used a 4 port Intel PRO/1000 Quad-Port PCIe 39Y6138 Ethernet adapter as the switching fabric, while connectivity to the controller was provided by the on-board Ethernet port of the computer's motherboard.

Data plane performance evaluation. The controller installs precomputed rules of an unlimited hard timeout at the beginning of each experiment. The use of unlimited timeout rules out the influence of the controller performance on the final results: the controller is never contacted when the evaluation workload is active. After installing the rules, the client sends 10,000 UPD packets to the server, which returns those packets to the client to measure the Round Trip Time (RTT) and Jitter through the OpenFlow switch. Again, the

number of packets was empirically chosen and large enough to bypass possible sudden unstable behaviors of the switches. The experiments were executed with UDP packets of sizes 64, 128, and 256 bytes, complying with related work and RFC 2544. When activated, the load generator machines generate full-duplex stress traffic of 1 Gbps using the Ostinato traffic generator[3], unless stated otherwise.

4 Results for OpenFlow 1.0

This section presents the results of the performance evaluation of different hardware and software OF 1.0 compliant switches. Here, we extend the results of our previous work [8] to two additional switches, DPDK and Zodiac.

4.1 Performance of different match types

In the first scenario, we evaluate how different header types affect the SDN switch performance. Our goal is to very if different attributes influence the time to match a given rule. We perform our tests based on individual matches of rules which are formed among the 12 attributes of OpenFlow 1.0, as shown in Table 3.

Table 3 Subsets of attributes used for matches

Type of match	Matched attributes
Port	Switch port
MAC	Source and destination MAC addresses
IP	Ethertype and source and destination IP addresses
UDP port	Ethertype, source and destination IPs, transport protocol, source and destination UDP/TCP ports
Exact	All fields

Table 4 shows mean delay and confidence interval in a scenario we use 64-byte packets. The performance is very similar for all match types, which suggests that the number of attributes or the type or match used does not substantially impact the packet delay. Note that the HP and Zodiac switches are exceptions. Later in this section, we further discuss this effect and present an explanation for this fact.

The confidence intervals show that there is a statistically significant difference in the results for different match types. There are three distinct behaviors for the match types per port, MAC, IP, and UDP port. The Extreme switches (X460 and x440), Pica8, Mikrotik, Datacom, DPDK, NetFPGA card, and

[3] `https://ostinato.org/`

Table 4 Confidence intervals for the delay in different match types (in ms)

Switch	Match type				
	Port	MAC	IP	UDP port	Exact
Extreme x460	0.125 - 0.127	0.124 - 0.126	0.124 - 0.126	0.124 - 0.126	0.121 - 0.125
Extreme x440	0.174 - 0.175	0.131 - 0.133	0.130 - 0.132	0.119 - 0.121	0.117 - 0.119
Open vSwitch	0.514 - 0.519	0.502 - 0.508	0.504 - 0.508	0.513 - 0.517	0.508 - 0.511
NetFPGA	0.201 - 0.202	0.197 - 0.198	0.196 - 0.202	0.201 - 0.201	0.194 - 0.197
Datacom	0.198 - 0.200	0.201 - 0.202	0.199 - 0.200	0.199 - 0.200	0.199 - 0.200
DPDK	0.162 - 0.163	0.158 - 0.158	0.160 - 0.161	0.161 - 0.161	0.144 - 0.158
Pica8	0.200 - 0.200	0.197 - 0.199	0.196 - 0.201	0.197 - 0.198	0.200 - 0.200
Mikrotik	0.248 - 0.248	0.248 - 0.249	0.251 - 0.253	0.248 - 0.249	0.243 - 0.243
HP	1.069 - 1.147	1.049 - 1.113	1.064 - 1.309	1.038 - 1.106	0.743 - 1.001
LinkSys-WRT	1.025 - 1.051	1.017 - 1.045	1.019 - 1.049	1.017 - 1.046	1.018 - 1.047
Zodiac	0.284 - 0.331	0.284 - 0.330	0.280 - 0.326	0.273 - 0.316	0.388 - 0.514

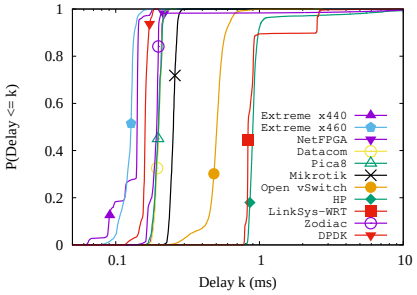

Fig. 3 Cumulative distributions of delays for the match by IP

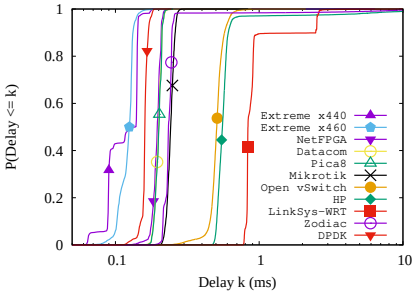

Fig. 4 Cumulative distributions of delays for the exact match

Zodiac presented the lowest delays, achieving from 0.1 up to 0.3 ms, approximately. Meanwhile, Open vSwitch obtained 0.5 ms. HP switch and LinkSys-WRT were the worst performers, having an average delay of approximately 1 ms. To illustrate this, the graph in Figure 3 shows the cumulative distribution function (CDF) of the packet delays when matching the IP address. The same behavior appeared in matches by MAC, Port, and UDP port. One can cluster the switches into the three previously mentioned groups. Again, HP and LinkSys-WRT were the worst performers, showing packet delays of near 1 ms.

For the exact match, the results were very similar to matching IP addresses. The HP switch also had the lowest delay for this scenario. Figure 4 shows the CDF of packet delays for exact matches. The results for the HP device are more to the left, almost reaching the curve for Open vSwitch. This shows that for, this type of match, the HP switch has better performance, improving by 21%. Meanwhile, the Extreme switches are still the best performers, and OpenWRT Linksys continues to perform the worst.

We performed tests to identify the influence of the packet size in matches, the tests were repeated with a packet size of 256 instead of 64 bytes. For increased packet sizes, the HP, Extreme X460, Mikrotik, Pica8, Datacom,

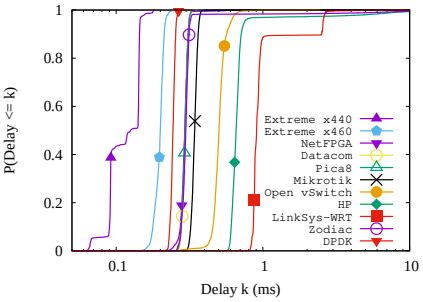

Fig. 5 Cumulative distributions of delays for the exact match on 256-bytes packets

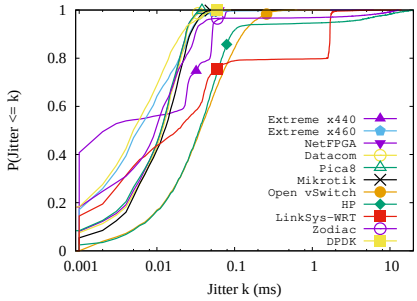

Fig. 6 Cumulative distributions of jitter for 64-bytes packets

DPDK, NetFPGA, and Zodiac had the highest increases in delay. Other switches had no significant increase in their delays, as shown in Figures 5 (256 byte packets) and 4 (64 byte packets). The shift of the curves for Extreme X460, HP, Mikrotik, Datacom, and NetFPGA is noticeable between the graphics.

Figure 6 shows the CDF of packet jitter for exact matches. Extreme and DPDK switches have the lowest jitters (less than 1 ms). For the HP switch, we see that it performs similarly to a software device. This indicates that HP does not run OF in hardware.

Finally, it is worth evaluating the performance gains of the software acceleration of the DPDK stack. The DPDK version of OvS improved the performance of the PC software switch by approximately 70%, in accordance with the performance boost of 87.5% presented by [11] for packet forwarding. This significant speedup allowed DPDK to perform similarly to commercial switches such as the Extreme and Pica8 switches, even outperforming the latter as shown by figures 3 and 4.

4.2 Legacy and OpenFlow switch modes performance

We compare how each switch performs in legacy L2 mode with the performance using OpenFlow. In OpenFlow, the installed rules create a preset path from a certain input port to another output port. This path is fixed for both directions and matches the same ports used on the legacy mode. As a consequence, this test is performed with only two flow rules, being one for each direction of communication.

Figure 7 shows the difference in performance for both modes (speedup), also considering three packet sizes. The Speedup divided the performance in L2 legacy by the latency using OpenFlow forwarding. Hence, values less than 1.0 indicate a performance degradation in OF mode, while values greater than 1.0 indicate superior performance using OF.

Many switches present similar performance in both modes, such as the Extreme switches (X460 and x440), Open vSwitch, Mikrotik, Pica8, Datacom, DPDK and NetFPGA. This means that those switches have a well-optimized

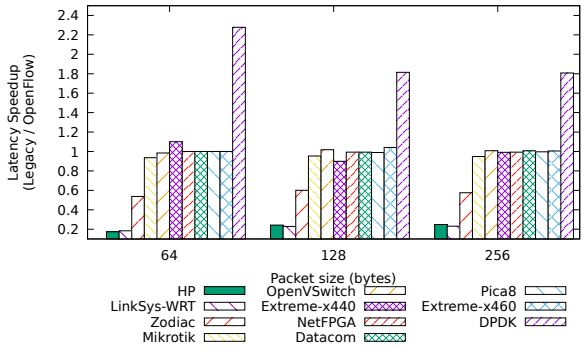

Fig. 7 Switch performance on OF mode when compared to L2 switch legacy mode

implementation, possibly taking advantage of the hardware available on the platform. On the other hand, the performance of HP, Linksys-WRT, and Zodiac when running OpenFlow worse than the performance in legacy mode, ranging from 20% up to 50% of legacy performance. One reason for this difference is that the implementation prioritizes SRAM and not TCAM.

DPDK shows a significant speedup in OF mode. Indeed, the L2 legacy stack does not employ any DPDK optimization. As a consequence, legacy mode presents a noticeable worse performance.

4.3 Single and multi-flow performance

In this third scenario, we evaluate switches when the flow table has a single flow and compare the performance when the flow table is nearly full. This scenario evaluates how the flow tables are organized. For example, if the switch performs sequential or parallel matching, the complexity of the matching, e.g. if the implementation uses hashes or runs a binary search.

First of all, we ran tests to identify the maximum number of rules that each switch accepts in both OpenFlow and legacy modes. The results are shown in Table 5. Since the manufacturers seldom disclose the number of rules on their datasheets or by e-mail, we had to perform experiments in several switches to identify that value. To do so, we installed rules until the switch returned a table full message. Also, some switches become unstable with too many rules, so Table 5 shows the number of rules that still ensure a stable operation. For example, DPDK supports 17,500,000 rules; however, the switch is unstable when the number of rules is greater than 24,000 as it rarely responds to OpenFlow FlowStats commands.

Additionally, the maximum number of rules of the DPDK switch depends on the available physical memory of the host computer. Nevertheless, we have found in our tests that the response time increases and instabilities occur as the number of rules grows. Therefore, the number of rules was limited in order

Table 5 Number of supported rules on legacy and OpenFlow modes

Switch	Number of rules in legacy mode	Number of rules in OF mode
Extreme x460	2,048	1,200
Extreme x440	1,024	248
NetFPGA	16	124
Datacom	32,768	2,051
DPDK	17,500,000	24,000
Pica8	12,000	4,096
Mikrotik	16,318	389
Open vSwitch	1,000,000	750,000
HP	2,048	16,000
LinkSys-WRT	100	100
Zodiac	142	142

Table 6 Confidence intervals of delays in tables with one and multiple rules (in ms)

Switch	One rule	Multiple rules
Extreme x460	0.121 - 0.125	0.124 - 0.128
Extreme x440	0.109 - 0.110	0.114 - 0.115
NetFPGA	0.197 - 0.198	0.196 - 0.197
Datacom	0.199 - 0.200	0.199 - 0.200
DPDK	0.158 - 0.159	0.155 - 0.176
Pica8	0.196 - 0.200	0.199 - 0.200
Mikrotik	0.243 - 0.244	0.246 - 0.247
Open vSwitch	0.508 - 0.511	0.513 - 0.517
HP	1.049 - 1.120	1.046 - 1.330
LinkSys-WRT	1.028 - 1.055	1.215 - 1.242
Zodiac	0.326 - 0.372	1.763 - 1.809

to cope with a stable operation rather than using the entire physical memory of the host computer.

For most of the switches, the legacy mode can accommodate more rules than the OF mode. One reason for that is that OF matches need more information. For example, the Extreme switches (X460 and x440) store rules on access-control-lists (ACLs) in hardware for OF and legacy mode. For legacy mode, the size of the ACL is "single", while the ACL is set to "double" in OpenFlow. This halves the number of rules in OF mode.

Meanwhile, software switches do not have such strict limitations. Open vSwitch stores up to 1 million rules on its hash tables when considering exact matches. However, this number varies based on the types of rules and network conditions. As an example, wildcard matches (when we do not match all the fields) employ a linear table with only 100 rules.

The HP switch is also an interesting case. As described in the manual, the OF mode creates a flow table in software. It is not possible to disable it to employ faster memories, e.g. the TCAM. This difference in table size, which indicates the use of SRAM, may be the cause for the bad results on previous scenarios.

Table 7 Confidence intervals for the delay in header rewrites (in ms)

Switch	Type of rewrite					
	No rewrite	MAC	IP	VLAN priority	IP ToS	UDP port
Extreme x460	0.125 – 0.127	0.125 – 0.129	-	0.125 – 0.128	-	-
Extreme x440	0.114 - 0.115	0.113 - 0.115	-	-	-	-
NetFPGA	0.193 – 0.197	0.191 – 0.193	-	0.194 – 0.195	-	0.193 – 0.194
Datacom	0.198 - 0.203	0.199 - 0.200	-	-	-	-
DPDK	0.158 - 0.159	0.160 - 0.161	0.156 - 0.157	-	0.159 - 0.160	0.161 - 0.162
Pica8	0.200 - 0.201	0.199 - 0.200	0.200 - 0.200	0.200 - 0.201	-	-
Mikrotik	0.244 – 0.248	0.244 – 0.247	-	0.245 – 0.248	-	-
Open vSwitch	0.508 - 0.511	0.511 - 0.516	0.510 - 0.515	-	0.506 - 0.510	0.512 - 0.515
HP	1.121 – 1.212	1.128 – 1.203	1.107 – 1.178	1.107 – 1.183	1.122 – 1.194	1.071 – 1.132
LinkSys-WRT	1.032 – 1.058	1.034 – 1.061	1.034 – 1.060	-	1.030 – 1.056	1.033 – 1.060
Zodiac	0.326 - 0.372	0.324 - 0.371	0.344 - 0.392	0.329 - 0.375	0.318 - 0.362	-

Table 6 shows the confidence intervals for average packet delays in two situations: for a flow table with just one flow rule, and a flow table having many rules. In the latter, the match will take place for the last rule that has been installed. The number of rules for the latter case is the maximum number of rules found in Table 5. For almost every switch, we found no difference in the delays for a full table or a table with one rule only. This shows that the organization of the flow table does not affect the performance when many rules are installed. Exceptions are the LinkSys-WRT and the Zodiac switches, which have an average increase in packet delay of around 0.2 and 1.4 ms, respectively.

4.4 Packet header rewrite performance

The fourth scenario evaluates the packet header rewrite operation. This scenario performs rewrites in fields such as MAC and IP destination addresses, VLAN priority, IP Type of Service, and UDP port.

Table 7 lists the mean value and confidence intervals for the packet delays when they suffer rewrite actions on five fields of their headers, individually. For comparison, the table also lists the delays for packets processed without rewrites. In all cases, there is no significant increase in the average delay time of the modified packets, when compared to unmodified packets. This indicates that the rewrite operation can occur on the order of a few microseconds. However, most of the switches do not support packet rewrites on all OF 1.0 headers. This lack of support, coupled with the fact that only one header rewrite is allowed per rule, shows that there are still aspects to improve on existing OF implementations, especially in hardware.

Again, DPDK shows performance similar to the most efficient commercial switches. When comparing DPDK to Open vSwitch, one finds a mean speedup of 3 three times, showing the importance of OS-level optimization for effective packet forwarding.

4.5 Performance of Portstats and Flowstats Operations

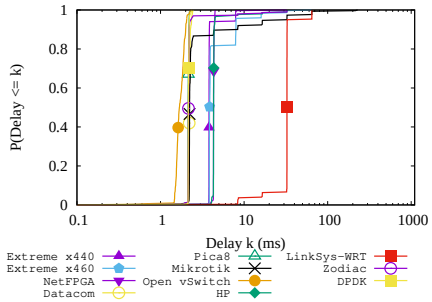

Fig. 8 Flowstats delays for an almost empty table under heavy workload

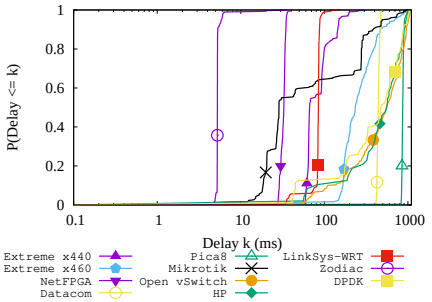

Fig. 9 Flowstats delays for a nearly full table under heavy workload

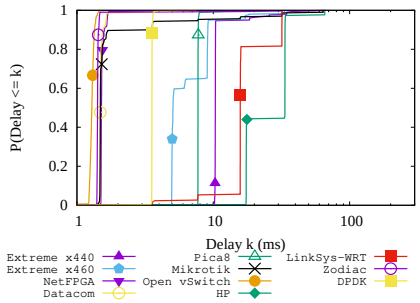

Fig. 10 Portstats delays for an almost empty table under heavy workload

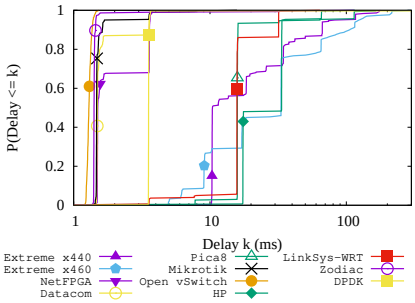

Fig. 11 Portstats delays for a nearly full table under heavy workload

The fifth scenario analyses, for each switch type, portstats, and flowstats operation performance. In this evaluation, the controller asks, every second, for statistics (flowstats or portstats). It also collects the response time. Iperf[4] version 2.08 generates the network load. For the light load, we used the default 1 Mbit/s configuration of Iperf for UDP packets, while for the heavy load we set up Iperf to generate UDP packets at 800 Mbit/s, near the maximum throughput supported by the hardware interface of our traffic generators. The only exception is the Zodiac switch whose network interface supports up to 100 Mbit/s; therefore, corresponding to the heavy load traffic.

When running this test, we found out that some switches become unstable when their tables reach their capacity. Thus, we have to empirically discover the maximum number of rules that each switch accepts while maintaining a stable behavior. The number has been defined based on Table 5 because many of the switches crashed for portstats or flowstats when the flow table was nearly full. In this case, we installed rules until the switches reached 90% of the number of rules shown in Table 5, in order to obtain a stable behavior.

Figures 8 and 9 show the CDF for flowstats delays for one single rule, and also for a nearly full flow table, respectively. For the two figures, there

[4] https://iperf.fr/

is a heavy load. For all switches, we have seen an increase in the flowstats delay. This result agrees with intuition because it should be more complex to return statistics when more rules are installed. Meanwhile, for portstats, the reduction in performance is smaller for more OF rules. As shown in Figures 10 and 11, a few of the devices (Pica8, NetFPGA, Extreme x460, and x440 and Datacom) present performance degradations for a large number of rules.

The Northbound ZodiacFx OF presented many switch disconnections when the table was full, similar to the Mikrotik switch. Since there are no OpenFlow messages to check the occupancy of the tables, we had to decrease the number of installed table rules to achieve stable operation. The switches did not present any disconnections with almost empty tables. We succeeded in DPDK decreasing the table to 24,000 rules when it lost only 2.3% of its FlowStats replies. PortStats primitives demand less processing than FlowStats.

Table 8 shows the flowstats delay, as well as the confidence intervals, for different workloads. The effects of the workload increase are different among the switches. There was an increase in delay for DPDK, Mikrotik, and Linksys-WRT. Open vSwitch, HP, Pica8, Datacom, NetFPGA, and Zodiac did not show variations on the delay. Curiously, we have seen a decrease in the delay for the Extreme switches (x460 and x440).

Table 9 depicts the means of portstats delay. A higher workload generated a higher delay on the Extreme x460, HP, and Linksys switches. This is expected since flowstats should be more affected by the number of flows than portstats. However, the results indicate that the type of workload is more important to performance than the size of the flow table. We have seen differences in performance due to the number of rules only for Extreme x440, Pica8, HP, and Zodiac.

When comparing the hardware and software switches, there is no significant performance difference for portstats and flowstats operations. The highest difference occurs for flowstats, where curiously the software implementations had a better performance. Also, in this scenario, we see that the OS optimizations of DPDK do not generate significant gains for flowstats and portstats operations since OvS and DPDK performed similarly. As DPDK doesn't have TCAM it doesn't have the O(1) complexity on each prefix matching.

4.6 Discussion

In general, the OF 1.0 optional specifications were not completely implemented in hardware switches. This finding is important to academia and industry since they can not expect that the entire OF standard, including matching all the headers, is supported. A classic example is the VLAN field. Stability is another problem. When the rule table is almost full, some data planes become unstable, while others crash when a certain number of rules is installed, and some can only handle a restricted number of OF control messages over time. About the OF operation's performance, user can not choose where to install the rules, for example, SRAM or TCAM. When a switch has both memories,

Table 8 Flowstats delays confidence intervals (in ms)

Switch	Table with one rule		Nearly full table	
	Light load	Heavy load	Light load	Heavy load
Extreme x460	4.430 - 5.762	4.646 - 6.042	275.726 - 358.626	256.013 - 332.985
Extreme x440	3.639 - 4.735	3.595 - 4.677	80.360 - 104.520	76.908 - 100.032
NetFPGA	3.748 - 4.874	3.758 - 4.888	27.108 - 35.258	27.971 - 36.381
Datacom	1.939 - 2.521	1.938 - 2.520	381.685 - 496.443	384.893 - 500.615
DPDK	2.159 - 2.174	2.156 - 2.167	21.623 - 76.778	77.332 - 163.675
HP	4.048 - 5.264	4.224 - 5.494	375.217 - 699.857	363.165 - 720.289
Mikrotik	7.340 - 9.548	5.518 - 7.176	13.659 - 17.767	130.058 - 169.160
Pica8	1.876 - 2.440	1.868 - 2.430	751.442 - 977.370	756.914 - 984.488
Open vSwitch	1.763 - 2.293	1.502 - 1.954	459.776 - 598.012	483.018 - 628.242
LinkSys-WRT	7.659 - 9.961	28.582 - 37.176	31.619 - 41.125	71.672 - 93.220
Zodiac	1.878 - 4.782	2.230 - 3.219	5.084 - 7.030	5.185 - 5.758

Table 9 Portstats delays confidence intervals (in ms)

Switch	Table with one rule		Nearly full table	
	Light load	Heavy load	Light load	Heavy load
Extreme x460	5.987 - 7.787	6.288 - 8.178	25.574 - 33.264	35.053 - 45.591
Extreme x440	9.566 - 12.442	9.913 - 12.893	21.577 - 28.065	23.582 - 30.672
NetFPGA	1.332 - 1.732	1.341 - 1.745	2.260 - 2.940	1.903 - 2.475
Datacom	1.345 - 1.749	1.327 - 1.725	1.486 - 1.932	1.542 - 2.006
DPDK	3.543 - 3.574	3.331 - 3.800	3.546 - 3.552	3.283 - 4.230
HP	15.728 - 20.456	23.706 - 30.834	15.904 - 20.686	23.862 - 31.036
Mikrotik	1.480 - 1.924	2.860 - 3.720	3.017 - 3.923	1.445 - 1.879
Pica8	6.791 - 8.833	6.791 - 8.833	19.700 - 25.622	18.373 - 23.897
Open vSwitch	1.261 - 1.641	1.120 - 1.456	1.258 - 1.636	1.131 - 1.471
LinkSys-WRT	4.480 - 5.826	15.876 - 20.650	4.627 - 6.019	15.288 - 19.884
Zodiac	1.319 - 2.141	1.319 - 2.014	1.304 - 1.828	1.067 - 2.936

it automatically decides where to store the rule by checking the match type (exact or inexact).

Testing a highly optimized version of Open vSwitch using the Intel DPDK suite of Linux kernel modification shows that some OS optimization may improve the performance. DPDK got about two times the performance on the same Open vSwitch hardware by deploying the Huge Table filesystem as buffers of 2 Mbytes and using dedicated cores allocated to serve the DPDK network driver at each network interface. In fact, in our tests, only two cores were dedicated to the Operating System, and the six remaining cores were used exclusively by the DPDK network driver.

The application determines which type of OF switch to use since each data plane has a difference in performance, number of OF rules, packet rewrite capability. In general, software switches can have larger forwarding tables, are easier to upgrade and so are more compliant with OF standards. Moreover, hardware switches can only execute packet rewrite on fewer fields and do not support larger tables. Regarding the performance of OF software switch, with DPDK and XDP, OS-optimized code can execute as fast as hardware switches.

The Extreme switches presented the best performance results in general when comparing only the hardware switches. HP switch had performance similar to the software switches, which is a strong indication that the HP switch performs operating rule and parsing using software instead of using specialized hardware.

In conclusion, research and industry must know that the current implementations have limitations. If complete OF compliance is needed, we recommend software-based implementations. On the other hand, for production networks, the manager must analyze carefully the hardware switches performance and limitations and choose the one that best suits its network requirements.

5 Results for OpenFlow 1.3

This section presents the results of the data plane performance evaluation of different hardware and software OpenFlow 1.3 compliant switches. The focus of this scenario is on the new functionality brought by OF 1.3, so we do not replicate the test cases performed in OF 1.0. Further, since DPDK performed better than Open vSwitch, this section considers only DPDK in the software switch category.

5.1 Single and multi-flow performance

This scenario evaluates the data plane performance of the switches in two different ways. First, we evaluate the data plane performance when the flow table has only one rule, consisting of the exact match (Table 3) plus two extra match fields supported by OF 1.3, IP ECN (Explicit Congestion Notification), and IP DSCP (Differentiated Services Code Point). This rule always "matches", allowing network traffic through the switch. Second, we evaluate the switch performance when the flow table is nearly full. High priority rules have an invalid match field. Only the rule of lowest priority is entirely valid, so packets are tested for all rules in the flow table. The goal of comparing single and multi-flow performance is to verify the influence of how the internal table is organized, for example, if matching uses binary search or hashing, or if matching is sequential or parallel, and so on.

Testing multi-flow performance requires knowledge of the maximum number of rules that can be installed on each switch. Table 10 lists the maximum number of rules found on datasheets while operating in the legacy mode as well as the maximum number of rules we could empirically install on OF 1.3 mode. Notice that OF 1.3 usually accommodates a smaller number of rules than the legacy mode, depending on the switch. As we believe that HP is a software switch we can suppose that this model HP 2920 has a bigger quantity of memory available. The reason this might happens is due to additional attributes that OF 1.3 mode must store in comparison with the legacy mode, requiring more storage memory. For example, the Extreme x440 requires two

Table 10 Number of supported rules on legacy and OpenFlow 1.3

Switch	Number of rules in legacy mode	Number of rules in OF 1.3
Extreme x440	1,024	128
Extreme x460	2,048	1,280
DPDK	700,000	14,000
Pica8	12,000	4,096
HP	2,048	14,000

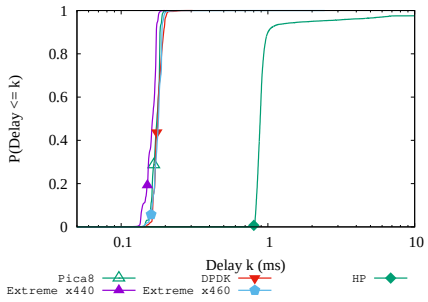

Fig. 12 Delays for a single-flow performance under no workload

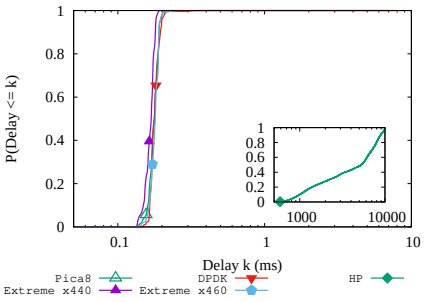

Fig. 13 Delays for a multi-flow performance under no workload

memory entries to store a single OF rule. Furthermore, because of the already mentioned instability of the switches for a full table, we have installed 90% of the number of rules presented in Table 10.

Additionally, each switch may have its own peculiarities in terms of storage. Extreme switches (x440 and x460) keep rules on ACLs in hardware. Extreme x440 has four memory slices, accommodating 256 rules per slice while operating in legacy mode. However, three memory slices are automatically allocated to the operating system, remaining a single slice for storing the OpenFlow rules. Extreme x460 also reserves three memory slices for the operating system, remaining five out of eight memory slices for OF rules.

DPDK and HP switches create flow tables in software. This results in a much greater storage capacity of OpenFlow rules. A maximum number of 14,000 rules is imposed by the SDN controller. Beyond that, the ONOS controller throws "out of memory" exceptions. It is possible to increase the amount of memory each thread may access, allowing more than 14,000 rules. However, we respect the default configuration of ONOS as the imposed number of rules is already considerably huge by comparison with other evaluated switches.

Figures 12 and 13 show the CDF for single and multi-flow performance, respectively, in terms of delay. Background traffic generators are deactivated, and the UDP packet size is 64 bytes. There is no significant performance degradation from single to multi-flow scenarios but the HP switch. The latter performs badly in both scenarios by comparison with other switches. The performance is even worse in the multi-flow scenario by the extent of three to four orders of magnitude as shown by the inner graph in figure 13.

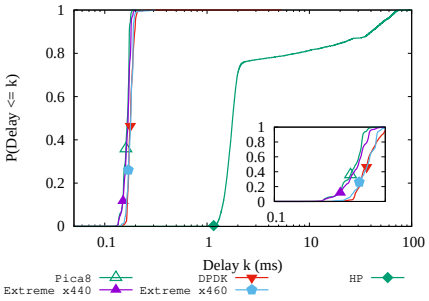

Fig. 14 Delays for a single-flow performance under heavy workload

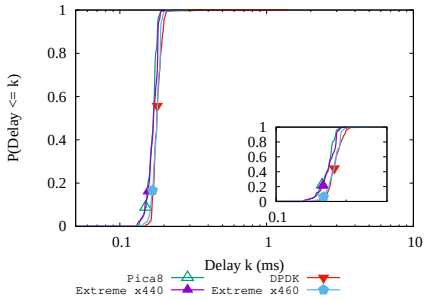

Fig. 15 Delays for a multi-flow performance under heavy workload

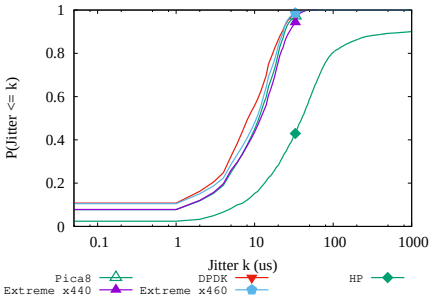

Fig. 16 Jitters for a single-flow performance under no workload

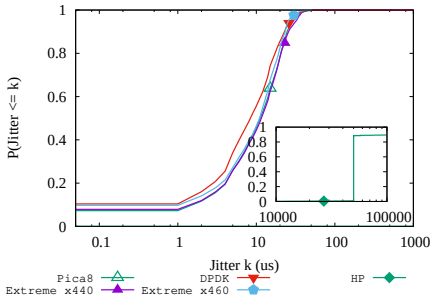

Fig. 17 Jitters for a multi-flow performance under no workload

Figures 14 and 15 also display the CDF for single and multi-flow performance in terms of delay. However, the background traffic generated between two machines is now activated. The heavy workload consists of 1 Gbps full-duplex traffic for all devices except the HP switch. The latter was unable to cope with such traffic, coping with only 1 Mbps full-duplex. Even though the HP switch has a lighter workload, it presents the worst performance as shown in Figure 14. The performance of the HP switch is slightly worse than Figure 12 (background load deactivated) but is much better than Figure 13 (multi-flow under no workload). Furthermore, the HP switch does not cope with high volume traffic in the multi-flow scenario and, therefore, is not shown in figure 15. Other switches have not presented significant deviations from the previous scenarios, as figures 12 and 13 show.

Jitter evaluation leads to similar results to the presented above as shown in Figures 16, 17, 18, and 19. The testing scenarios are the same as before. Only the HP switch presents significant deviation among the graphs. Multi-flow performance (Figure 17) is much worse than the single-flow performance (Figures 16 and 18), being the HP switch unable do handle workload during the multi-flow test under heavy workload (Figure 15).

The UDP packet size does not affect the results to a great extent. Figures 20 and 21 show the CDF in the multi-flow scenario for UDP packet sizes of 64

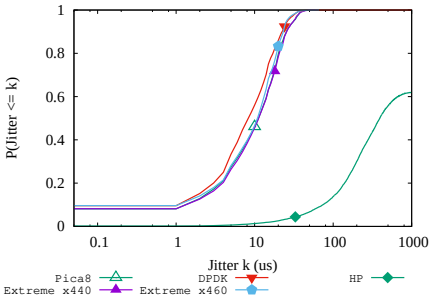

Fig. 18 Jitters for a single-flow performance under heavy workload

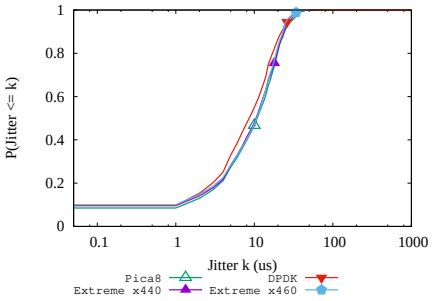

Fig. 19 Jitters for a multi-flow performance under heavy workload

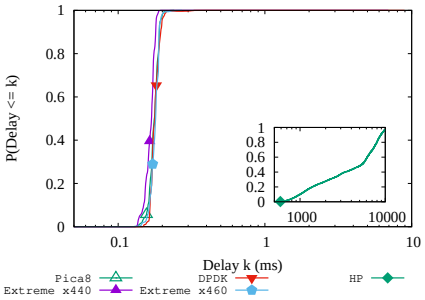

Fig. 20 Delays for a multi-flow performance under no workload, UDP packet size of 64 bytes

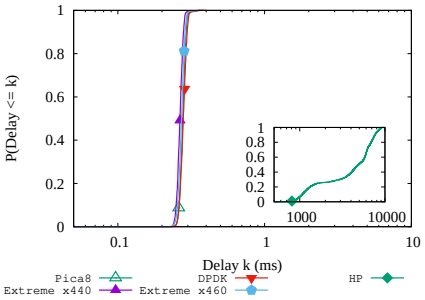

Fig. 21 Delays for a multi-flow performance under no workload, UDP packet size of 256 bytes

and 256 bytes, respectively. The performance shift is approximately 0.10 ms from figure 20 to 21. Indeed, a similar performance shift has been observed among all tests of single and multi-flow performance, indicating that the packet size does not affect the data plane performance significantly.

5.2 Multi-table performance

This scenario evaluates the data plane performance of OF 1.3 compliant switches when using multiple tables. This feature is not available in OF 1.0 and was firstly introduced in OF 1.1, together with a more flexible pipeline, albeit the "next-table" capabilities were introduced only in OF 1.3 [27].

We evaluate the data plane performance of nearly full tables, as previously defined. The flow rules are the same as the previously described ones for single and multi-flow performance. For each table, high priority rules have an invalid match field, and only the lowest priority rule is entirely valid. The latter consists of a "goto" rule to the next table for all tables but the last one, which is an exact valid match, allowing traffic through the switch. Therefore, packets are tested for all flow rules in all flow tables.

Table 11 Number of tables and rules per table for legacy and OpenFlow 1.3

Switch	Number of tables	Number of rules per table
DPDK	110	128
Pica8	66	32
HP	4	3,500

Multi-table performance does not only require knowledge of the maximum number of rules but also the maximum number of flow tables of each switch. Table 11 shows the number of tables and the number of rules for each switch in this scenario. Extreme switches are not presented in this table. Although Extreme switches are compliant with OF 1.3, they do not cope with multiple tables (as indicated by the manufacturer in [25]) and, therefore, are not evaluated in this section.

Each switch has its own peculiarities in terms of storage. As mentioned in section 4.3, the DPDK and HP switches create flow tables in software. Similarly to multi-flow analysis, the storage capacity is bounded by the ONOS controller to approximately 14,000 flow rules in total. We have distributed those flow rules among 110 flow tables of 128 flow rules each for DPDK. Despite the software implementation, the HP switch has only four tables, each populated with 3,500 flow rules, which is the maximum number of flow rules supported in ONOS.

Pica8 has subtle mechanisms for dealing with multiple tables. The switch has a single TCAM memory for storing 4,096 flow rules at most in legacy mode (Table 10). Compliance with multiple tables is obtained through normalization of flows into only one hardware table [32]. The normalization process consists of simulating the multi-table logic on a single table, so rules are modified to replicate the same behavior as using multiple tables. The switch copes with 66 tables at most and performs multiple expansion and compression in TCAM during the normalization process. We have empirically found that the switch becomes unstable beyond 32 rules per table, as it leads to a lack of memory during the normalization process. Therefore, we have set 32 rules per table for the multi-table performance tests.

Figures 22 and 23 show the multi-table performance in terms of delay for UDP packets of 64 bytes. Firstly, Figure 22 presents the results when the load traffic generators are deactivated. Pica8 and DPDK switch have very similar performance. For example, the average delay for DPDK and Pica8 is 0.179 ms and 0.176 ms in Figure 22 with small dispersion, respectively. On the other hand, the HP switch presents on its CDF a dispersion range of up to four orders of magnitude by comparison with others. It is also noticeable that the HP switch presents a big variance in its performance, as shown by the subgraph in Figure 22. Secondly, Figure 23 displays the results under a workload of 1 Gbps full-duplex, the maximum throughput supported per port. Despite the slightly worse performance of DPDK, both Pica8 and DPDK switches perform very similar to Figure 22. We have tested the HP switch under workloads from

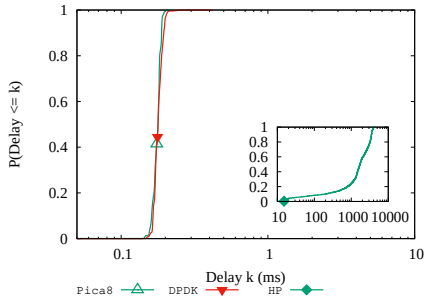

Fig. 22 Delays for multi-table performance under no workload.

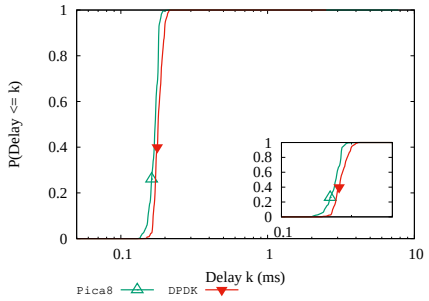

Fig. 23 Delays for multi-table performance under a heavy workload.

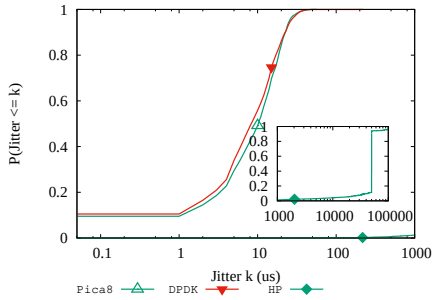

Fig. 24 Jitter for multi-table performance under no workload

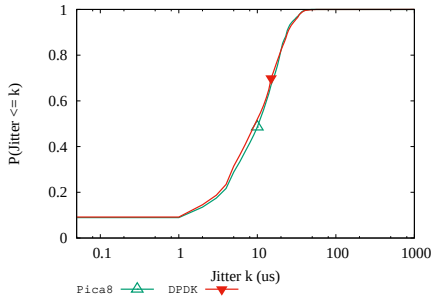

Fig. 25 Jitter for multi-table performance under heavy workload

1 Mbps to 1 Gbps full-duplex, and the switch was unable to cope with any of them.

Figures 24 and 25 show the multi-table performance in terms of jitter. The testing environment is the same as for the delay measurements. Pica8 and DPDK perform very similarly to each other and are not affected by the workload to any significant extent as shown by Figures 24 and 25. The HP switch deviates from others to up to 4 orders of magnitude under no workload, figure 24. Additionally, it does not cope with any workload from 1 Mbps to 1 Gbps, and therefore it is not displayed in Figure 25.

Similarly to single and multi-flow performance, the UDP packet size does not affect the multi-table performance to a great extent. The performance shift is equivalent to the one displayed in Figures 20 and 21, approximately 0.10 ms.

5.3 Discussion

Not all switches in table 10 fully implement features of OF 1.3, albeit vendors may tag them as OF 1.3 compliant devices. For example, Extreme switches lack support for multiple tables. This is relevant to both academia and indus-

try, as one may assume that all features are supported only based on overall information of compliance with the protocol, which may not be true.

Switches usually become unstable when the flow table is full (or nearly full). We have observed that in some cases the operating system partially takes or uses the memory used for allocation of flows. This often occurs without any notice. Then, installing flows to full capacity may lead to a lack of memory, resulting in unstable operation of switches. We have set an empirical threshold of approximately 90% of the flow table capacity, so switches operate properly.

Similarly to OF 1.0, software-based switches tend to be more compliant with OF 1.3 specification as published by Open Network Foundation[27]. Among the tested switches, HP and DPDK are software-based and have supported all the required features in single and multi-flow and multi-table tests. On the other hand, Extreme switches do not cope with multiple tables. Albeit Pica8 copes with all tests, it has a subtle mechanism for dealing with multiple tables through normalization.

Additionally, software-based switches have fewer restrictions in terms of storage. Both HP and DPDK are bounded by the SDN controller in our tests. The imposed number of rules is significantly greater than the other switches, as shown in tables 10 and 11.

The downside of software-based switches seems to be performance. HP is worse than others by 4 orders of magnitude, not even supporting a heavy workload in some cases. DPDK also has a worse overall performance by comparison with Extremes and Pica8. However, the deviation of DPDK is little or even insignificant. This corroborates the observation in the OF 1.0 tests, in which software implementations together with OS optimizations may lead to a similar overall performance of hardware-based switches.

The packet size does not seem to play a significant role in data plane performance. The small deviations of approximately 0.1 ms on packet sizes from 64 bytes to 256 bytes do not indicate any significant impact on overall performance.

6 Conclusion

In this article, we evaluate software and hardware OpenFlow switches implementing OF versions 1.0 and 1.3. The objective is to understand how mature the production, as well as research-oriented switches, are because the use of OpenFlow is growing in academia and commercial deployments. Furthermore, the amount of OF-compliant devices increases daily.

Our evaluation considered eight commercial hardware switches, one academic hardware platform (NetFPGA), and two open-source software switches. The evaluation measured the delay on packet delivery, the performance of switches on different types of header match rules, and if there was a performance gap in switches operating on OpenFlow mode when compared to its legacy L2 switch.

Results have shown that there are many differences in the way OpenFlow is implemented, and the implementation strategy impacts the overall performance. Some switches perform OpenFlow operations in hardware, while others implement it in software.

Also, we have seen that hardware-optimized implementations affect the final performance as well as the available features. For example, hardware switches do not provide features that are assumed to be "universally supported" in SND papers: flow tables with thousands of entries, the capability to rewrite the packet fields, as well as matches in every field of L3 and L4 protocols supported in OF. OS-optimized software implementations, on the other hand, maybe an attractive choice. They support a larger number of flow rules, have a performance comparable to hardware switches, and in our evaluation supported more OF features than the evaluated hardware switches.

Newer versions of OF are not fully supported by switches. In our evaluation, few switches support OpenFlow 1.3 and fewer have support to the multi-table feature. Despite the multi-table support, Pica8 performs table normalization to fit multiple tables in a single hardware table. The DPDK does not only support multiple tables but also performs very similar to Pica8, suggesting, again, OS-optimized software implementations as an attractive choice.

As future work, we intend to expand the list of evaluated switches by reaching out to the research community. We also plan to evaluate the conformance of switches with newer versions of OpenFlow.

Acknowledgments

The authors would like to thank CAPES, CNPq, FAPEMIG for their financial support in this research. This research has been partly funded by the FUTEBOL project (European Project H2020 grant no. 688941, jointly funded by RNP). We also thank Christian E. Rothenberg for his invaluable reviews and RNP for providing the Datacom router for evaluation.

References

1. Appelman, M., De Boer, M.: Performance analysis of OpenFlow hardware. Semester thesis project report, University of Amsterdam (2012)
2. Azodolmolky, S., Nejabati, R., Pazouki, M., Wieder, P., Yahyapour, R., Simeonidou, D.: An analytical model for software defined networking: A network calculus-based approach. In: 2013 IEEE Global Communications Conference (GLOBECOM), pp. 1397–1402 (2013). DOI 10.1109/GLOCOM.2013.6831269
3. Berde, P., Gerola, M., Hart, J., Higuchi, Y., Kobayashi, M., Koide, T., Lantz, B., O'Connor, B., Radoslavov, P., Snow, W., Parulkar, G.: Onos: Towards an open, distributed sdn os. In: Proceedings of the Third Workshop on Hot Topics in Software Defined Networking, HotSDN '14, pp. 1–6.

ACM, New York, NY, USA (2014). DOI 10.1145/2620728.2620744. URL http://doi.acm.org/10.1145/2620728.2620744
4. Bianco, A., Birke, R., Giraudo, L., Palacin, M.: Openflow switching: Data plane performance. In: IEEE International Conference on Communications (ICC), pp. 1–5 (2010)
5. Bosshart, P., Daly, D., Gibb, G., Izzard, M., McKeown, N., Rexford, J., Schlesinger, C., Talayco, D., Vahdat, A., Varghese, G., Walker, D.: P4: Programming protocol-independent packet processors. SIGCOMM Comput. Commun. Rev. **44**(3), 87–95 (2014). DOI 10.1145/2656877.2656890
6. Bradner, S., McQuaid, J.: RFC 2544: Benchmarking methodology for network interconnect devices. https://www.ietf.org/rfc/rfc2544.txt (1999)
7. Bradner, S.O., Dubray, K., McQuaid, J., Morton, A.: RFC 6815: Applicability statement for RFC 2544: Use on production networks considered harmful. https://www.ietf.org/rfc/rfc6815.txt (2012). DOI 10.17487/RFC6815
8. Costa, L.C., Vieira, A.B., d. B. e. Silva, E., Macedo, D.F., Gomes, G., Correia, L.H.A., Vieira, L.F.M.: Performance evaluation of openflow data planes. In: 2017 IFIP/IEEE Symposium on Integrated Network and Service Management (IM), pp. 470–475 (2017). DOI 10.23919/INM.2017.7987314
9. de Britto e Silva, E., Moura, H., Fanelli, G., d. R. Miranda, M., Macedo, D.F., Vieira, L.F.M., Vieira, M.A.M.: Comparison of data center traffic division policies using sdn. In: NOMS 2018 - 2018 IEEE/IFIP Network Operations and Management Symposium, pp. 1–5 (2018)
10. Demestichas, P., Georgakopoulos, A., Karvounas, D., Tsagkaris, K., Stavroulaki, V., Lu, J., Xiong, C., Yao, J.: 5g on the horizon: Key challenges for the radio-access network. IEEE Vehicular Technology Magazine **8**(3), 47–53 (2013). DOI 10.1109/MVT.2013.2269187
11. Emmerich, P., Raumer, D., Wohlfart, F., Carle, G.: Assessing soft-and hardware bottlenecks in pc-based packet forwarding systems. ICN 2015 p. 90 (2015)
12. Feamster, N., Rexford, J., Zegura, E.: The road to SDN: an intellectual history of programmable networks. In: ACM SIGCOMM Computer Communication Review, vol. 44, pp. 87–98 (2014)
13. Foundation, T.L.: Data plane development kit. https://www.dpdk.org (2018)
14. Heise, P., Geyer, F., Obermaisser, R.: Deterministic openflow: Performance evaluation of sdn hardware for avionic networks. In: 2015 11th International Conference on Network and Service Management (CNSM), pp. 372–377 (2015). DOI 10.1109/CNSM.2015.7367385
15. Inc, S.: Openflow performance testing. http://www.spirent.com/~/media/White\%20Papers/Broadband/PAB/OpenFlow_Performance_Testing_WhitePaper.pdf (2015)
16. Khondoker, R., Zaalouk, A., Marx, R., Bayarou, K.: Feature-based comparison and selection of software defined networking (SDN) controllers.

In: World Congress on Computer Applications and Information Systems (WCCAIS), pp. 1–7 (2014)
17. Koohanestani, A.K., Osgouei, A.G., Saidi, H., Fanian, A.: An analytical model for delay bound of openflow based sdn using network calculus. Journal of Network and Computer Applications **96**, 31 – 38 (2017). DOI https://doi.org/10.1016/j.jnca.2017.08.002. URL http://www.sciencedirect.com/science/article/pii/S1084804517302485
18. Kuźniar, M., Perešíni, P., Kostić, D., Canini, M.: Methodology, measurement and analysis of flow table update characteristics in hardware openflow switches. Computer Networks **136**, 22 – 36 (2018). DOI https://doi.org/10.1016/j.comnet.2018.02.014. URL http://www.sciencedirect.com/science/article/pii/S1389128618300811
19. Lin, Y., Lai, Y., Wang, C., Lai, Y.: Ofbench: Performance test suite on openflow switches. IEEE Systems Journal **12**(3), 2949–2959 (2018). DOI 10.1109/JSYST.2017.2720758
20. Lockwood, J.W., McKeown, N., Watson, G., Gibb, G., Hartke, P., Naous, J., Raghuraman, R., Luo, J.: NetFPGA - an open platform for gigabit-rate network switching and routing. In: IEEE International Conference on Microelectronic System Education (2007)
21. Macedo, D.F., Guedes, D., Vieira, L.F.M., Vieira, M.A.M., Nogueira, M.: Programmable Networks—From Software-Defined Radio to Software-Defined Networking. IEEE Communications Surveys and Tutorials **17**(2), 1102–1125 (2015)
22. Mallon, S., Gramoli, V., Jourjon, G.: Are today's sdn controllers ready for primetime? In: 2016 IEEE 41st Conference on Local Computer Networks (LCN), pp. 325–332 (2016). DOI 10.1109/LCN.2016.60
23. McKeown, N., Anderson, T., Balakrishnan, H., Parulkar, G., Peterson, L., Rexford, J., Shenker, S., Turner, J.: Openflow: Enabling innovation in campus networks. SIGCOMM Comput. Commun. Rev. **38**(2), 69–74 (2008)
24. Moura, H., Alves, A.R., Borges, J.R.A., Macedo, D.F., Vieira, M.A.M.: Ethanol: A software-defined wireless networking architecture for ieee 802.11 networks. Computer Communications **149**, 176 – 188 (2020). DOI https://doi.org/10.1016/j.comcom.2019.10.010. URL http://www.sciencedirect.com/science/article/pii/S0140366419305997
25. Networks, E.: Openflow 1.3 features supported in exos (2018). URL https://gtacknowledge.extremenetworks.com/articles/Solution/OpenFlow-1-3-features-supported-in-EXOS
26. Nguyen-Ngoc, A., Lange, S., Gebert, S., Zinner, T., Tran-Gia, P., Jarschel, M.: Performance evaluation mechanisms for flowmod message processing in openflow switches. In: 2016 IEEE Sixth International Conference on Communications and Electronics (ICCE), pp. 40–45 (2016). DOI 10.1109/CCE.2016.7562610
27. Open Networking Foundation: OpenFlow Switch Specification (2012). Version 1.3.1
28. Open vSwitch. http://openvswitch.org/ (2013)

29. Osman, M., Núñez-Martínez, J., Mangues-Bafalluy, J.: Hybrid sdn: Evaluation of the impact of an unreliable control channel. In: 2017 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN), pp. 242–246 (2017). DOI 10.1109/NFV-SDN.2017.8169866
30. Pantou: Pantou : OpenFlow 1.0 for OpenWRT. http://archive.openflow.org/wk/index.php/Pantou_:_OpenFlow_1.0_for_OpenWRT (2015)
31. Pfaff, B., Pettit, J., Koponen, T., Jackson, E.J., Zhou, A., Rajahalme, J., Gross, J., Wang, A., Stringer, J., Shelar, P., Amidon, K., Casado, M.: The design and implementation of open vswitch. In: Proceedings of the 12th USENIX Conference on Networked Systems Design and Implementation, NSDI'15, pp. 117–130. USENIX Association, Berkeley, CA, USA (2015). URL http://dl.acm.org/citation.cfm?id=2789770.2789779
32. Pica8: Configuring multi-table (2018). URL https://docs.pica8.com/display/PICOS2111cg/Configuring+Multi-Table
33. POX: Pox wiki. https://openflow.stanford.edu/display/ONL/POX+Wiki (2015)
34. Rotsos, C., Sarrar, N., Uhlig, S., Sherwood, R., Moore, A.W.: OFLOPS: An open framework for OpenFlow switch evaluation. In: International conference on Passive and Active Measurement (PAM), pp. 85–95 (2012)
35. Sezer, S., Scott-Hayward, S., Chouhan, P.K., Fraser, B., Lake, D., Finnegan, J., Viljoen, N., Miller, M., Rao, N.: Are we ready for SDN? Implementation challenges for software-defined networks. IEEE Communications Magazine **51**(7), 36–43 (2013). DOI 10.1109/MCOM.2013.6553676
36. Silva, L., Vieira, M., Guedes, D., Ferreira, R.A.: Software-defined networking with services oriented by domain names. Telecommunication Systems **74**, 67–82 (2020). DOI 10.1007/s11235-019-00635-y
37. Song, H.: Protocol-oblivious forwarding: Unleash the power of SDN through a future-proof forwarding plane. In: ACM Workshop on Hot Topics in Software Defined Networking (HotSDN), pp. 127–132 (2013). DOI 10.1145/2491185.2491190
38. Sünnen, D.: Performance evaluation of OpenFlow switch. Semester thesis project report, Swiss Federal Institute of Technology Zurich (2011)
39. Wang, S.Y., Liu, S.Y., Chou, C.L.: Design, implementation and performance evaluation of software openflow flow counters in a bare metal commodity switch. In: 2016 IEEE Symposium on Computers and Communication (ISCC), pp. 651–656 (2016). DOI 10.1109/ISCC.2016.7543811
40. Xiong, B., Yang, K., Zhao, J., Li, W., Li, K.: Performance evaluation of openflow-based software-defined networks based on queueing model. Computer Networks **102**, 172 – 185 (2016). DOI https://doi.org/10.1016/j.comnet.2016.03.005. URL http://www.sciencedirect.com/science/article/pii/S138912861630069X