Performance Evaluation of OpenFlow Data Planes

Leonardo C. Costa*, Alex B. Vieira*
Erik de Britto e Silva¶**, Daniel F. Macedo¶
Geraldo Gomes‡, Luiz H. A. Correia‡, Luiz F. M. Vieira¶
*Universidade Federal de Juiz de Fora (UFMG), Brazil
¶Universidade Federal de Minas Gerais (UFJF), Brazil
‡Universidade Federal de Lavras (UFLA), Brazil
**Universidade Federal de Ouro Preto (UFOP), Brazil

E-mails: leonardocosta@ice.ufjf.br, alex.borges@ufjf.edu.br, {erik,damacedo,lfvieira}@dcc.ufmg.br, gccgomes@sistemas.ufla.br, lcorreia@dcc.ufla.br,erik@decsi.ufop.br

Abstract—**The decoupling of data and control planes of network switches is the main characteristic of Software Defined Networks. The OpenFlow (OF) protocol implements this concept and it is found today in various off-the-shelf equipment. Despite being widely employed in industry and research there is no systematic evaluation of OF data plane performance in the literature. In this paper we evaluate the performance and maturity of the main features of OF 1.0 on nine hardware and software switches. Results show that the performance varies significantly among implementations. For instance, packet delays vary by one order of magnitude among the evaluated switches, while the packet size does not impact the performance of OF switches.**

I. Introduction

The Software Defined Networking (SDN) paradigm is changing the management and operation of computer networks [4]. In fact, network evolution and innovation has been widely facilitated as consequence of the decoupling of the data and control planes in SDN. Among a wide number of advantages, SDN allows the quick deployment of new services, reducing network operating costs.

There is a number of SDN implementations, such as POF and P4 [3], [14]. However, the OpenFlow protocol is considered the *de facto* SDN standard. In part, the success of OpenFlow is due to the fact that vendors can easily implement it on existing switches and routers. As a consequence, we observe an increasing number of SDN networks using OpenFlow, such as Google's WAN [7] and AmLight [5].

Despite being widely employed in industry and academia there is no systematic evaluation of OpenFlow data plane performance in the literature. Such an evaluation is useful for network administrators since they will be able to assess which device is appropriate to deploy on production networks. Moreover, it is also useful to researchers, as they will be able to understand current OpenFlow implementations limitations.

There are several studies in the literature that assess the performance of OpenFlow controllers [1], [2], [13]. Only a few of these focus on the data plane of OpenFlow switches and their implementation [1], [2], [13]. However, these studies perform their evaluation without decoupling the OpenFlow data plane from the SDN controller. The performance of the controller only counts for the first packet of a new flow, while the data plane performance impacts all the packets on a flow. Hence, the highest impact on the user experience will be defined by the quality of the data plane implementation.

In this paper we evaluate the performance and maturity of the main features of OpenFlow 1.0 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. In a controlled environment, we systematically evaluate the OpenFlow implementation by (i) accessing traditional performance metrics such as latency and jitter under certain switch operations, (ii) comparing the switches' performance in both legacy and OpenFlow modes, and (iii) evaluating the performance of OpenFlow operations such as rule matching and querying packet and flow statistics.

Results show that the OpenFlow performance varies significantly. E.g., packet delays vary by one order of magnitude, while the packet size practically does not impact the switch performance. We also note that, in some cases, hardware switches present almost the same performance of software switches, which may indicate a software implementation within the hardware. In sum, we believe that our work provides useful insights about current OpenFlow implementations on real devices, aiding network administrators and researchers to choose the appropriate device according to their needs.

II. Related Work

OpenFlow [9] is an open protocol that allows a central entity to program the flow tables in switches and routers. The protocol defines an API that interconnects the network equipment to a controller [4]. OpenFlow has become the reference SDN platform, being used in the majority of SDN deployments and research [8]. With the growing interest of the industry, more networking equipment from companies like HP, NEC, Pronto, Extreme, Cisco, Brocade, Juniper and Huawei, support OpenFlow [4].

Bianco et al. [2] analyzed the performance of a single switch, virtually deployed on a bridge. Authors compare the performance of OpenFlow versus legacy mode, in IP routing and Ethernet switching use cases. In turn, [1] perform a comparison between different OpenFlow switches. Three OpenFlow platforms were compared, and again, the evaluation methodology depended heavily on controller participation.

Rotsos et al. [12] propose an open and generic framework for OpenFlow switch testing. They developed a tool that tests the interaction between the switches' forwarding engines and the control application of hardware and software switches. However, their framework cannot evaluate the performance of the data plane separately from the control plane.

Spirent sells a host of software for the evaluation network traffic and network testing [6]. It released a white paper on the difficulties of evaluating OpenFlow switches, proposing a methodology for their evaluation and listed frequent evaluation pitfalls.They did not release, to the best of our knowledge, any performance results on OpenFlow switches. In this work, we follow a similar methodology and we also discuss some of the problems found during the execution of tests.

In sum, this work differs from the state of the art as follows: (i) it compares different hardware and software OpenFlow switches. (ii) it analyzes aspects related to the switches' data plane, removing the influence of the controller. (iii) it compares the performance of each switch on both legacy and OpenFlow modes. (iv) the results give significant hints whether OpenFlow is implemented in hardware, or runs in generic processors within the switch.

III. Evaluation Scenario and Methodology

The goal of the evaluation is to analyze different commercial switches in OpenFlow mode vs legacy mode and make a characterization of each switches' performance in OpenFlow mode. The evaluated switches run OpenFlow version 1.0. Despite the existence of newer versions, i.e. version 1.5 at the time of writing this paper, OpenFlow 1.0 is still the most popular. Moreover, most stable firmwares do not implement newer versions of OpenFlow.

Evaluation Topology. Fig. 1 shows the evaluation topology, which presents three machines connected to an OpenFlow switch (a client, a server and a network controller). We have used POX as the controller [11]. Note that, in each scenario, the switches and the testing machines vary. However, test applications are the same for every setup.

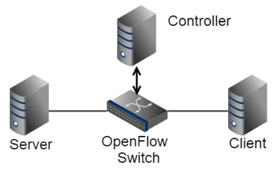

Fig. 1. Topology of the evaluation setup.

Evaluated switches. We compare OpenFlow implementations for both commercial hardware and popular software switches, considering OpenFlow version 1.0. Due to the high cost of OF switches, tests are limited to equipment purchased by the partner universities and research labs. Table I shows the evaluated switches and summarizes their characteristics.

Performance evaluation methodology. To remove the influence of the controller on the results, the controller installs precomputed rules during experiment startup. As a consequence, there are no "packet in" events during the experiments. After the rule setup period, the client sends 10,000 UDP packets to the server, which returns those packets to the client. Then, we calculate the Round Trip Time (RTT) of each request, at a microsecond granularity. We have performed experiments varying the UDP packet sizes from 64, 128 to 256 bytes. These packet sizes are based on related works and RFC 2544. Finally, unless stated otherwise, the results are presented with confidence intervals of 99%.

IV. Results

A. Performance of different match types

First scenario evaluates the effect of different header types. They are based on subsets of individual matches formed among the 12 attributes of OpenFlow version 1.0 (Table II). The goal is to verify if the match execution time is influenced by different attributes. Table III shows the confidence intervals of the average delay for 64-byte packets. For each match type, the performance is similar, suggesting that the type or number of attributes used to perform a match does not substantially affect the packet delay. The only exception occurred for exact matches in the HP switch. A possible explanation for this effect will be presented later on this section.

There is a substantial difference between the confidence intervals when comparing the same match type in different switches. The Extreme switches (X460 and x440), Pica8, Mikrotik, Datacom and NetFPGA card have the smallest average delay values of all switches, between 0.1 and 0.3 ms, approximately. In turn, the delay of Open vSwitch is close to 0.5 ms. HP switch and LinkSys WRT54GL [10] have the worst performance, with the confidence interval edging 1 ms. Figure 2 shows the cumulative distributions of the package delays on each switch for match by IP. It is important to say that this behavior is also observed in the curves of matches by Port, MAC and UDP port. The graph displays this difference in performance among the three groups of switches. Both HP and LinkSys OpenWRT average delay values are strongly influenced by packet delays of more than 2 ms.

The results for the exact match differ slightly from those obtained in match by IP. Again, the HP switch had a lower average delay compared to other match types. Figure 3 shows the cumulative distribution of packet delays for exact matches. The curve corresponding to HP switch is moved further to the left, approaching the Open vSwitch curve. This means that the packet delays in this match, for the HP switch, are relatively lower when compared to delays in other matches on the same switch, by about 21%. This might be due to some HP switch engine that performs exact matches in hardware (e.g. TCAM), in contrast to non-exact matches, which must be performed in software. Furthermore, for the exact match, Extreme switches continued as the fastest and the OpenWRT Linksys implementation as the slowest of the analyzed switches.

TABLE I

LIST OF EVALUATED SWITCHES

Brand	Model	OS/Firmware	No. of ports	CPU	Stable OF version
Extreme	**Summit x460-24p**	ExtremeXOS 15.4.2.8	28	Single Core CPU 500 MHz	1.0, 1.3
Extreme	**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 990 MHz	1.0
Pica8	**P-3297**	PicOS Version 2.1.5	48	P2020 Triumph2	1.0, 1.4
Mikrotik	**Atheros AR9344**	RouterOS 6.34.2	24	Single Core CPU 600 MHz	1.0
Open vSwitch	**OvS 2.3.0**	Linux Ubuntu 14.04	-	Intel Core i7 CPU 2.80 GHz	1.0
HP	**HP2920-24G**	Firmware K 15.5 i	24	Tri Core ARM1176 625 MHz	1.0, 1.3
LinkSys	**WRT54GL**	OpenWRT + Pantou OF	4	Broadcom BCM5352 200 MHz	1.0, 1.3

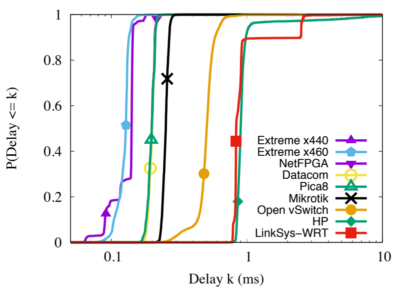

Fig. 2. Cumulative distributions of delays for match by IP.

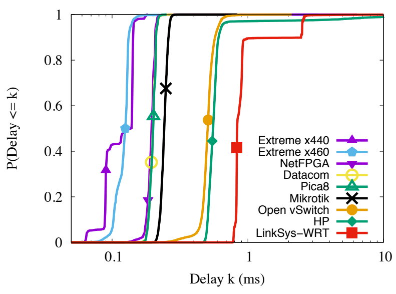

Fig. 3. Cumulative distributions of delays for exact match.

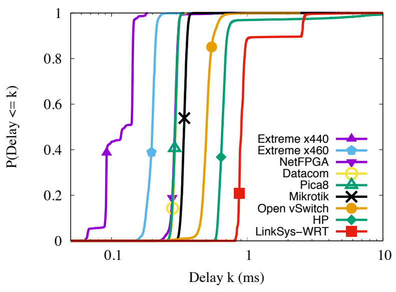

Fig. 4. Cumulative distributions of delays for exact match on 256-byte packets.

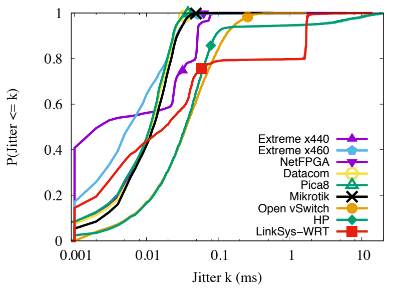

Fig. 5. Cumulative distributions of jitter for 64-byte packets.

TABLE II

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

To test the influence of the packet size in matches, the tests were repeated with a packet size of 256 bytes. For increased packet sizes, the HP, Extreme X460, Mikrotik, Pica8, Datacom and NetFPGA had the highest increases in delay. Other switches had no significant increases in their delays, as shown in Figures 4 (256 byte packets) and 3 (64 byte packets).

Figure 5 shows the cumulative distribution of packet jitter for exact matches. The jitter on Extreme switches is noticeably lower when compared to others, staying below 1 ms. In turn, the HP switch curve resembles the curve of the analyzed software switches. This can corroborate the suspicion that HP's implementation does not perform hardware acceleration.

B. Legacy and OpenFlow switch modes performance

The second scenario compares the performance of each switch when operating on both OpenFlow and legacy modes. To simulate the input-output forwarding on OF mode, the controller installs a rule that matches the input port and forwards the packets to a fixed output port.

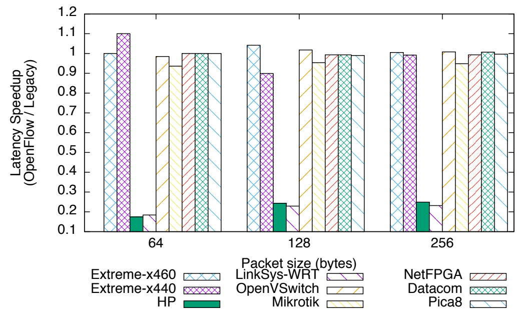

Fig. 6. Switch performance on OF mode when compared to legacy mode.

Figure 6 shows the switch performance on OF mode when compared to the switch performance on legacy mode, for three different packet sizes. The Speedup is calculated by dividing the legacy mode latency by the OF mode latency.Results show that OF mode performance is similar to legacy mode performance on Extreme switches (X460 and x440), Open vSwitch,

TABLE III
CONFIDENCE INTERVALS FOR DELAY IN DIFFERENT MATCH TYPES (IN MS).

Switch	**Match type**				
	Port	**MAC**	**IP**	**UDP port**	**Exact**
Extreme x460-24p	0.125 - 0.127	0.124 - 0.126	0.124 - 0.126	0.124 - 0.126	0.121 - 0.125
Extreme x440-48p	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
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 - OpenWRT	1.025 - 1.051	1.017 - 1.045	1.019 - 1.049	1.017 - 1.046	1.018 - 1.047

Mikrotik, Pica8, Datacom and NetFPGA. This indicates that those OF firmwares exploit all the hardware capabilities of the switch. Meanwhile, the OF mode performance on both HP and Linksys-OpenWRT switches are lower than the legacy mode performance, reaching only 20% of normal performance in both, approximately. This is an indication that the OF firmware does not exploit well the available hardware, suggesting for example the use of SRAM instead of TCAM.

C. *Single and multi-flow performance*

The third scenario evaluates the switch performance when the flow table has only one rule, returning always "match", compared to a table with multiple rules. The objective is to verify if the performance can be influenced by the flow table internal organization (e.g. if matching is done in parallel or sequentially, if matching uses hashing or binary search).

The first step in this test was to discover the maximum number of rules that could be installed on each switch in OF mode and in legacy mode, as shown in Table IV. For most switches, this was performed experimentally, by sending OF rules until a table full message was received, since the manufacturers did not disclose this information on their datasheets or by e-mail.

In most cases, the number of supported rules in OF is smaller than the number of rules supported in legacy mode. This may occur because of the additional attributes that must be stored. For example, the Extreme switches (X460 and x440) store rules on access-control-lists (ACLs) in hardware for OF and legacy mode. This may explain the similar performance of the two operating modes in the previous scenario. On OF mode, however, as there are more attributes or operations to be stored, the ACL size is set to "double", so fewer rules are supported on this mode.

Software implementations have fewer rule limitations. Open vSwitch has a hash table with storage capacity of up to 1 million rules for exact matches only, however it limits the number of rules according to rule types and real time network conditions. To store rules for non-exact (wildcard) matches, Open vSwitch uses a linear table of up to 100 positions.

On the HP switch, the OF firmware creates a new flow table in software. This new table has a large rule storage capacity, and can't be disabled. This also points that the HP switch must employ SRAM to store OF rules, and could explain its poor performance on previous scenarios.

Table V shows the confidence intervals for average packet delays in two situations: when the flow table contains only one rule, and in the case where the flow table has multiple rules, and the match occurs on the rule installed last. The amount of installed rules is equal to the maximum number of rules found in the previous experiment, listed in Table IV. For most of the switches, the intervals intersect themselves or have a difference of a few microseconds amongst themselves. This may indicate that the large amount of installed rules and their arrangement in the flow table does not significantly overload the switches or cause a substantial increase on delays. The only exception is LinkSys-OpenWRT, which has an average increase in packet delay of around 0.2 ms.

D. *Flowstats and portstats operations performance*

The fourth scenario evaluates the performance of each switch for flowstats and portstats operations. In the tests, the controller requests statistics (portstats or flowstats) every second to the switch, and the controller measures the response time. The switch is experiencing load generated using Iperf[1]. For the light load, Iperf generates UDP packets at 1MB/s, while for the heavy load Iperf generates packets at its maximum rate.

During the setup of the experiment, it was verified that some switches become unstable when their table is full. Thus, we empirically determined the maximum number of rules that each switch accepts while maintaining a stable behavior. In fact, this number is around 10% lower than the number presented in Table IV, and almost all switches hung when issuing flowstats/portstats requests with nearly full tables.

Figures 7 and 8 show the cumulative distributions for flowstats delays for one single rule, and also for an almost full flow table, respectively. In both cases, the system is operating under a heavy workload. Note that there is a considerable increase in flowstats delay, for all switches, when comparing an almost empty flow table to a nearly full table. This is an expected behavior, since the flowstats message returns information for each active rule.

In turn, the effect of the number of installed rules is reduced for portstats. In fact, as shown in Figures 9 and 10, this influence occurs only for some switches (Extreme x460 and x440, Pica8, NetFPGA and Datacom).

Table VII summarizes flowstats delay, presenting the confidence intervals for the mean delays of the tested switches, under different workloads. The increase of system workload influenced flowstats delays in different ways. For example, Mikrotik and LinkSys switches had a considerable increase in

[1]https://iperf.fr/

TABLE IV
NUMBER OF SUPPORTED RULES ON LEGACY AND OPENFLOW MODES.

Switches	Number of rules in legacy mode	Number of rules in OF mode
Extreme x460-24p	2048	1200
Extreme x440-48p	1024	248
NetFPGA	16	124
Datacom	32768	2051
Pica8	12000	4096
Mikrotik	16318	389
Open vSwitch	1000000	750000
HP	2048	16000
LinkSys - OpenWRT	100	100

TABLE V
CONFIDENCE INTERVALS OF DELAYS IN TABLES WITH ONE AND MULTIPLE RULES (IN MS).

Switches	One rule	Multiple rules
Extreme x460-24p	0.121 - 0.125	0.124 - 0.128
Extreme x440-48p	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
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 - OpenWRT	1.028 - 1.055	1.215 - 1.242

TABLE VI
CONFIDENCE INTERVALS FOR THE DELAY IN HEADER REWRITES (IN MS).

Switches	Type of rewrite					
	No rewrite	MAC	IP	VLAN priority	IP ToS	UDP port
Extreme x460-24p	0.125 – 0.127	0.125 – 0.129	-	0.125 – 0.128	-	-
Extreme x440-48p	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	-	-	-	-
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

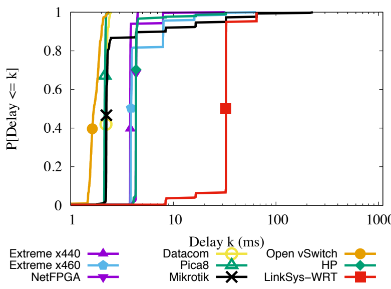

Fig. 7. Flowstats delays for an almost empty table under heavy workload.

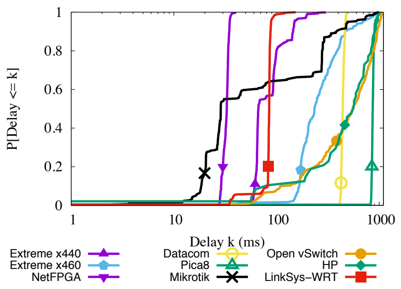

Fig. 8. Flowstats delays for a nearly full table under heavy workload.

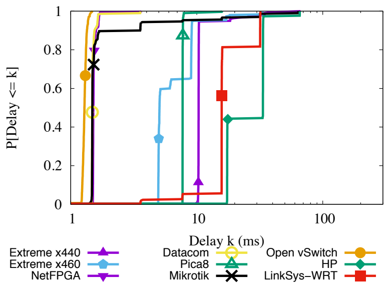

Fig. 9. Portstats delays for an almost empty table under heavy workload.

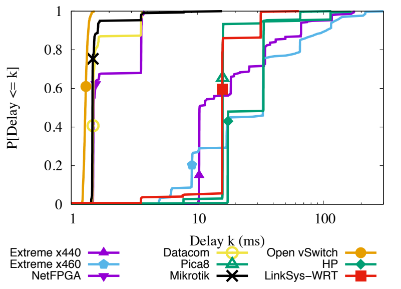

Fig. 10. Portstats delays for a nearly full table under heavy workload.

their flowstats delay. Open vSwitch, HP, Pica8, Datacom and NetFPGA kept their delays virtually unchanged, despite the increased workload. Interestingly, the Extreme switches (x460 and x440) experienced a decrease on their mean flowstats delays. Meanwhile, the OpenWRT implementation had a severe performance degradation for the heavy workload.

In turn, Table VIII shows the means of portstats delay. In this case, the increase in system workload also influenced the portstats delay, mainly for Extreme x460, HP and LinkSys switches. It is worth mentioning that flowstats is more sensitive to the flow table size and workload than portstats. Furthermore, the workload contributes more to the performance than the number of installed rules. Except for HP and OpenWRT, the number of flows did not affect the portstats performance.

Unlike forwarding performance, for portstats and flowstats commands there is no noticeable performance gap among hardware and software implementations. This is more significant for flowstats, where software switches performed sometimes better than hardware switches.

E. Discussion

Overall, none of the hardware switches fully implemented the optional features of the OpenFlow 1.0 specifications. This is very relevant to researchers, who might assume that all the features as well as match types are supported. One example is the VLAN attribute, which is used in many switches to separate the control plane port from the data plane ports.

Another issue is stability. Some data planes become unstable

TABLE VII
CONFIDENCE INTERVALS FOR FLOWSTATS DELAYS (IN MS).

Switches	Table with one rule		Nearly full table	
	Light load	Heavy load	Light load	Heavy load
Extreme x460-24p	4.430 - 5.762	4.646 - 6.042	275.726 - 358.626	256.013 - 332.985
Extreme x440-48p	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
Pica8	1.876 - 2.440	1.868 - 2.430	751.442 - 977.370	756.914 - 984.488
Mikrotik	7.340 - 9.548	5.518 - 7.176	13.659 - 17.767	130.058 - 169.160
Open vSwitch	1.763 - 2.293	1.502 - 1.954	459.776 - 598.012	483.018 - 628.242
HP	4.048 - 5.264	4.224 - 5.494	375.217 - 699.857	363.165 - 720.289
LinkSys - OpenWRT	7.659 - 9.961	28.582 - 37.176	31.619 - 41.125	71.672 - 93.220

TABLE VIII
CONFIDENCE INTERVALS FOR PORTSTATS DELAYS (IN MS).

Switches	Table with one rule		Nearly full table	
	Light load	Heavy load	Light load	Heavy load
Extreme x460-24p	5.987 - 7.787	6.288 - 8.178	25.574 - 33.264	35.053 - 45.591
Extreme x440-48p	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
Pica8	6.791 - 8.833	6.791 - 8.833	19.700 - 25.622	18.373 - 23.897
Mikrotik	1.480 - 1.924	2.860 - 3.720	3.017 - 3.923	1.445 - 1.879
Open vSwitch	1.261 - 1.641	1.120 - 1.456	1.258 - 1.636	1.131 - 1.471
HP	15.728 - 20.456	23.706 - 30.834	15.904 - 20.686	23.862 - 31.036
LinkSys - OpenWRT	4.480 - 5.826	15.876 - 20.650	4.627 - 6.019	15.288 - 19.884

when the rule table is nearing its capacity, others simply crash when their limit is reached, and others handle a limited number of OF messages per second.

Regarding the performance of the OF operations, it is not possible to choose where rules will be installed (in the TCAM or in SRAM). Switches supporting both types of memory automatically choose where to install the rule depending on the match type (exact or inexact). This might change in OF 1.3 data planes, which will support multiple tables.

The choice of which OF switch to use, for research or on a production network, depends heavily on the application, since the data planes have different maximum number of OF rules, support for packet rewrites, as well as performance. Overall, software switches are more compliant with the OF specification and support large forwarding tables, however their forwarding delay may be up to one order of magnitude slower than a hardware switch. However, hardware switches usually do not support large flow tables, and support packet rewrites on less fields.

When analyzing only the hardware-based switches, the Extreme switches had the best performance overall. The HP switch, on the other hand, presented a performance similar to that of software switches. This gives strong evidence that the HP switch might perform rule matching on the embedded CPU without resorting to any hardware optimization.

Finally, the overall conclusion of the paper is that researchers and practitioners must be aware of the limitations of existing implementations. If full OF compliance is necessary (e.g. for research), software-based implementations are recommended. However, for production networks, the administrator must carefully analyze the requirements of their networks to identify the most adequate switch.

V. CONCLUSION

This paper evaluated the performance of hardware and software OpenFlow data planes. The aim of this study was to evaluate the maturity of existing implementations, since OpenFlow has already become a technology with many successful use cases on production networks, and its use tends to increase in the coming years.

Results showed that the switches implement OpenFlow in different ways. Some of them implement it in hardware, others in software. Moreover, the implementation and effective use of the equipment's hardware significantly affects the final performance. Although hardware switches typically operate one order of magnitude faster than software switches, their implementation lacks many features that are taken for granted on many SDN papers: large flow tables, support for header rewriting and matching on all L3 and L4 fields.

ACKNOWLEDGMENT

The authors would like to thank Brazilian Founding agencies CAPES, CNPq, FAPEMIG. 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 and RNP.

REFERENCES

[1] M. Appelman and M. De Boer, "Performance analysis of OpenFlow hardware," Semester thesis project report, University of Amsterdam, February 2012.

[2] A. Bianco, R. Birke, L. Giraudo, and M. Palacin, "Openflow switching: Data plane performance," in *IEEE International Conference on Communications (ICC)*, 2010, pp. 1–5.

[3] P. Bosshart, D. Daly, G. Gibb, M. Izzard, N. McKeown, J. Rexford, C. Schlesinger, D. Talayco, A. Vahdat, G. Varghese, and D. Walker, "P4: Programming protocol-independent packet processors," *SIGCOMM Comput. Commun. Rev.*, vol. 44, no. 3, pp. 87–95, Jul. 2014.

[4] N. Feamster, J. Rexford, and E. Zegura, "The road to SDN: an intellectual history of programmable networks," in *ACM SIGCOMM Computer Communication Review*, vol. 44, 2014, pp. 87–98.

[5] J. Ibarra, J. Bezerra, H. Morgan, L. F. Lopez, I. Cox, D. A., M. Stanton, I. Machado, and E. Grizendi, "Benefits brought by the use of OpenFlow/SDN on the AmLight intercontinental research and education network," in *IFIP/IEEE International Symposium on Integrated Network Management (IM)*, 2015, pp. 942–947.

[6] S. Inc, "Openflow performance testing," http://www.spirent.com/~/media/White\%20Papers/Broadband/PAB/OpenFlow_Performance_Testing_WhitePaper.pdf, 2015.

[7] S. Jain, A. Kumar, S. Mandal, J. Ong, L. Poutievski, A. Singh, S. Venkata, J. Wanderer, J. Zhou, M. Zhu, J. Zolla, U. Hölzle, S. Stuart, and A. Vahdat, "B4: Experience with a globally-deployed software defined WAN," in *Proceedings of ACM SIGCOMM*, 2013, pp. 3–14.

[8] D. F. Macedo, D. Guedes, L. F. M. Vieira, M. A. M. Vieira, and M. Nogueira, "Programmable networks-from software-defined radio to software-defined networking," *IEEE Communications Surveys and Tutorials*, vol. 17, no. 2, pp. 1102–1125, 2015.

[9] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "Openflow: Enabling innovation in campus networks," in *SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 2, 2008, pp. 69–74.

[10] Pantou, "Pantou : OpenFlow 1.0 for OpenWRT," http://archive.openflow.org/wk/index.php/Pantou_:_OpenFlow_1.0_for_OpenWRT, 2015.

[11] POX, "Pox," https://openflow.stanford.edu/display/ONL/POX+Wiki, 2015.

[12] C. Rotsos, N. Sarrar, S. Uhlig, R. Sherwood, and A. W. Moore, "OFLOPS: An open framework for OpenFlow switch evaluation," in *Int. Conference on Passive and Active Measurement (PAM)*, 2012.

[13] D. Snnen, "Performance evaluation of OpenFlow switch," Semester thesis project report, Swiss Federal Institute of Technology Zurich, February 2011.

[14] H. Song, "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)*, 2013, pp. 127–132.