Performance Benchmarking of SDN Experimental Platforms

Philippos Isaia
Department of Computer Science
Loughborough University
Loughborough, LE11 3TU, UK
Email: p.isaia@lboro.ac.uk

Lin Guan
Department of Computer Science
Loughborough University
Loughborough, LE11 3TU, UK
Email: l.guan@lboro.ac.uk

***Abstract*—There is a huge number of SDN experimental platforms available such as simulators, emulators and actual testbeds, each of them having different performance metrics. This paper presents a series of performance tests, that can be performed in each of the available platforms, in order to evaluate and rank them in various performance categories. These tests cover performance categories such as experiment setup/teardown time, resources needed in the form of CPU and RAM, as well as the fair use and fair share of those resources by the experimental platform. In addition, ping delay, response failure rate and scalability are also measured. All the performance tests presented in this paper have been implemented in Mininet emulator in order to evaluate its performance. After the data analysis, the most noticeable results are (i) response failure increases as the number of links increases, in some cases by 95%, (ii) CPU load balancing is more efficient as the number of nodes increases and (iii) initial ping delay is huge compared to average ping delay, in some cases up to 1725 times larger. Finally, performance results indicate that Mininet has several scalability issues.**

I. Introduction

Software-defined Networking (SDN) [1] is a computer networking architecture that decouples the control plane from the data plane in traditional networking devices (switches), allowing direct programmability. Each SDN switch has an external controller which inspects data packets and creates rules that are then added to the switch's flow table. SDN research provided numerous new ideas that solved many networking problems, allowing networks to evolve and meet modern demands.

In order for research proposals and ideas to be validated, they have to be tested in simulators, emulators or actual testbeds using an SDN implementation, usually the OpenFlow [2] protocol. All the experimental environments lack of a way of performance testing and rating. It is very important for each research idea to be tested in an environment suitable to the experimental needs in order to provide useful results.

This paper presents a number of tests for simulation and emulation experimental platforms, which evaluate their performance in specific areas. As a result, using these tests a ratings table can be created, allowing researchers to evaluate and rate all the available platforms and find the one that suits their needs. Furthermore, this paper goes a step ahead to perform and analyse all the proposed tests on the Mininet [3] platform.

978-1-4673-9486-4/16/$31.00 © 2016 IEEE

The remainder of this paper is organised as follows. Section II, presents the background of this work as well as previous efforts. All the performance tests of our proposal are described in Section III. Section IV describes the setup for performing the tests on Mininet, whereas the results are presented in Section V. Finally, a conclusion of the work is presented in Section VI.

II. Background

There are plenty of simulators and emulators for SDN. Some noticeable simulator examples are fs-sdn [4], NS-3 [5] and EstiNet 9.0 [6]. The most popular and tested SDN experimental platform is Mininet [3] emulator. When it comes to testbeds, there are plenty of OpenFlow-Enabled or OpenFlow-Only switches as well as NetFPGA [7] and Pantou [8] project which uses a custom router firmware called OpenWrt [9], in order to allow Low-End home routers to act as OpenFlow switches.

In the area of benchmarking these SDN specific environments there is not much done other than the work presented by the developers of each environment. In this section the majority of benchmarks presented come directly from their developers, and cannot be used in a useful comparison due to the fact that they do not share the same parameters, topologies and methodologies.

In [10] several basic benchmarks are presented for EstiNet such as the main memory consumption using incremental number of OpenFlow switches, the Average Ping Delay (APD) together with its Standard Deviation as well as the rate of No Response Failure. Furthermore, it suggests that EstiNet solves several problems existing in Mininet, which result from the fact that Mininet is highly depended on the Operating System Scheduler. This is partially true due to the fact that EstiNet solves the problem if and only if the simulation mode is used. In the emulation mode it still has the same problems as Mininet.

In paper [11] an OpenFlow Extension for OMNeT++ [12] using the INET Framework [13] is presented. Unfortunately, it is not as comprehensive as [10], since it only states the mean Round-Trip Time (RTT) of different spanning trees, which is one of the problems that solves.

In [3] Mininet benchmarks such as end-to-end bandwidth, setup time, stop time and memory usage are presented. They cannot be considered as comprehensive due to the fact that each result comes from a totally different topology. It would have been more comprehensive to compare results coming from a same shape topology with different number of switches or nodes present in the topology than compare topologies with totally different structures. Furthermore, all of the results come from a virtual machine running on an Apple MacBook Pro, and there is no indication in the paper about any effect the laptop's operating system has on the results. The most comprehensive Mininet benchmark comes from a 2012 technical report [14], which takes four typical topologies and tests them with different number of switches, giving out metrics like throughput and fairness.

III. Performance Tests

In order to review, benchmark and rate the available experimental platforms, a series of different experiments in specific areas have to be performed. First of all, each of the topologies chosen have to isolate one or more bottlenecks of each platform. In addition, each topology has to be compared to results coming from a same shape topology with the only difference of being smaller or larger in number of components present. It is unfair to compare for example a Tree with a Fat Tree topology.

In order to compare each topology and find the performance of each platform, here are all the metrics that have been taken into account:
1) Setup Time, *2)* Teardown Time, *3)* CPU Usage, *4)* Scalability *5)* CPU Cores Load Balancing (CCLB), *6)* RAM Usage, *7)* Initial Ping Delay (IPD), *8)* Average Ping Delay (APD), *9)* No Response Failure Rate (NRFR), *10)*Fair Share of Resources (FSR)

Setup and *Teardown* times are an important measure of *a)* the efficiency in the use of the available resources and *b)* scalability. CPU usage is important in order to evaluate *a)* the amount of system resources needed for each topology and for a specific number of nodes, *b)* scalability. In addition, *c)* it confirms if a higher profile system will result in more accurate experimental results. Furthermore, *d)* it allows elimination of inconsistent readings caused by a fully loaded CPU.

The CPU usage tests consist of two readings, one is the *Initial CPU Usage* (I-CPU), which is the CPU usage by the platform after it has created the experimental topology but before it has begun with the specified tests. The second reading is the *CPU Usage During Experimentation* (CPU-DE), meaning is the average CPU usage during the time that the specified tests are executed.

CPU Cores Load Balancing (CCLB) is a measure of *a)* scalability and *b)* the ability of the platform to initialise all the available CPU cores equally (i.e. good use of a multi-core CPU). CCLB is measured by calculating the standard deviation of core usage (for all the available cores) at each time interval and then finding the average of those standard deviation values for the whole duration of the experiment. Therefore, a value close to 0 indicates an excellent load balancing. *Random-access Memory* (RAM) will indicate *a)* how much resource hungry the platform is, *b)* if it will benefit from systems with more RAM and *c)* scalability.

Ping delay is an important indication on how close the platform is to real life implementation. There are two important types of Ping delays, the *Initial Ping Delay* (IPD) and the *Average Ping Delay* (APD). The IPD is highly affected by the time it takes for the controller to add a flow table rule to the OpenFlow switch. The APD is the average of all the ping delays excluding IPD. Using an SDN experimental platform without initialising a controller (i.e. eliminating the installation of rules in the network device flow table) is unrealistic and cannot be compared to real-life. On the other hand adding the huge IPD into the APD will result in a significant increase that will not reflect the entire test as well as real-life implementation with pro-active controllers. Thus, splitting the Ping delay into IPD and APD is the best solution.

No Response Failure Rate (NRFR) is the percentage of failed attempts for communication during Ping test. This appears regularly in emulations due to the fact that each test has to be allocated some processing time by the Operating System (OS) Scheduler. The OS Scheduler is not designed specifically for the experimental platform and as a result it does not allocate processing time efficiently from the platform's point of view, therefore it fails to perform as in real world.

Finally, *Fair Share of Resources* (FSR) indicates platform's performance in resource allocation. In this paper FSR is represented by the Coefficient of Variation (CV) of the delay when all the hosts in the topology perform Ping command at the same time. CV is given by $C_v = \sigma/\mu * 100$, where σ is the standard deviation of delay and μ is the average delay. In some topologies is impossible to get FSR results, for example topology T3 (see Section IV) has only 2 hosts running ping therefore FSR results cannot be obtained.

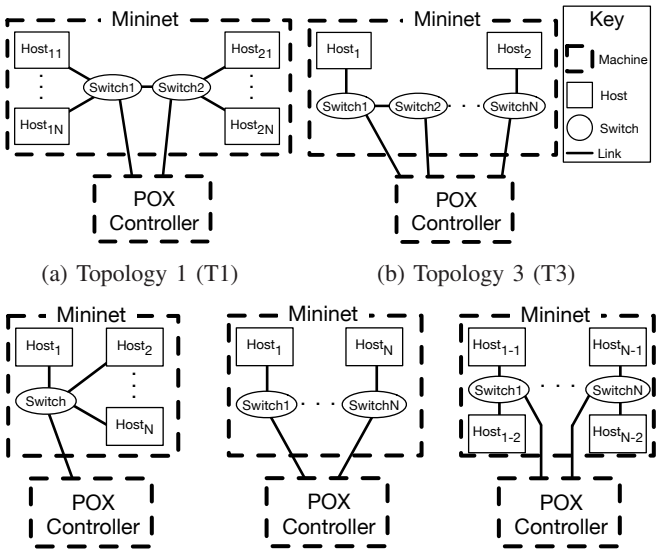
IV. Implementation

Due to the fact that Mininet is widely used in OpenFlow experimentation, it has been used to perform all the tests described in Section III. Five different topologies have been used with variable number (N) of switches or nodes, where N had the values of 1, 100, 500 and 1000. In order to minimise experimental error, each experiment was repeated 30 times.

The first topology *T1* (Fig. 1a) examines Mininet's performance with a bottleneck link, the link between Switch 1 and Switch 2. For Ping and Bandwidth (using iPerf [15]) tests $Host_{1N}$ was communicating with $Host_{2N}$ resulting in a one-to-one connection with the appropriate switch for each host.

The second topology *T2* (Fig. 1c) has a bottleneck link but compared to T1 it forms a one-to-many connection where $Host_1$ pings a number of other nodes (i.e. from $Host_2$ up to $Host_N$).

The topologies *T3* and *T4* (Figs. 1b and 1d), are both "Linear" meaning one switch connected next to each other, and examine the effect of several switches on both the ping delay and FSR. The difference between the two topologies



(a) Topology 1 (T1) (b) Topology 3 (T3)

(c) Topology 2 (T2) (d) Topology 4 (T4) (e) Topology 5 (T5)

Fig. 1: Experimentation Topologies

TABLE I: Mininet Experimental Machines Specifications

	Low-End	**High-End**
OS	Ubuntu 14.04.1 LTS	
CPU Vendor	Intel	
Architecture	x86-64	
Cores	4	8
CPU GHz per core	2.4	
Cache (MB)	4	
RAM (MB)	8192	16384
Hard Disk (GB)	32	50

is the fact that T3 has only two hosts, one connected to $Switch_1$ and one to $Switch_N$ whereas T4 has as many hosts as switches with each switch having one host connected to it. In T3, ping and iPerf were performed between hosts $Host_1$ and $Host_2$ whereas in T4 were performed between hosts $Host_1$ and $Host_N$. Even though the extra hosts in T4 ($Host_2$ *up to* Host_{N-1}) are not actively participating in the experiment, they are used to examine if some resources are still assigned to them.

Finally, *T5* (Fig. 1e) is a Host-Switch-Host topology meaning each of the switches is connected to two separate hosts, and none of the switches are connected together. The scope is mainly scalability and sharing of resources evaluation. Ping and iPerf tests were performed between hosts $Host_{N-1}$ and $Host_{N-2}$ and switch $Switch_N$ had the request handling responsibility.

All of the tests have been performed on two different systems (see TABLE I), in order to evaluate Mininet's behaviour in limited resources at first in the Low-End machine and then when it has more resources in the High-End machine. In addition, since Mininet supports several types of switches, all of the experiments were repeated using Open vSwitch [16] (OVS), Indigo Virtual Switch (IVS) [17] and Mininet's Reference Switch (MRS). An initial analysis indicated that both OVS and MRS results were identical and IVS faced a lot of problems, giving inconsistent results in topologies with high number of nodes. Due to the fact that OVS is currently the leading industry and research switch, in this paper only OVS results are presented. Finally, in all of the results, the experimental error is presented in the form of standard deviation.

V. Analysis

In T1, setup and teardown times are not affected by the system resources since both Low and High end systems results are identical, both of them increase linearly as the number of hosts in the system increases. Furthermore, in both cases the teardown time is always higher than the setup time ranging from 3 up to 7 times higher. I-CPU snows an anomaly, at *N=1* and *N=100* is higher in High-End system even though it should be higher in the Low-End system for any number of *N*. CPU-DE is always higher in the High-End system even though there are more resources available. In the Low-End system at *N=1* CCLB is not very efficient, but once the number of hosts increases it becomes more efficient. In the High-End system slightly more RAM was used which in combination with CPU readings means that Mininet took an advantage of the availability of resources. In addition, even though APD is identical for both systems at any number of hosts N, when it comes to *N=500* and *N=1000* the IPD is lower in the High-End system. Finally, NRFR shows failed responses only at *N=1000*, with the Low-End system reaching 86.4% whereas High-End system is at 42.9%. All of the results for T1 are summarised in Table II.

TABLE II: Topology 1 (T1) Summary

Low-End System				
Hosts (N)	1	100	500	1000
Setup (s)	0.045 ± 0.002	1.21 ± 0.06	7.13 ± 0.36	19.79 ± 0.99
Teardown (s)	0.156 ± 0.008	7.37 ± 0.368	46.33 ± 2.32	110.53 ± 5.527
I-CPU (%)	2.74 ± 0.48	3.37 ± 0.699	10.39 ± 6.19	16.61 ± 6.06
CPU-DE (%)	3.35 ± 1.91	5.38 ± 0.19	24.27 ± 0.67	35.38 ± 3.28
CCLB	2.21	0.22	0.78	3.79
RAM (MB)	253 ± 4.39	478 ± 5.51	1394 ± 100.82	2549 ± 189.68
IPD (ms)	8.76 ± 0.49	12.6 ± 1.01	83.8 ± 8.02	93.1 ± 5.19
APD (ms)	0.06 ± 0.13	0.07 ± 0.14	0.09 ± 0.13	0.099 ± 0.17
NRFR (%)	0.0	0.0	0.0	86.35
FSR (%)	62.30	279.02	517.21	47.30
High-End System				
Hosts (N)	1	100	500	1000
Setup (s)	0.077 ± 0.004	1.309 ± 0.065	6.76 ± 0.34	18.93 ± 0.95
Teardown (s)	0.235 ± 0.0118	7.48 ± 0.374	45.83 ± 2.291	116.75 ± 5.84
I-CPU (%)	4.74 ± 0.41	5.43 ± 0.70	9.35 ± 2.25	11.97 ± 0.44
CPU-DE (%)	6.31 ± 0.26	8.62 ± 1.60	32.88 ± 3.39	40.48 ± 2.62
CCLB	0.28	1.71	3.62	2.80
RAM (MB)	316 ± 8.54	559 ± 50.90	1542 ± 141.6	2847 ± 167.9
IPD (ms)	5.55 ± 0.35	16 ± 1.36	57.8 ± 5.44	66.8 ± 3.88
APD (ms)	0.06 ± 0.17	0.07 ± 0.18	0.08 ± 0.12	0.08 ± 0.11
NRFR (%)	0.0	0.0	0.0	42.94
FSR (%)	124.00	328.05	513.85	530.07

In T2 both the setup and teardown times were significantly lower than T1 but once again the teardown time was significantly higher than the setup time. I-CPU usage was low for both Low and High-End systems, but at *N=500* and *N=1000* the High-End system uses half the CPU Low-End system uses.

CCLB was identical to T1, indicating that at high number of hosts Mininet becomes more efficient. RAM usage was less compared to T1 but again in the High-End system more of the available RAM was used compared to the Low-End system.

Furthermore, even though the High-End system performed better in APD, in IPD Low-End system performed better especially in scenarios with higher number of switches (at *N=500* Low-End IPD was 61% lower than High-End and at *N=1000* 51% lower). In addition, NRFR is identical for both systems and it is also much lower than in T1. Full summary of T2 is provided in Table III.

Comparing T1 to T2 it is clear that the more nodes are present in the system the more time it takes for both setup and teardown, therefore T2 is faster. For the exact same reason, T2 uses less RAM. In addition the IPD is much higher for T1 due to the fact that the controller has to setup flow table rules for two switches, but APD is unaffected.

TABLE III: Topology 2 (T2) Summary

Low-End System				
Hosts (N)	1	100	500	1000
Setup (s)	0.059 ± 0.003	0.66 ± 0.033	1.23 ± 0.062	7.71 ± 0.39
Teardown (s)	0.098 ± 0.005	2.96 ± 0.15	7.39 ± 0.37	45.97 ± 2.29
I-CPU (%)	2.83 ± 0.25	3.30 ± 0.68	6.77 ± 1.29	12.20 ± 3.1
CPU-DE (%)	3.63 ± 0.1	5.71 ± 0.59	31.11 ± 0.49	74.12 ± 3.75
CCLB	0.12	0.68	2.84	4.33
RAM (MB)	227 ± 0.88	342 ± 34.39	464 ± 15.24	1411 ± 141.8
IPD (ms)	3.65 ± 0.25	5.5 ± 0.49	8.8 ± 0.58	23.8 ± 1.5
APD (ms)	0.07 ± 0.09	0.07 ± 0.06	0.08 ± 0.07	0.08 ± 0.06
NRFR (%)	0.0	0.0	0.0	30.91
FSR (%)	56.37	376.23	351.68	231.76
High-End System				
Hosts (N)	1	100	500	1000
Setup (s)	0.103 ± 0.005	0.652 ± 0.033	3.32 ± 0.166	7.06 ± 0.353
Teardown (s)	0.122 ± 0.006	3.697 ± 0.185	22.29 ± 1.115	44.41 ± 2.22
I-CPU (%)	2.09 ± 0.06	2.33 ± 0.33	3.3 ± 1.302	6.18 ± 0.329
CPU-DE (%)	0.15 ± 0.32	3.81 ± 0.83	26.38 ± 2.35	45.67 ± 1.31
CCLB	0.34	0.889	2.51	1.397
RAM (MB)	396 ± 9.17	438 ± 30.52	941 ± 19.49	1570 ± 155.77
IPD (ms)	3.19 ± 0.22	5.96 ± 0.56	22.7 ± 1.71	48.6 ± 2.91
APD (ms)	0.05 ± 0.08	0.04 ± 0.05	0.06 ± 0.07	0.06 ± 0.04
NRFR (%)	0.0	0.0	0.0	30.91
FSR (%)	75.01	387.75	274.27	221.74

In T5, the setup time is higher than the teardown time. Compared to T1 and T2, the main difference is the number of switches in the topology, therefore it seems that switches take more time to setup and less time to teardown. Furthermore, T5 has from 3 to 3000 hosts but it seems that it is not the number of hosts that affects the result since both T1 and T2 have increasing number of hosts but teardown remains higher than setup time.

In T5 I-CPU usage in Low-End system is significantly higher than in High-End system whereas CPU-DE for both systems is almost identical. CCLB followed the same pattern as in T1 and T2, becoming more efficient as the number of switches increases. Also RAM usage is almost identical for both systems but in almost all the cases the High-End system uses slightly more RAM. APD is about the same for both systems, as well as the IPD except from the *N=1000* experiment where High-End performs slightly better. The value of NRFR remains at 0 which shows that all the ping packets reached their destination, therefore they are not affected by the number of switches in the network. Full summary of T5 is provided in Table IV.

T3 confirmed that it takes more time to setup a switch than a host and much less time to teardown a switch than a host. This topology provided two unexpected results, and the first

TABLE IV: Topology 5 (T5) Summary

Low-End System				
Switches (N)	1	100	500	1000
Setup (s)	0.135 ± 0.007	5.274 ± 0.264	127.54 ± 6.34	514.4 ± 257.2
Teardown (s)	0.094 ± 0.005	9.228 ± 0.46	69.22 ± 3.46	153.5 ± 7.67
I-CPU (%)	3.65 ± 0.03	6.01 ± 1.29	28.79 ± 5.51	38.17 ± 7.69
CPU-DE (%)	5.62 ± 0.18	14.24 ± 0.73	36.39 ± 1.42	56.95 ± 3.28
CCLB	0.21	0.84	1.64	3.78
RAM (MB)	237 ± 18.08	569 ± 26.38	2123 ± 75.99	3735 ± 340.49
IPD (ms)	3.34 ± 0.32	13.4 ± 1.16	62.2 ± 5.34	150 ± 15.64
APD (ms)	0.07 ± 0.07	0.07 ± 0.05	0.08 ± 0.09	0.09 ± 0.11
NRFR (%)	0.0	0.0	0.0	0.0
FSR (%)	98.04	321.82	805.00	636.72
High-End System				
Switches (N)	1	100	500	1000
Setup (s)	0.125 ± 0.006	5.75 ± 0.288	99.31 ± 4.965	500.57 ± 25.03
Teardown (s)	0.124 ± 0.006	11.18 ± 0.559	73.71 ± 3.685	164.78 ± 8.239
I-CPU (%)	2.65 ± 0.03	3.35 ± 1.29	9.1 ± 2.51	11.84 ± 7.69
CPU-DE (%)	4.13 ± 0.19	9.84 ± 0.71	34.13 ± 3.46	52.12 ± 3.92
CCLB	0.20	0.75	3.69	4.19
RAM (MB)	310 ± 13.14	649 ± 27.12	2078 ± 12.49	3893 ± 288.58
IPD (ms)	3.4 ± 0.26	15.9 ± 0.88	59.8 ± 5.38	138 ± 9.68
APD (ms)	0.03 ± 0.03	0.09 ± 0.27	0.07 ± 0.07	0.08 ± 0.13
NRFR (%)	0.0	0.0	0.0	0.0
FSR (%)	79.11	323.72	491.38	838.85

one is the fact that at *N=500* and *N=1000* the Low-End system performed better than the High-End system in both setup and teardown times.

TABLE V: Topology 3 (T3) Summary

Low-End System				
Switches (N)	1	100	500	1000
Setup (s)	0.109 ± 0.005	5.73 ± 0.287	67.85 ± 3.39	135.7 ± 6.79
Teardown (s)	0.112 ± 0.006	5.73 ± 0.309	41.87 ± 2.094	83.75 ± 4.187
I-CPU (%)	3.17 ± 0.875	25.34 ± 2.35	44.72 ± 14.35	49.46 ± 16.31
CPU-DE (%)	4.95 ± 1.30	16.96 ± 10.19	43.12 ± 10.34	70.41 ± 16.38
CCLB	1.51	11.77	15.02	15.42
RAM (MB)	231 ± 2.58	366 ± 22.05	866 ± 68.47	1642 ± 103.53
IPD (ms)	4.92 ± 0.27	789 ± 50.63	2300 ± 225.8	4740 ± 305.7
APD (ms)	0.07 ± 0.07	16.49 ± 120.26	51.72 ± 306.8	143.35 ± 683.07
NRFR (%)	0.0	0.0	64.79	88.35
High-End System				
Switches (N)	1	100	500	1000
Setup (s)	0.104 ± 0.005	5.47 ± 0.273	185.5 ± 9.276	230.95 ± 15.70
Teardown (s)	0.137 ± 0.007	7.205 ± 0.36	42.09 ± 2.104	94.65 ± 4.732
I-CPU (%)	1.89 ± 0.001	10.03 ± 0.54	20.88 ± 1.31	33.08 ± 3.46
CPU-DE (%)	2.03 ± 0.829	8.39 ± 8.35	25.695 ± 5.53	24.93 ± 3.769
CCLB	0.88	4.03	5.91	8.93
RAM (MB)	308 ± 12.17	433 ± 54.41	961 ± 51.43	1737 ± 158.41
IPD (ms)	1.93 ± 0.15	2183 ± 185.09	3653 ± 312.18	7304 ± 965.2
APD (ms)	0.06 ± 0.07	16.001 ± 119.35	103.4 ± 462.4	353.7 ± 748.2
NRFR (%)	0.0	0.0	64.08	85.48

I-CPU had a significant increase from *N=1* to *N=500* in both Low and High-End systems. In all the experiments the High-End system used less I-CPU than the Low-End. CPU-DE was almost identical for both systems. CCLB value indicated once again that load balancing becomes more efficient as the number of switches increases. RAM followed the trend of the previous scenarios which finds the High-End system always using more RAM than the Low-End system. The second unexpected result and the most significant one is APD and IPD. Except from *N=1*, in all the other number of switches the Low-End performed much better than the High-End. Another noticeable result is the standard deviation of APD which is significantly higher than the average, meaning that the readings have a huge difference between them. NRFR value is identical for both systems, at *N=100* is zero and then tends to increase as *N* increases. Full summary of T3 is provided in Table V.

In T4 after a certain number of switches the setup time becomes higher than the teardown time. At *N=1000* setup time

is about 50 times higher than teardown time. Compared to T3, both Low and High-End systems performed roughly the same.

I-CPU had a less significant increase from *N=1* to *N=100* compared to T3. In all the experiments both Low and High-End systems used about the same I-CPU. Following the pattern of all the previous topologies, CCLB becomes more efficient as the number of switches increases. RAM didn't follow the trend of the previous scenarios since both systems used about the same RAM except at *N=1000* switches where the High-End system used 200MB less than the Low-End system. Furthermore, in this topology the High-End system performed much better in IPD at large number of *N* whereas at low number of *N* the Low-End system performed better. In APD High-End system performed better except at *N=100* where Low-End performed better. The number of NRFR showed that at high number of *N* it increases dramatically. All of the results are summarised in Table VI.

TABLE VI: Topology 4 (T4) Summary

Low-End System				
Switches (N)	1	100	500	1000
Setup (s)	0.057 ± 0.003	8.24 ± 0.412	310.68 ± 15.53	10382.5 ± 519.13
Teardown (s)	0.106 ± 0.005	12.006 ± 0.6	87.91 ± 4.396	213.43 ± 10.67
I-CPU (%)	5.52 ± 0.377	12.27 ± 3.54	18.35 ± 4.71	19.97 ± 1.66
CPU-DE (%)	6.03 ± 1.77	17.03 ± 14.64	58.69 ± 1.88	70.33 ± 5.05
CCLB	2.04	2.10	2.14	3.61
RAM (MB)	327 ± 30.20	577 ± 14.03	1643 ± 230.16	3329 ± 47.31
IPD (ms)	2.4 ± 0.146	1141 ± 66.298	16946 ± 1002.77	130573 ± 3429.63
APD (ms)	0.06 ± 0.05	15.07 ± 115.76	2226.27 ± 4546.2	6042 ± 7134.65
NRFR (%)	0.0	0.0	33.34	95.1
FSR (%)	65.62	410.37	713.42	821.49
High-End System				
Switches (N)	1	100	500	1000
Setup (s)	0.089 ± 0.004	6.72 ± 0.336	299.23 ± 14.96	10397.95 ± 519.898
Teardown (s)	0.128 ± 0.006	13.54 ± 0.677	74.78 ± 3.739	214.29 ± 10.715
I-CPU (%)	5.02 ± 0.677	12.27 ± 1.299	18.35 ± 4.59	20.12 ± 1.85
CPU-DE (%)	2.20 ± 1.09	7.78 ± 3.5	27.65 ± 5.86	25.72 ± 3.93
CCLB	1.16	3.75	4.21	6.26
RAM (MB)	327 ± 25.58	585 ± 74.37	1678 ± 83.55	3129 ± 73.5
IPD (ms)	3.06 ± 0.249	2628 ± 188.76	16855 ± 1149.33	73093 ± 3553.72
APD (ms)	0.04 ± 0.05	24.63 ± 172.35	2045.34 ± 4353.71	4385 ± 5632.8
NRFR (%)	0.0	0.0	6.25	63.29
FSR (%)	79.32	386.64	564.80	728.24

Summarising all the results, Mininet balances the load equally to all the available CPU cores efficiently with some minor exceptions (T3). Mininet makes good use of CPU, but at some experiments it could have used less CPU. NRFR is affected by Mininet but not as much as by congestion control. IPD is highly affected by the number of nodes in an experiment, the more the nodes the longer it takes for the flow installation processes, whereas APD is not affected except in the case of Linear topologies. FSR increases in both Low and High-End systems as *N* increases meaning Mininet is not allocating resources fairly. Mininet RAM usage is fairly small, but Mininet is hungry for RAM meaning that if there is more RAM available, Mininet will use it even though it can run the same topology with less RAM.

VI. Conclusion

In this paper a series of performance tests that can be used in order to examine various SDN experimental platforms are presented. These performance tests indicate the time needed for a platform to create and destroy a topology, the CPU percentage used by each topology both at topology creation and during experimentation as well as the RAM needed. Additionally, both Initial and Average Ping Delays are measured as well as the number of ping packets that failed to reach the destination. Finally, the fairness in sharing of resources by the platform is measured.

Using five topologies that had the purpose of exposing several bottlenecks and critical performance areas, Mininet Emulator was tested using the proposed set of performance metrics. From the results it is concluded that a) setup time is highly affected by the number of switches, at low number of switches teardown time is much higher than setup time whereas exactly the opposite happens in scenarios with high number of switches. b) Mininet uses more RAM for the same topology if more RAM is available. c) The number of failed ping packets increases as the number of links included in the packets path increases. d) Initial Ping Delay is huge compared to Average Ping Delay. e) Load balancing between CPU cores becomes more efficient as the number of nodes, in the topology, increases.

References

[1] O. N. Fundation, "Software-defined networking: The new norm for networks," *ONF White Paper*, 2012.

[2] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "OpenFlow: enabling innovation in campus networks," *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 2, pp. 69–74, 2008.

[3] B. Lantz, B. Heller, and N. McKeown, "A network in a laptop: rapid prototyping for software-defined networks," in *Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks*, p. 19, ACM, 2010.

[4] M. Gupta, J. Sommers, and P. Barford, "Fast, accurate simulation for SDN prototyping," in *Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking*, pp. 31–36, ACM, 2013.

[5] T. Henderson, M. Lacage, G. Riley, M. Watrous, G. Carneiro, T. Pecorella, and others., "Network Simulator 3." https://www.nsnam.org. Accessed: 07-10-2015.

[6] EstiNet, "EstiNet." http://www.estinet.com. Accessed: 07-10-2015.

[7] J. Naous, D. Erickson, G. A. Covington, G. Appenzeller, and N. McKeown, "Implementing an openflow switch on the netfpga platform," in *Proceedings of the 4th ACM/IEEE Symposium on Architectures for Networking and Communications Systems*, pp. 1–9, ACM, 2008.

[8] "Pantou." http://archive.openflow.org/wk/index.php/Pantou_:_OpenFlow_1.0_for_OpenWRT. Accessed: 07-10-2015.

[9] "OpenWrt." https://openwrt.org. Accessed: 07-10-2015.

[10] S.-Y. Wang, C.-L. Chou, and C.-M. Yang, "EstiNet OpenFlow network simulator and emulator," *Communications Magazine, IEEE*, vol. 51, no. 9, pp. 110–117, 2013.

[11] D. Klein and M. Jarschel, "An OpenFlow extension for the OMNeT++ INET framework," in *Proceedings of the 6th International ICST Conference on Simulation Tools and Techniques*, pp. 322–329, ICST, 2013.

[12] A. Varga and R. Hornig, "An overview of the OMNeT++ simulation environment," in *Proceedings of the 1st international conference on Simulation tools and techniques for communications, networks and systems & workshops*, p. 60, ICST, 2008.

[13] A. Varga, R. Hornig, B. Seregi, L. Meszaros, and Z. Bojthe, "INET Framework." https://inet.omnetpp.org. Accessed: 07-10-2015.

[14] N. Handigol, B. Heller, V. Jeyakumar, B. Lantz, and N. McKeown, "Mininet Performance Fidelity Benchmarks." http://hci.stanford.edu/cstr/reports/2012-02.pdf. Accessed: 07-10-2015.

[15] "NLANR/DAST : Iperf - the TCP/UDP bandwidth measurement tool." http://sourceforge.net/projects/iperf/. Accessed: 07-10-2015.

[16] Nicira and Citrix, "Open vSwitch, An Open Virtual Switch," *Synergy*, 2009.

[17] D. Talayco and R. Lane, "Indigo Virtual Switch," *Project Floodlight*, 2013.