



# ABSTRACTION LAYER FOR IMPLEMENTATION OF EXTENSIONS IN PROGRAMMABLE NETWORKS

Collaborative project co-funded by the European Commission within the Seventh Framework Programme

Grant agreement no: 317880
Project acronym: ALIEN
Project full title: "Abstraction Layer for Implementation of Extensions in programmable Networks"
Project start date: 01/10/12
Project duration: 24 months

# Deliverable D5.3
# Experimental-driven research - Appendix IV (OpenFlow Performance experiment results)

## Version 7.9

**Due date:** 30/10/2014

**Submission date:** 12/11/2014

**Editor:** Mariusz Żal (PUT), Maider Huarte (EHU)

**Author list:** Jannusz Kleban, Marek Michalski, Remigiusz Rajewski, Mariusz Zal (PUT), Damian Parniewicz, Łukasz Ogrodowczyk, Bartosz Belter, Artur Binczewski (PSNC).

**Dissemination Level**

| | | |
|---|---|---|
| ☒ | **PU:** | Public |
| ☐ | **PP:** | Restricted to other programme participants (including the Commission Services) |
| ☐ | **RE:** | Restricted to a group specified by the consortium (including the Commission Services) |
| ☐ | **CO** | Confidential, only for members of the consortium (including the Commission Services) |


## <THIS PAGE IS INTENTIONALLY LEFT BLANK>

# Table of Contents

# Figure summary

# Table summary


## <THIS PAGE IS INTENTIONALLY LEFT BLANK>

# Executive Summary

This document complements the deliverable D5.3 entitled “Experimental-driven research” with details concerning testing scenario steps and all materials like logs, screenshots, packet dumps collected during performing of experimentation tests. We highly encourage to read first the D5.3 report which contains overview of the experiments, goals to be achieved, description of experiments environment and also discussion regarding experiment validation and analysis of obtained results.


## <THIS PAGE IS INTENTIONALLY LEFT BLANK>

# 1 OpenFlow performance testing

## 1.1 Testing scenarios

### 1.1.1 PacketIn test

For every packet that does not have a matching flow entry or if a packet matches an entry with a forward to In_Port action or when flow table is empty, an asynchronous packet-in event is sent to the controller. If the switch has sufficient memory to buffer packets that are sent to the controller, the packet-in events contain some fraction of the packet header (by default 128 bytes) and a buffer ID to be used by the controller, when it is ready for the switch to forward the packet. Switches that do not support internal buffering (or have run out of internal buffering) must send the full packet to the controller as part of the event.

ALIEN platforms are based on existing network devices and don't support such packet buffering. In In_Port action the OpenFlow controller receives whole data packets. It is very important to check, how many In_Packet actions can be served by Alien platforms and how length of data packets affect the delay of packet transfer. In order to measure In_Packet action throughput and packet delay, the following scenario, presented in Figure 1-1, was prepared. When ALIEN-OFLOPS application receives OFPT_HELLO messages, OFPT_FLOW_MOD message is sent. OFPT_FLOW_MOD contains OFPFC_DELETE action, which is applied to all installed flows. Simultaneously, through dev1 ALIEN-OFLOPS starts generate UDP packets, which contain (in payload field) sequence number and generation time. Since the OpenFlow switch flow table is empty, switch sends OFPT_PACKET_IN messages containing UDP packets. Basing on reception of OFPT_PACKET_IN message and generation time of conveyed UDP packet we can calculate delay of In_Packet action. Additionally, basing on sequence numbers loss probability is calculated. Test is repeated for different UDP packets sizes and different time interval between packets.

### 1.1.2 PacketOut test

This test scenario, presented in Figure 1-2, is similar to previous one. In this test ALIEN-OFLOPS application generates sequence of OFPT_PACKET_OUT messages, which convey UDP packets. As previously, UDP packets are marked by sequence numbers and OFPT_PACKET_OUT message generation time. The OFPT_PACKET_OUT message contains also of_port number determining the output port of the action. Basing on reception of UDP packets and generation time conveyed in this packet we can calculate delay of Out_Packet action. Moreover basing on sequence numbers loss probability is calculated. Test is repeated for different UDP packet sizes and different time interval between packets.

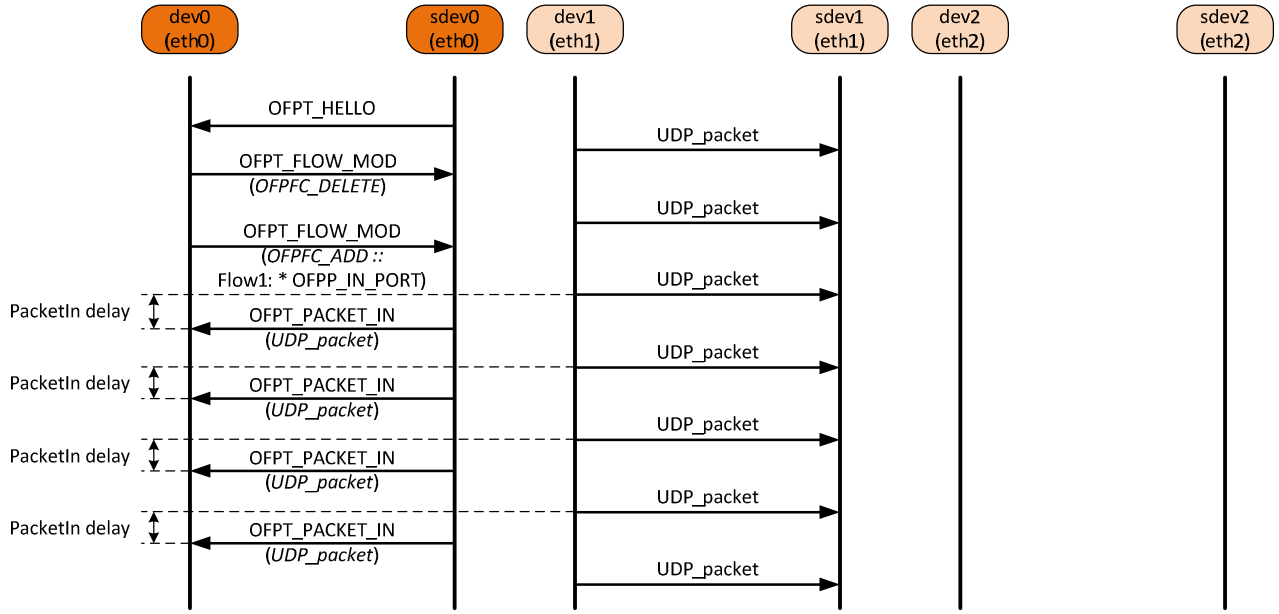


**Figure 1-1 Packet_In test message flow diagram**

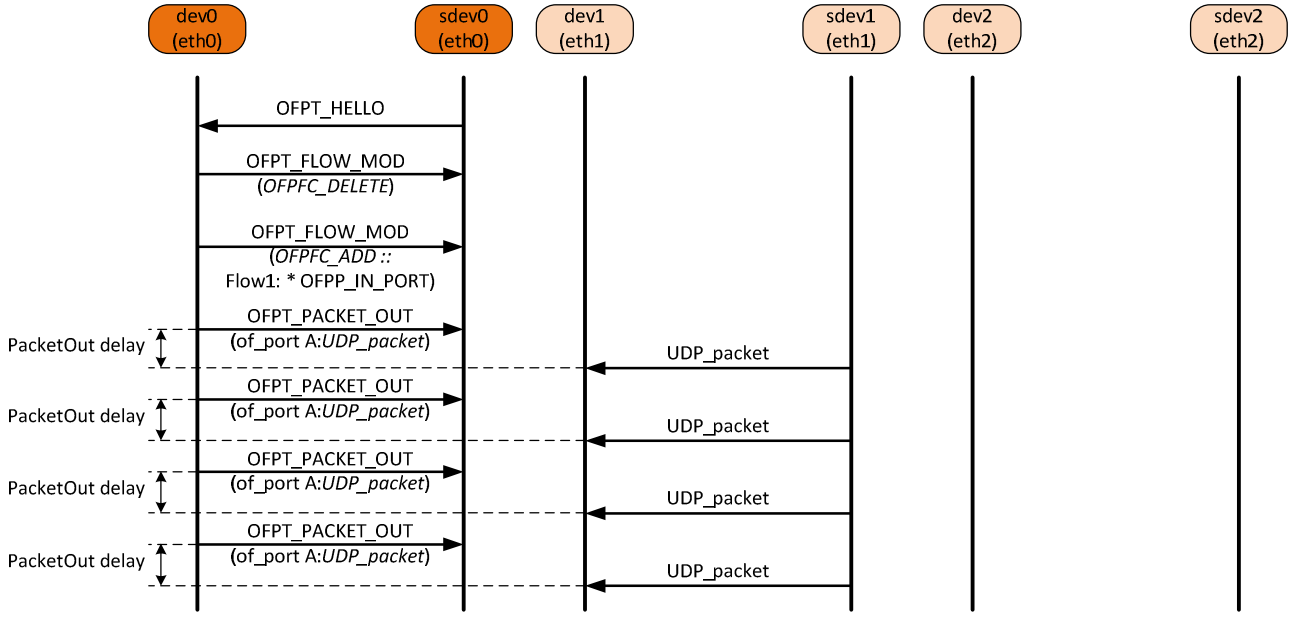


**Figure 1-2 Packet_Out test message flow diagram**

### 1.1.3 Path_delay test

The Path_delay test allows determining path delay in control and data plane. In case of control plane both, OFPT_ECHO_REQUEST and OFPT_ECHO_REPLY messages are used. The data field in echo request message must be copied into echo replay data field. When OFPT_ECHO_REQUEUST is generated into this field the current system time is written. The difference between OFPT_ECHO_REPLY message reception time and the time included in data field of this message, indicates the control path delay.

In the case of data plane path delay UDP packets are marked by the sequence numbers and packet generation time. In order to reduce packet processing time the flow table stores only one defined flow entry matches for incoming UDP packets and directs them into specific output port.

Path delay in control plane is determined without any parameters. Path delay in data plane is studied for different packet sizes and different time intervals between packets. The message flow diagram for this experiment is presented in Figure 1-3.

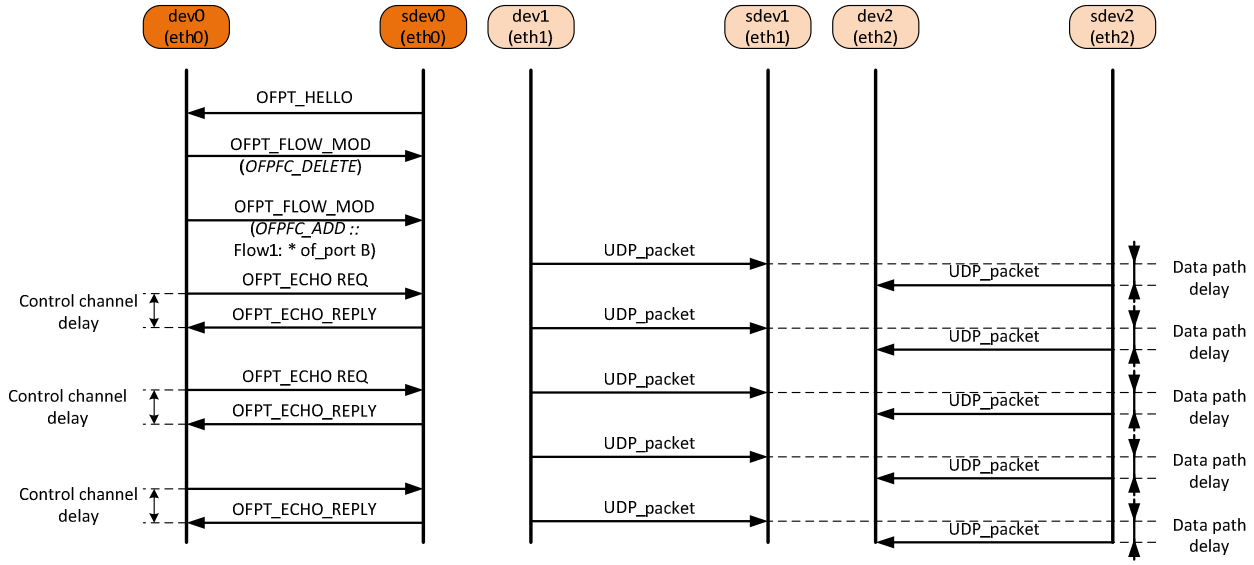


**Figure 1-3 Path delay in control and data plane test message flow diagram**

### 1.1.4 Flow_add test

The aim of this test is to determine time required to add the given number of flows. Only exact match flows entries are considered. This test procedure is repeated several times in order to achieve reliable test results. During whole test time ALIEN-OFLOPS application generates UDP packets containing in data field packet generation time and send these messages through dev1 port. In the first step Alien platform flow table is cleared. Next, the determined number of ADD_FLOW operation is generated and sent. It starts time counter. Only the last added flow entry matches generated UDP packets. According to action bound with this flow, the generated packets which appear on sdev1 port are directed to sdev2 port. When some packet occurs on dev2 port triggers the test procedure repetition and stops time counter. Time counter shows how long flows are installed. The message flow diagram for this experiment is presented in Figure 1-4.

### 1.1.5 Flow_mod test

The aim of this test is similar to Flow_add test. In Flow_mod test time required to modify the given number of table flow entries is determined. The message flow diagram for this experiment is presented in Figure 1-5.

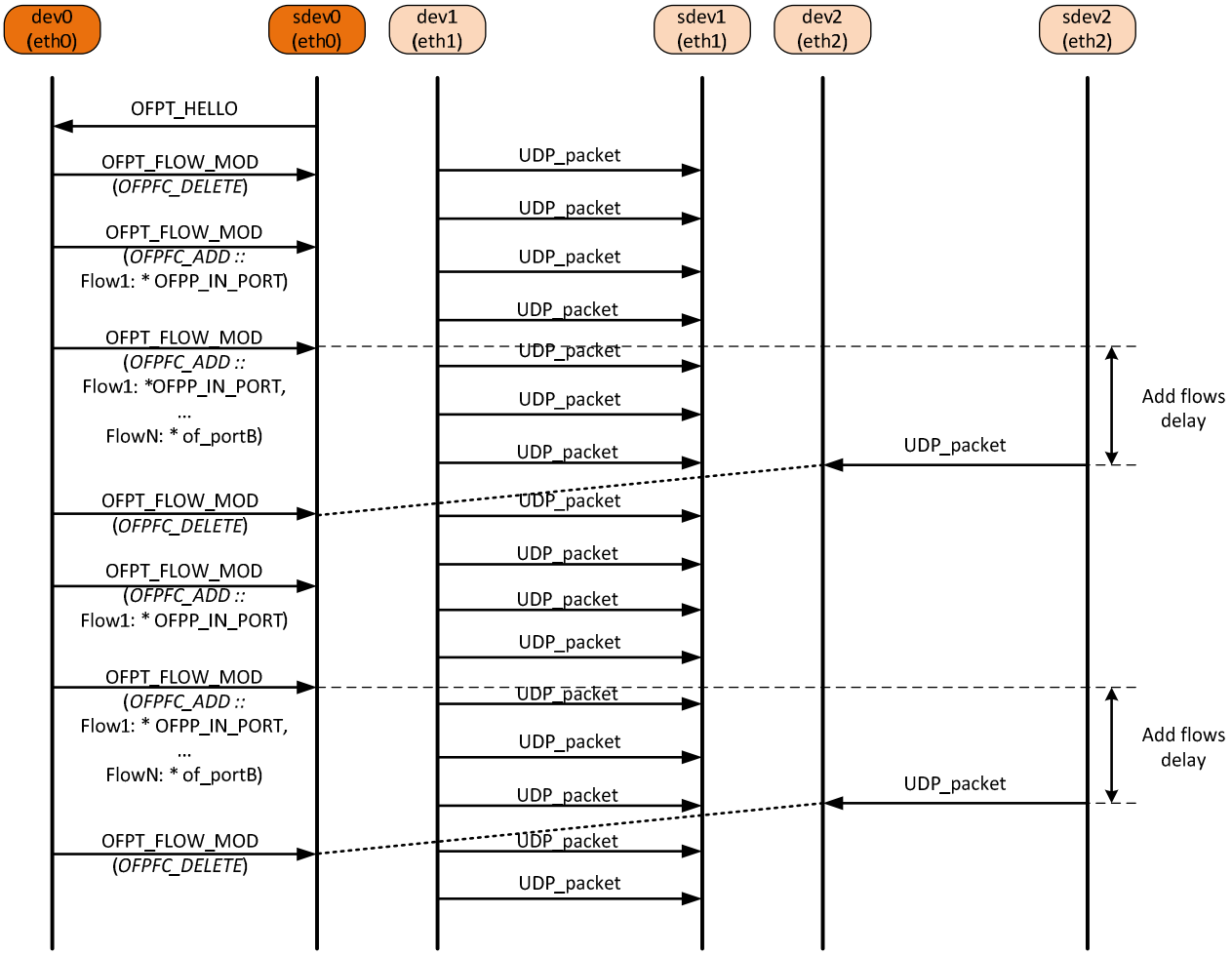


**Figure 1-4 The Add_flow test message flow diagram**

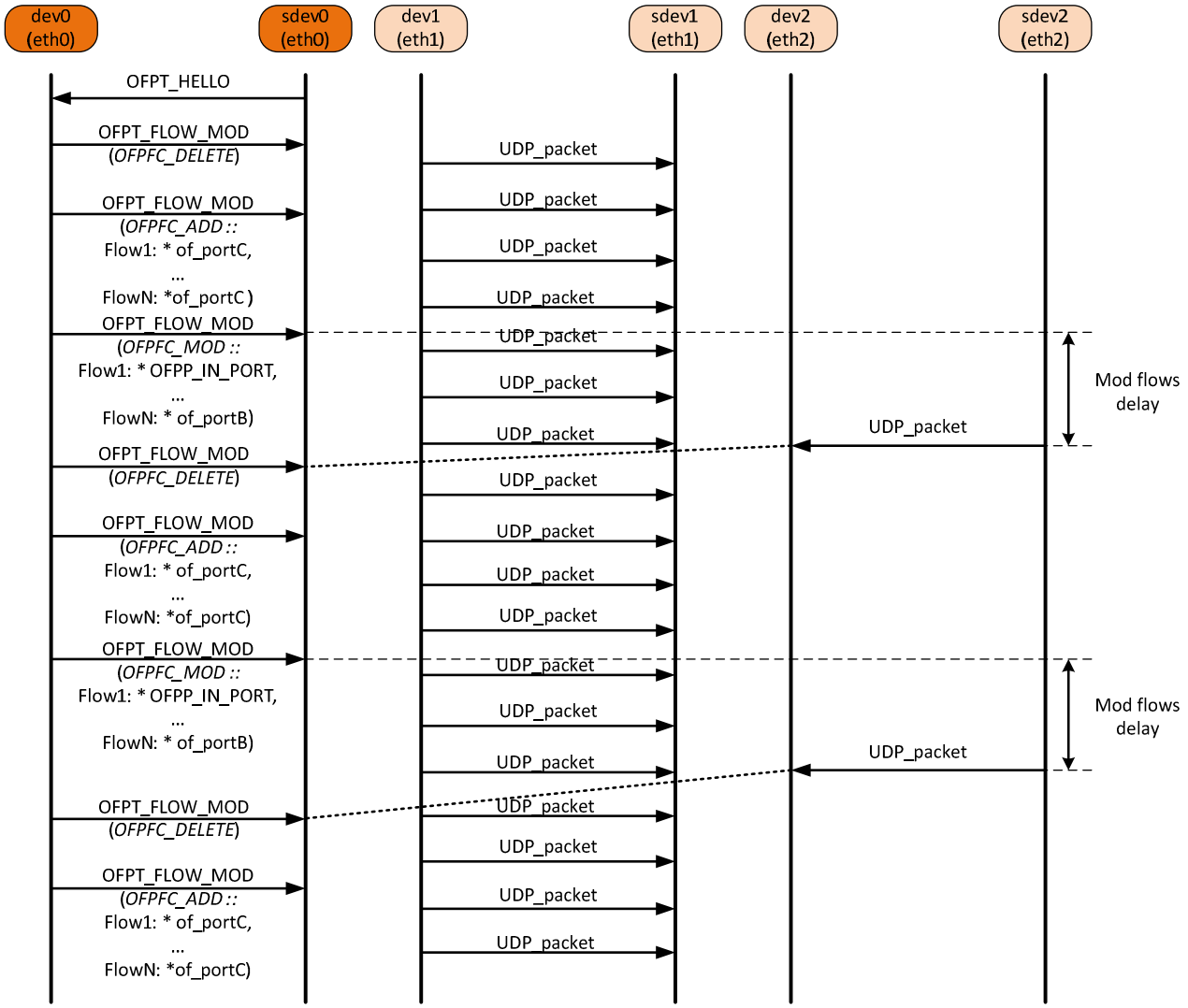


**Figure 1-5 The Mod_flow test message flow diagram**

## 1.2 Experiments results

### 1.2.1 NetFPGA

Performance tests of HAL implementation for the NetFPGA platform were done using the ALIEN OFLOPS framework and scenarios described in Section 5.1 of deliverable D.5.3. To test performance of OpenFlow-capable hardware based on 1G NetFPGA card (ports nf2c0 and nf2c1) the card was connected to OpenFlow testing platform (ports nf2c0 and nf2c1) also based on 1G NetFPGA card with ALIEN OFLOPS installed (see Figure 1-6). Communication channel between the NetFPGA platform and the OpenFlow testing platform was established using the OpenFlow v1.0 protocol as depicted in Figure 1-6. All tests were performed on machines available from range of OFELIA's IP addresses and can be managed remotely, however, the data connection have to be realized directly with no intermediate switches or another L2 bridges, (they can

be logically transparent, but some additional time delay could appears on each of such network element). So, the tester and tested equipment should be connected physically and they should be in the same facility.

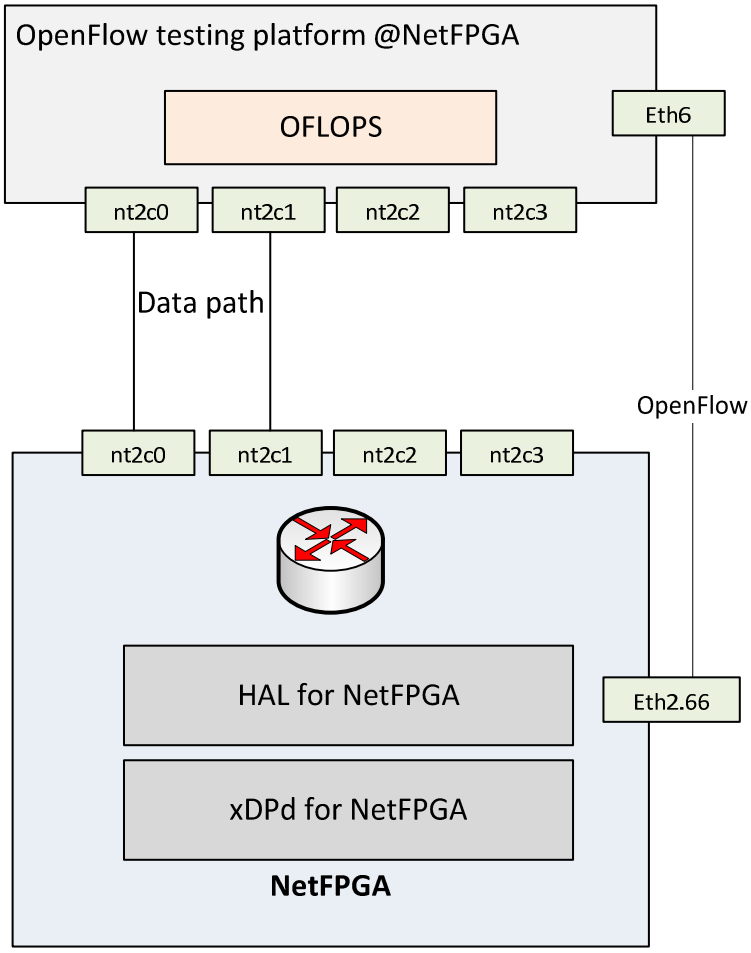


**Figure 1-6 Performance tests of NetFPGA HAL implementation - connections between devices**

xDPd was configured to work with ALIEN OFLOPS as the OpenFlow controller (xDPd.conf):

```
#Host A
config:{
   openflow:{
     logical-switches:{
             #Name of the switch dp0
             dp0:{
                     #Most complex configuration
                     dpid = "0x1201021606613301"; #Must be hexadecimal
                     version = 1.0;
                     description="Switch A";
                     #Controller connection(s)
                     controller-connections:{
                             main:{
                                     remote-hostname="10.216.66.12";
                                     remote-port = 6633;
```

```
                        };
                    };
                    #Reconnect behaviour
                    reconnect-time=1; #seconds
                    #Tables and MA
                    num-of-tables=1;
                    #Physical ports attached to this logical switch.
                    #This is mandatory
                    #The order and position in the array dictates the
                    # number of
                    # 1 ->nf2c0, 2 ->nf2c1, 3 ->nf2c2, 4 ->nf2c3
                    ports = ("nf2c0", "nf2c1", "nf2c2", "nf2c3");
            };
        };
    };
};
```

The following tests were done for the NetFPGA platform:

- Packet_In,
- Packet_Out,
- Data_path_delay,
- Control_channel_delay,
- Add_flow (with extension, in order to determine the maximum number of installed flows).

Results of the performance test for the NetFPGA platform were compared with the results obtained for the OpenVSwitch (OVS) OpenFlow switch implementation installed as a virtual machine. These results are presented on charts as well as in tables.

### 1) The Packet_In test

For Packet_In test the ALIEN OFLOPS framework generates UDP packets with a constant bit rate. Before the test the flow table was cleaned. So, UDP packets don't match any flow in the flow table stored in the NetFPGA card. All unmatched packets are forwarded to the controller (in this case to the ALIEN OFLOPS) in order to take the decision what to do with this new flow. Since the test must be repeated several times for a given set of parameters, ALIEN OFLOPS does not install any flow. The data flow within the testbed for this experiment is presented in Figure 1-7.

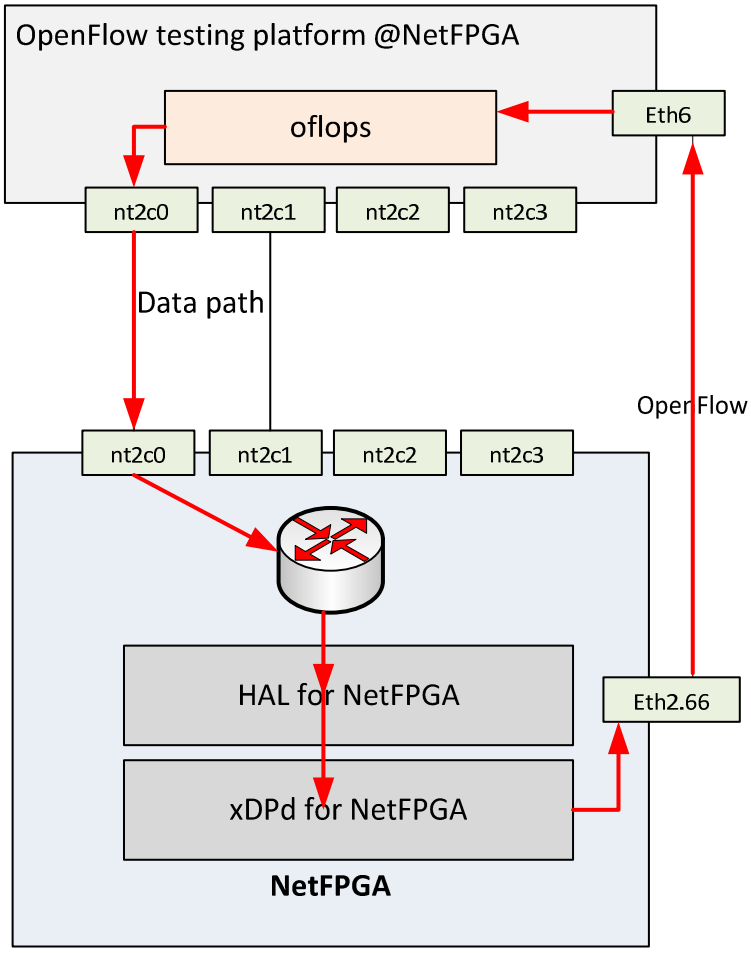


**Figure 1-7 Performance tests of NetFPGA – Packet_In**

*Test condition*

The Packet_In test was done for probe rates: 10, 50, 100, 250, 500, and 1000 Mbps. This parameter controls the data rate of the measurement probe. For all of these rates the following packet size was tested: 100, 300, 500, 700, 900, 1100, 1300, and 1500 B. In order, to approach experiment to the real network environment, we assumed that one per thousand generated data packets triggers of sending packet to the IN_PORT action.

The test duration was set to 60 seconds.

*Test measurements*

During Packet_In test the following parameters were measured:

- Mean and median delay of Packet_In messages together with a standard deviation – measured in µs. This delay includes also the delay of the data plane.

- Throughput – calculated as the number of packet_sent to the number of packet_received by ALIEN OFLOPS.

- Number of packets – number of packets generated by ALIEN OFLOPS during the test (the test duration was 60 seconds).

The interval between UDP packets sent may be calculated in the following way:

data_snd_interval = (pkt_size * 8000000) / (datarate * 1024 * 1024);

The results of the Packet_In test were compared with the OVS implementation.

*Test results*

In Figure 1-8 Performance tests of NetFPGA (Packet_In – 10Mbps) are presented for the following data rates: 10, 100, 250, 500, and 1000 Mbps.

Values for posts (Delay-OVS and Delay-NetFPGA are denoted by the blue and the orange colour posts, respectively) are described by the vertical axis (delay) on the left side of the chart. In turn, values of the throughput and the number of packets are described by the right axis (throughput/# packets) on the right side of the chart.

It should be noted that Packets-OVS and Packets-NetFPGA charts (blue and green line colour on charts, respectively) are overlapped for all range of 10Mbps chart (see Figure 1-8) and partially for 50 Mbps chart (see Figure 1-9). The Throughput-OVS and Throughput-NetFPGA (grey and orange line colour on charts, respectively) are overlapped as well for all range of 10Mbps chart (see Figure 1-8) and partially overlapped for 50Mbps chart (see Figure 1-9).

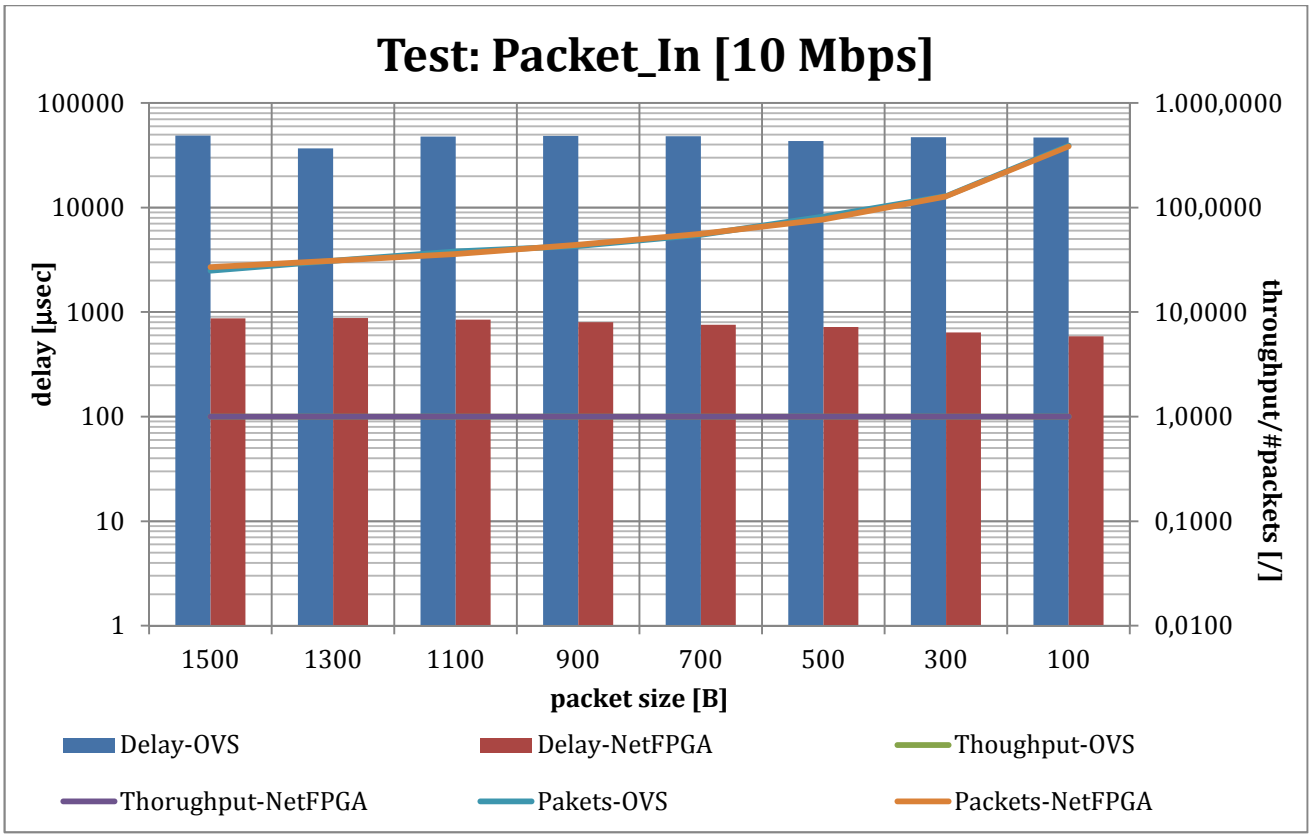


**Figure 1-8 Performance tests of NetFPGA (Packet_In – 10Mbps)**

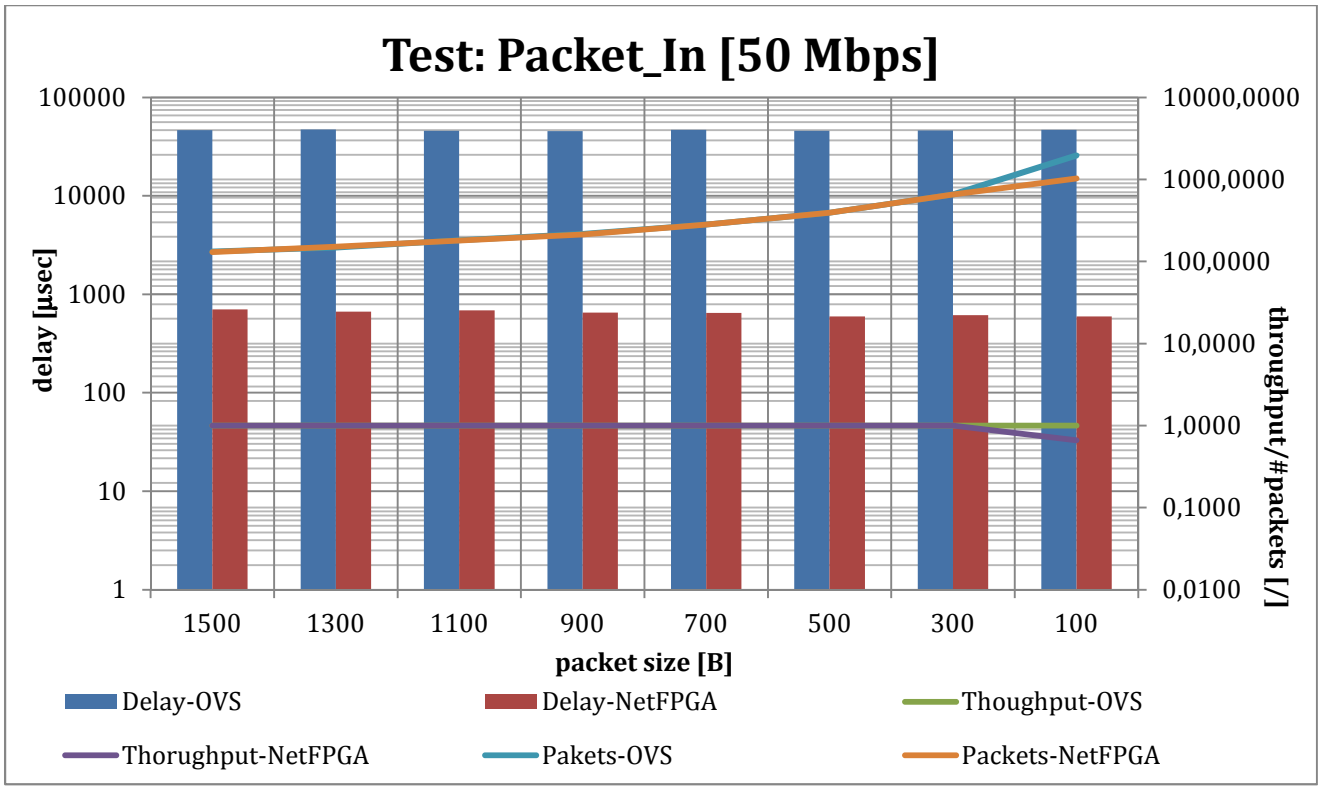


**Figure 1-9 Performance tests of NetFPGA (Packet_In – 50Mbps)**

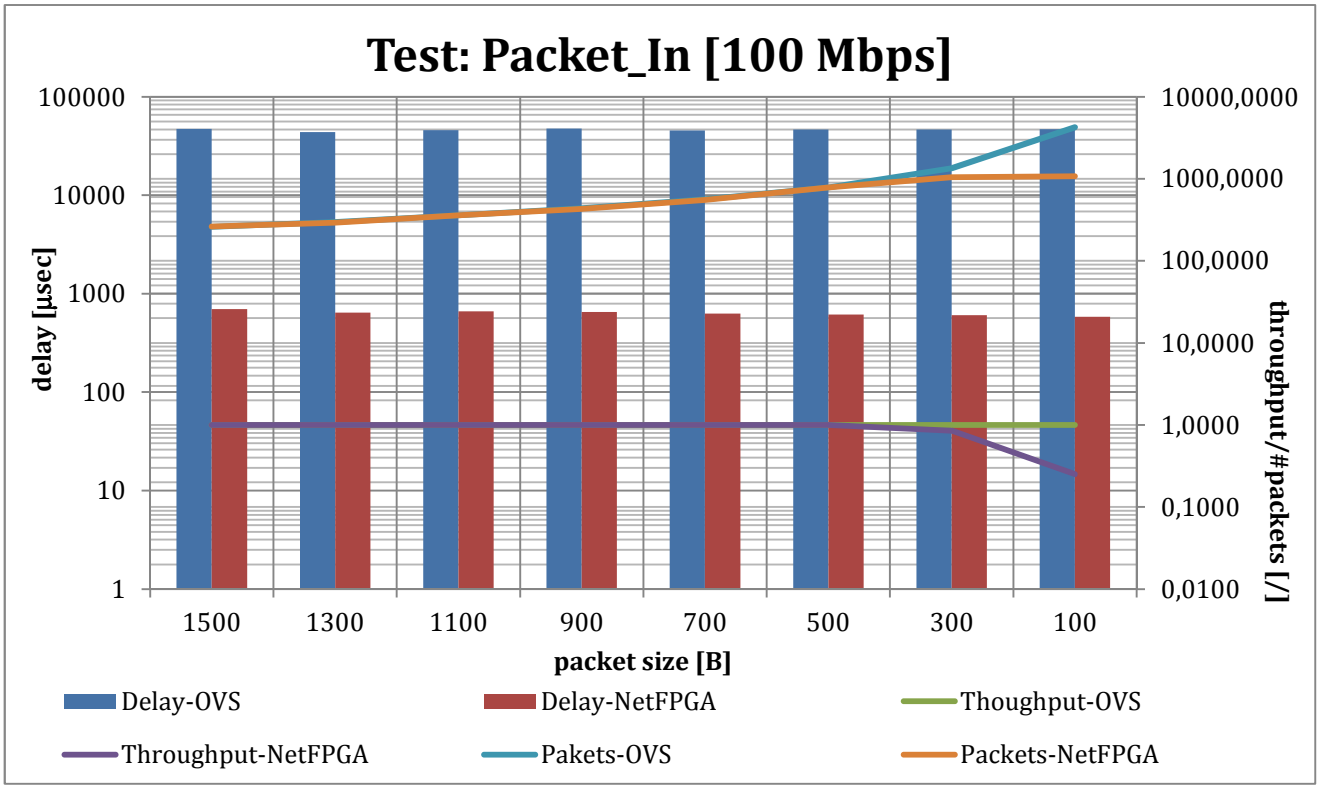


**Figure 1-10 Performance tests of NetFPGA (Packet_In – 100Mbps)**

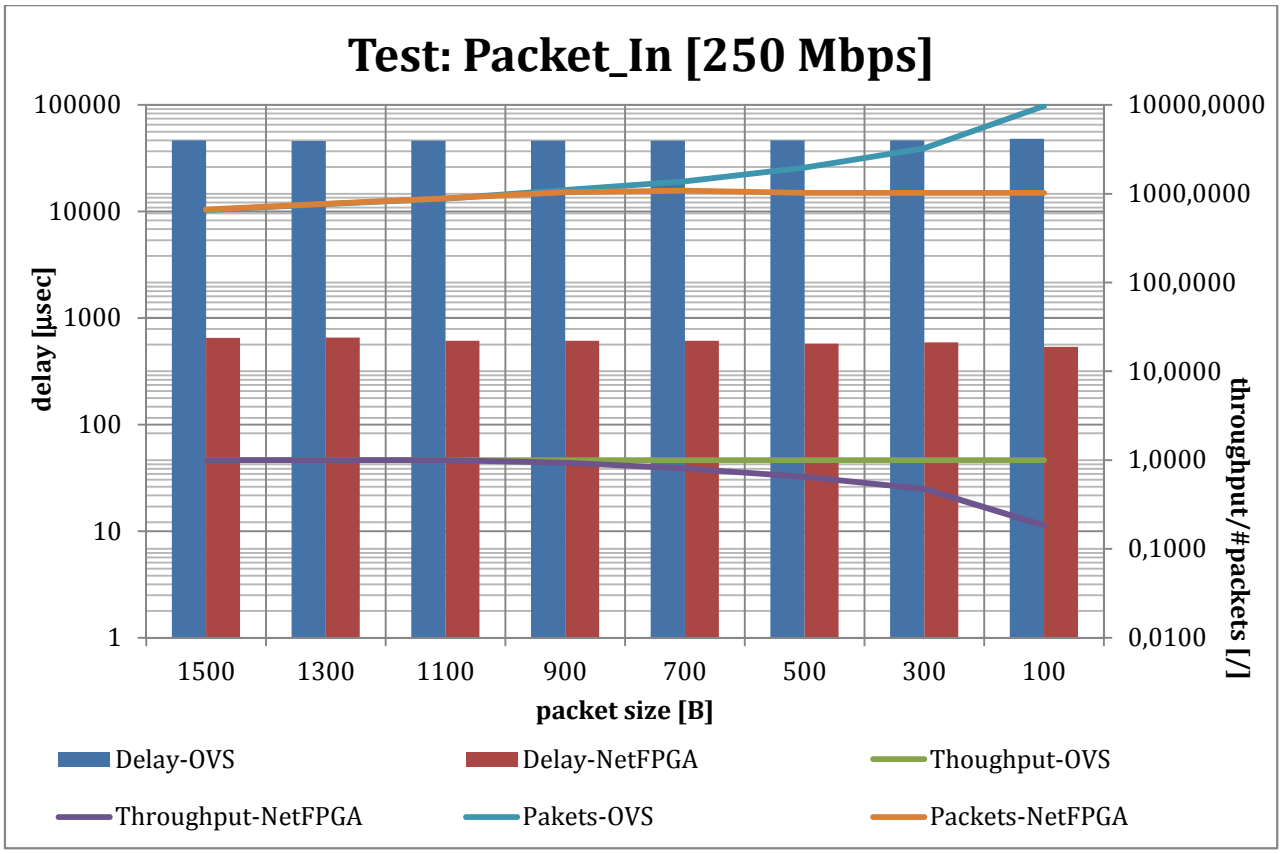


**Figure 1-11 Performance tests of NetFPGA (Packet_In – 250Mbps)**

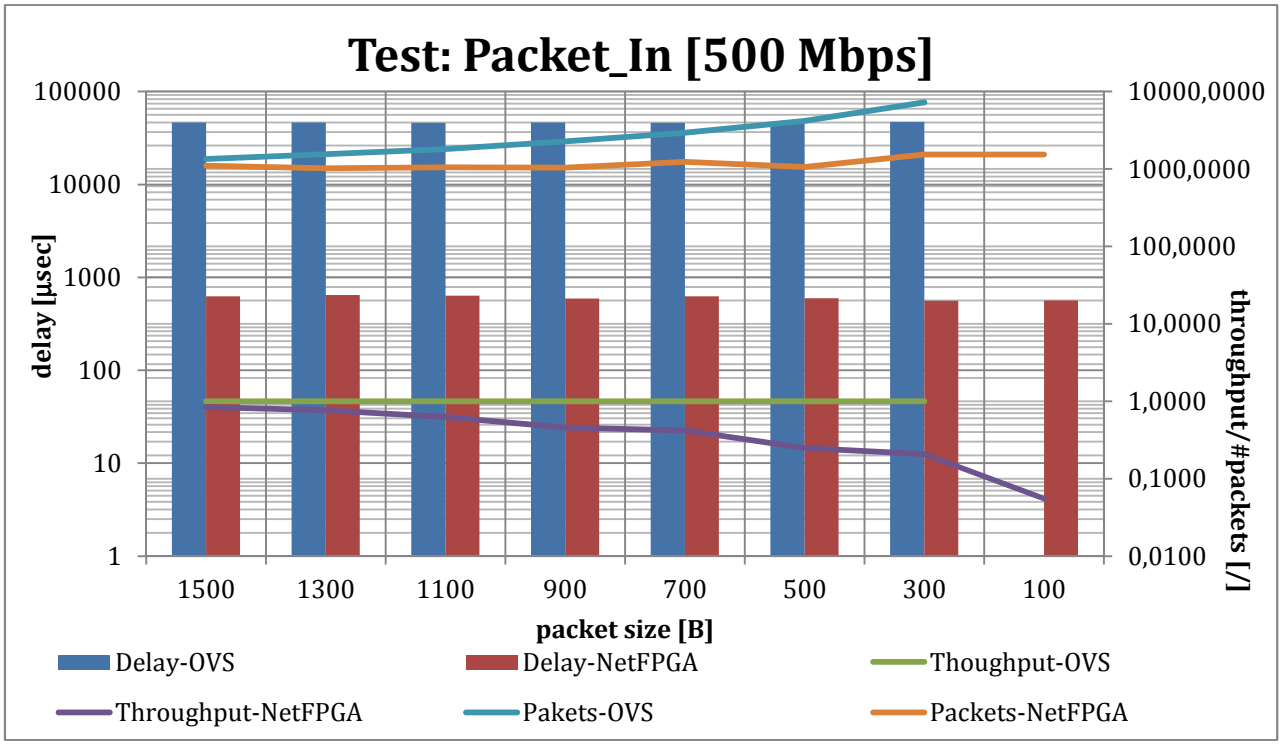


**Figure 1-12 Performance tests of NetFPGA (Packet_In – 500Mbps)**

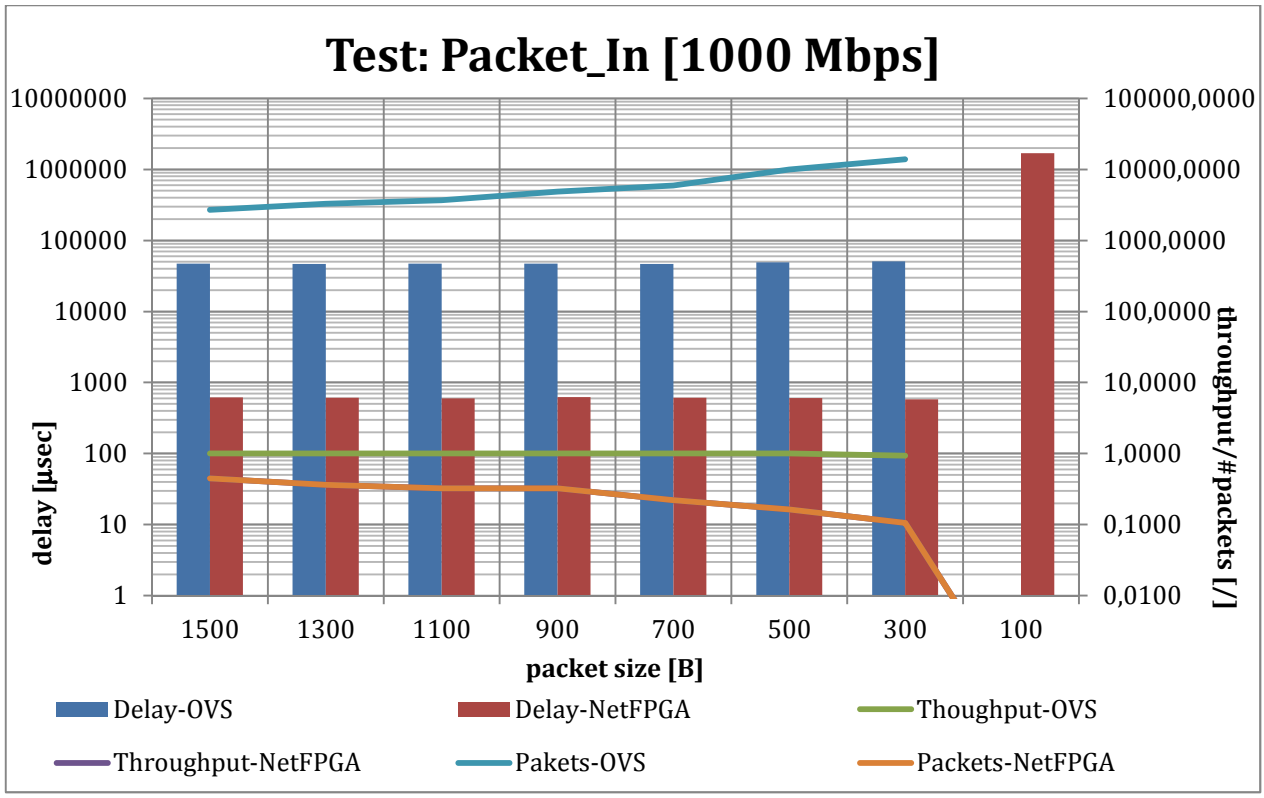


**Figure 1-13 Performance tests of NetFPGA (Packet_In – 1000Mbps)**

| Experiment parameters | | NetFPGA results | | | | |
|---|---|---|---|---|---|---|
| Probe rate (probe_rate) | Packet size (pkt_size) | mean | median | Standard deviation | throughput | packets |
| [Mbps] | [B] | [µsec] | [µsec] | [µsec] | [/] | .[/] |
| 10 | 1500 | 870 | 686 | 946 | 1,0000 | 27 |
| 10 | 1300 | 878 | 635 | 1319 | 1,0000 | 31 |
| 10 | 1100 | 850 | 628 | 1309 | 1,0000 | 36 |
| 10 | 900 | 800 | 615 | 1210 | 1,0000 | 44 |
| 10 | 700 | 759 | 635 | 908 | 1,0000 | 56 |
| 10 | 500 | 720 | 630 | 809 | 1,0000 | 77 |
| 10 | 300 | 637 | 580 | 655 | 1,0000 | 128 |
| 10 | 100 | 589 | 566 | 420 | 1,0000 | 384 |
| 50 | 1500 | 702 | 659 | 495 | 1,0000 | 130 |
| 50 | 1300 | 669 | 620 | 599 | 1,0000 | 151 |
| 50 | 1100 | 687 | 648 | 513 | 1,0000 | 179 |
| 50 | 900 | 652 | 633 | 309 | 1,0000 | 214 |
| 50 | 700 | 648 | 630 | 308 | 1,0000 | 282 |
| 50 | 500 | 596 | 582 | 273 | 1,0000 | 393 |
| 50 | 300 | 616 | 606 | 279 | 1,0000 | 658 |
| 50 | 100 | 595 | 580 | 550 | 0,6613 | 1024 |
| 100 | 1500 | 698 | 678 | 309 | 1,0000 | 263 |
| 100 | 1300 | 643 | 623 | 342 | 1,0000 | 293 |

| | | | | | | |
|---|---|---|---|---|---|---|
| 100 | 1100 | 663 | 649 | 282 | 1,0000 | 358 |
| 100 | 900 | 652 | 635 | 378 | 1,0000 | 429 |
| 100 | 700 | 630 | 622 | 222 | 1,0000 | 551 |
| 100 | 500 | 616 | 610 | 174 | 1,0000 | 785 |
| 100 | 300 | 606 | 600 | 225 | 0,8520 | 1043 |
| 100 | 100 | 581 | 575 | 231 | 0,2518 | 1077 |
| 250 | 1500 | 653 | 643 | 270 | 1,0000 | 663 |
| 250 | 1300 | 657 | 649 | 258 | 1,0000 | 768 |
| 250 | 1100 | 612 | 607 | 148 | 1,0000 | 887 |
| 250 | 900 | 613 | 606 | 199 | 0,9333 | 1036 |
| 250 | 700 | 613 | 611 | 135 | 0,8070 | 1080 |
| 250 | 500 | 575 | 568 | 236 | 0,6491 | 1024 |
| 250 | 300 | 593 | 588 | 213 | 0,4745 | 1024 |
| 250 | 100 | 535 | 536 | 12 | 0,1843 | 1024 |
| 500 | 1500 | 624 | 616 | 283 | 0,8401 | 1099 |
| 500 | 1300 | 646 | 641 | 153 | 0,7634 | 1024 |
| 500 | 1100 | 633 | 629 | 163 | 0,6266 | 1050 |
| 500 | 900 | 592 | 586 | 223 | 0,4555 | 1039 |
| 500 | 700 | 626 | 621 | 139 | 0,4216 | 1241 |
| 500 | 500 | 594 | 566 | 825 | 0,2485 | 1056 |
| 500 | 300 | 558 | 555 | 160 | 0,2077 | 1536 |
| 500 | 100 | 564 | 557 | 164 | 0,0553 | 1536 |
| 1000 | 1500 | 619 | 614 | 191 | 0,4473 | 1131 |
| 1000 | 1300 | 611 | 608 | 145 | 0,3655 | 1088 |
| 1000 | 1100 | 599 | 594 | 161 | 0,3237 | 1085 |
| 1000 | 900 | 623 | 618 | 192 | 0,3260 | 1024 |
| 1000 | 700 | 612 | 607 | 161 | 0,2220 | 1257 |
| 1000 | 500 | 606 | 593 | 408 | 0,1632 | 1529 |
| 1000 | 300 | 582 | 580 | 127 | 0,1056 | 1536 |
| 1000 | 100 | 1681487 | 2919803 | 1307625 | 0,0003 | 418 |

**Table 1-1 Performance tests of NetFPGA – Packet_In results**

We can conclude that the Packet_In message delay is stable (excluding the last experiment) and does not depend on packet size and interface speed. The small variation (about 200 µsec) in measured delay is caused by both, the packet generation delay and the capture delay. Both processes were implemented in software (Linux). We can observe that when packet size and/or interface speed grows, i.e. when the time interval between two generated UDP packets becomes shorter than 25 ms, then packets are lost.

Packet_In messages are generated by the OpenFlow switch only when it does not find a proper entry in the flow table for a frame/packet. Results of this test together with Packet_Out test presented bellow show responsiveness of the system for a new (unknown) flow. We suspect that the instability at the end of investigated range (stream of 1000Mbps of packets with length of 100bytes) is caused by very disadvantageous relation between huge speed and small length of packets, which results with huge amount of packets to be served. Stream of packets with the same length but smaller data rate, contains smaller number of packets. Also, the stream of packets with the same speed (in Mbps) but greater length of packets, contains smaller number of packets, which is easier to be served. In other words, in this case the

number of packets is a problem. In real world it is hard to imagine such a stream of small packets which uses whole throughput of link. It is rather an attack than a traffic, even the artificial traffic.

### 2) Packet_Out

For the Packet_Out test the ALIEN OFLOPS framework generates OFPT_PACKET_OUT messages which are received by the xDPd for NetFPGA OpenFlow endpoint (see Figure 1-14). The UDP packet conveyed by OFPT_PACKET_OUT contains (in data field) information about UDP packet's generation time. The output port for UDP packets, on which packet must be send, are written in one of action field. After reception of UDP packet, Packet_Out delay is calculated (based on the reception time and information from UDP packet payload).

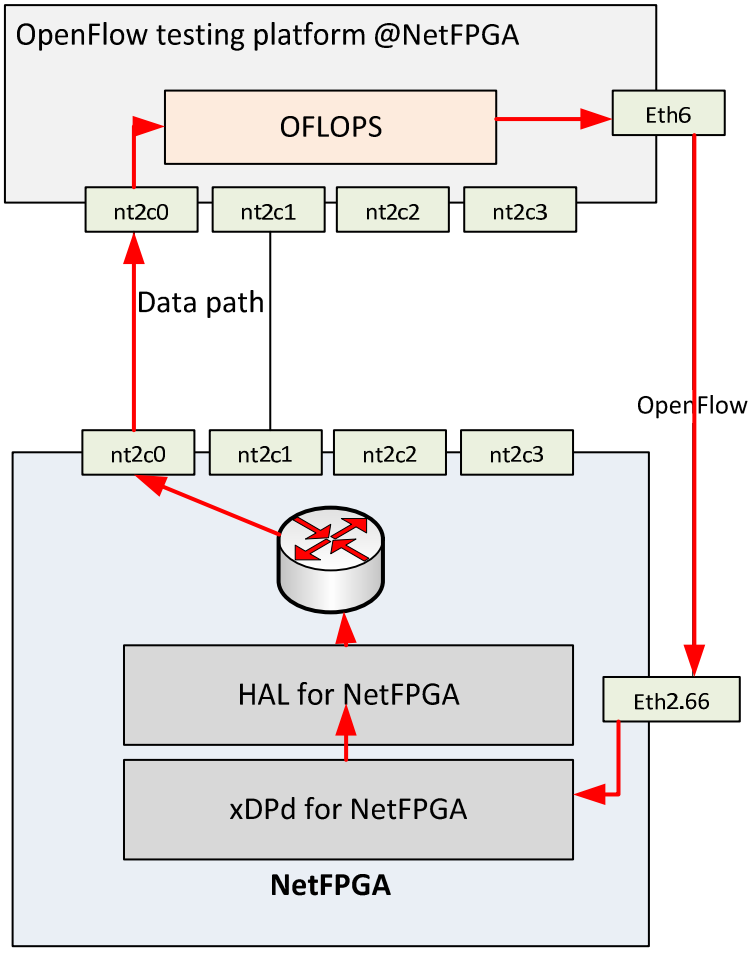


**Figure 1-14 Performance tests of NetFPGA – Packet_Out**

*Test conditions*

The Packet_Out test was done for probe rates: 10, 50, 100, and 250 Mbps. This parameter controls the data rate of the measurement probe. For all of these rates the following packet size was tested: 100, 300, 500, 700, 900, 1100, 1300, and 1500 B.

The test duration: 60 seconds.

*Test measurements*

During Packet_Out test the following parameters were measured:

- Mean and median delay of Packet_Out messages together with a standard deviation – measured in µs
- Throughput – calculated as the number of packet_sent to the number of packet_received by ALIEN OFLOPS.
- Number of packets – number of packets generated by the ALIEN OFLOPS during the test (duration time was 60 seconds).

The interval between UDP packets sent may be calculated in the following way:

data_snd_interval = (pkt_size * 8000000) / (datarate * 1024*1024);

The results of Packet_Out test were compared with results obtained for the OVS implementation (see Table 1-2).

*Test results*

On charts shown in Figure 1-15 - Figure 1-17 results of Packet_Out test are presented for the following data rates: 10, 50, and 100 Mbps. In case of 100 (when packt_size is smaller than 600B), 250, 500, and 1000Gbps the xDPd resets the TCP connection to ALIEN OFLOPS. The results of these tests are not presented here. There are not typical situations and they have to be deeply investigated and improved.

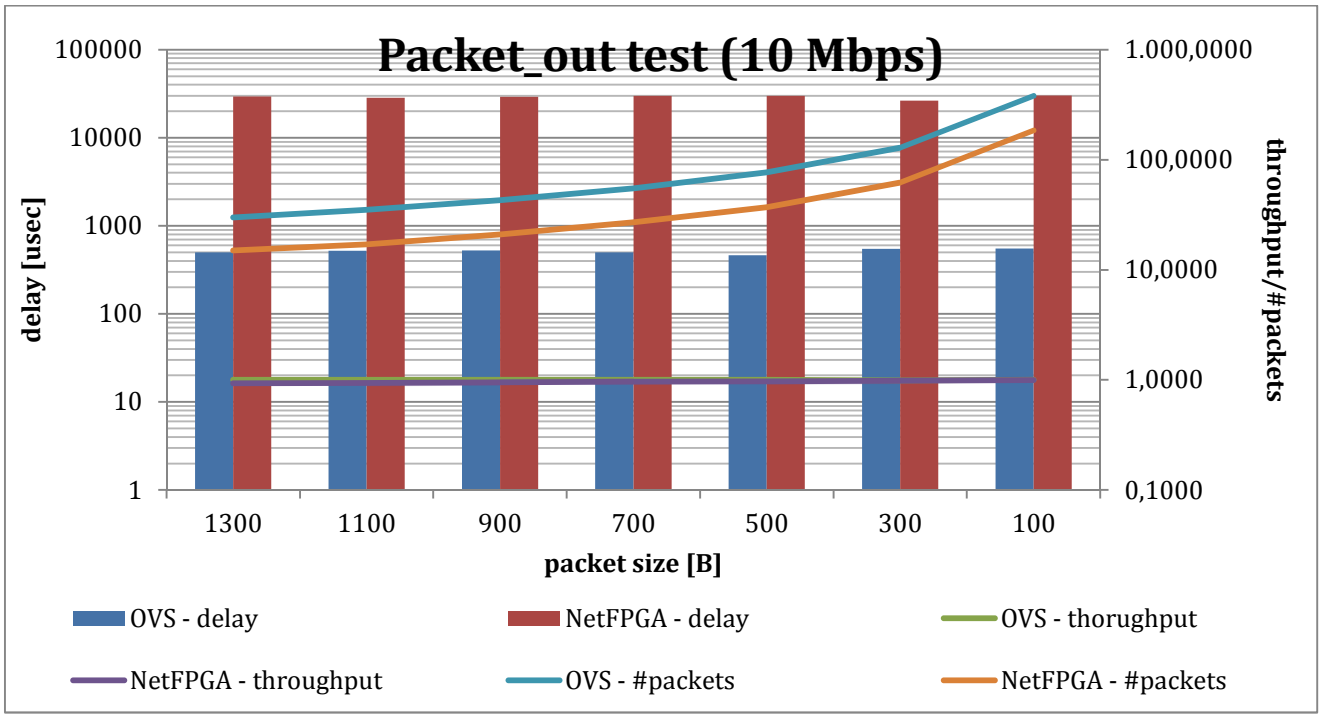


**Figure 1-15 Performance tests of NetFPGA (Packet_Out – 10Mbps)**

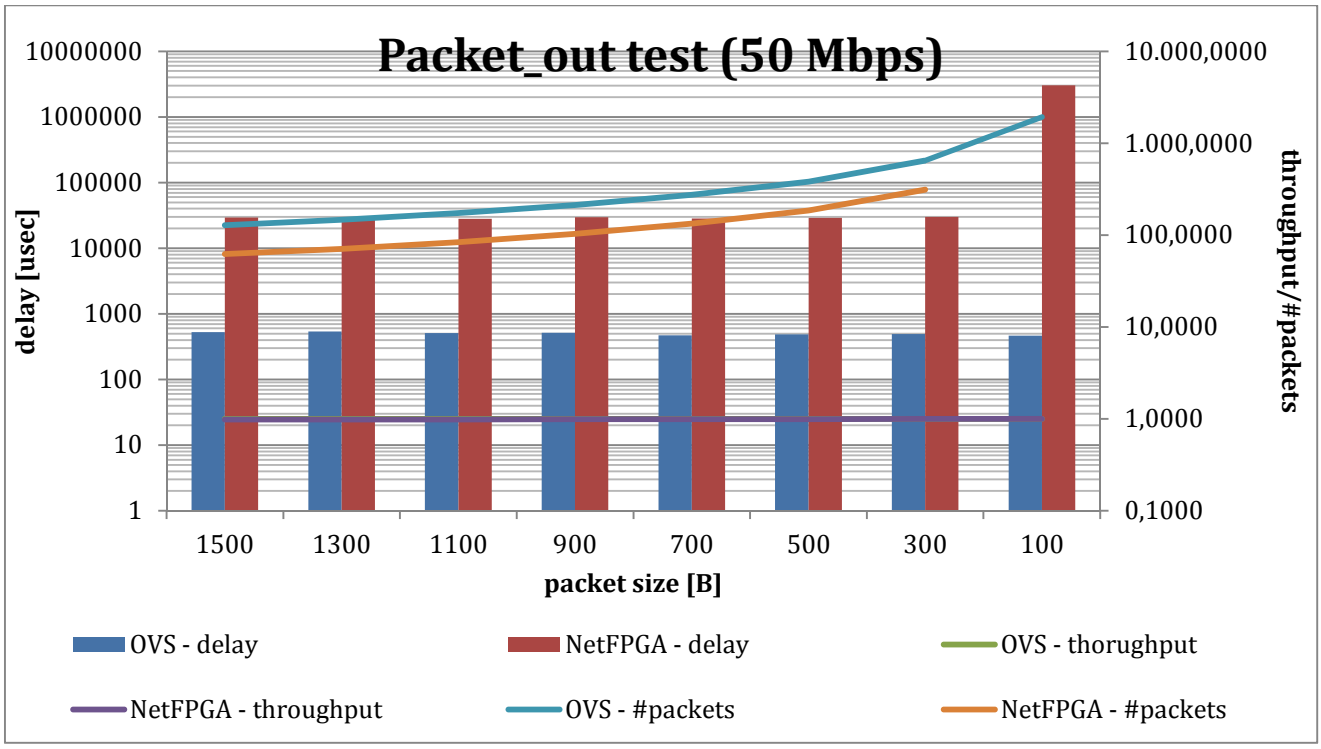


**Figure 1-16 Performance tests of NetFPGA (Packet_Out – 50Mbps)**

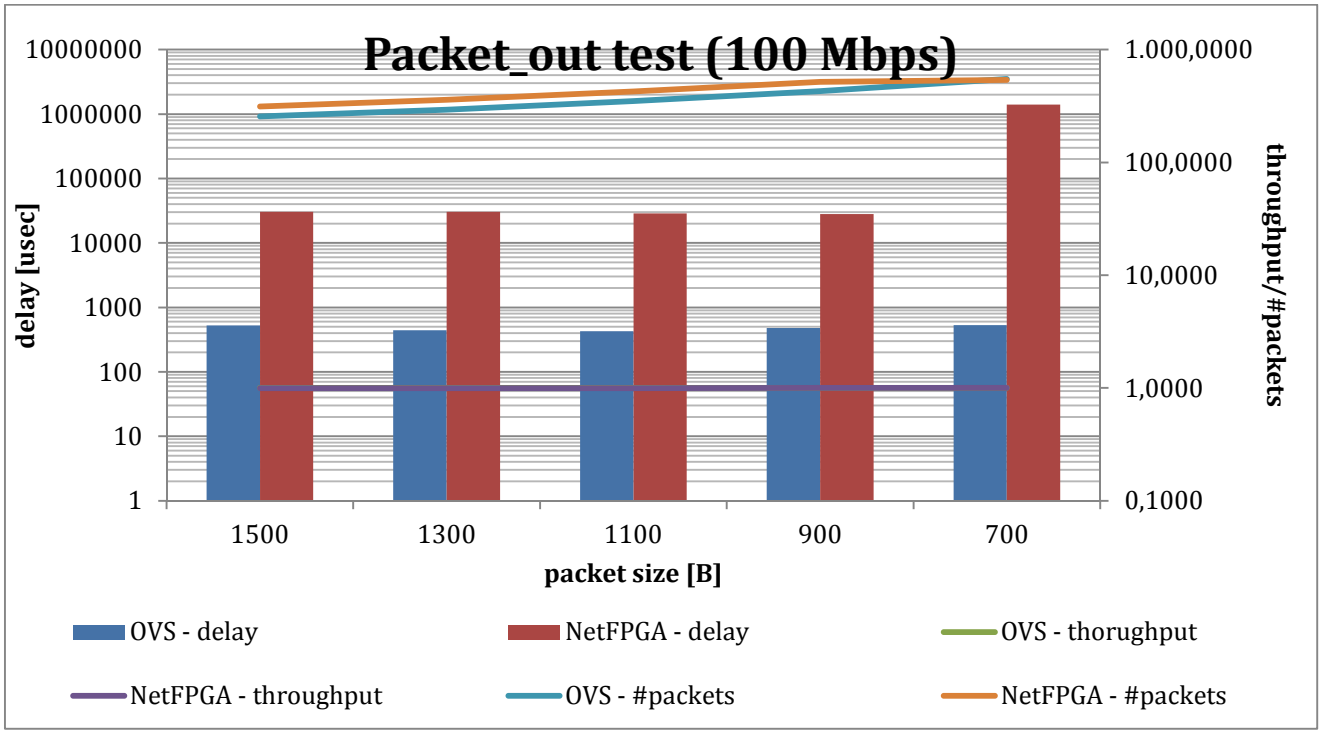


**Figure 1-17 Performance tests of NetFPGA (Packet_Out – 100Mbps)**

| Test parameters | NetFPGA |
|---|---|

| Probe rate (probe_rate) | Packet size (pkt_size) | mean | median | standard deviation | throughput | #packets |
|---|---|---|---|---|---|---|
| [Mbps] | [B] | [µsec] | [µsec] | [µsec] | [/] | [/] |
| 10 | 1500 | 28377 | 28191 | 2705 | 0,9231 | 13 |
| 10 | 1300 | 29347 | 27604 | 4843 | 0,9333 | 15 |
| 10 | 1100 | 28504 | 27313 | 4949 | 0,9412 | 17 |
| 10 | 900 | 29268 | 26953 | 5232 | 0,9524 | 21 |
| 10 | 700 | 29816 | 27517 | 5138 | 0,963 | 27 |
| 10 | 500 | 29911 | 29417 | 5099 | 0,973 | 37 |
| 10 | 300 | 26415 | 25884 | 3639 | 0,9839 | 62 |
| 10 | 100 | 30100 | 32761 | 5140 | 0,9946 | 185 |
| 50 | 1500 | 29500 | 28369 | 3793 | 0,9839 | 62 |
| 50 | 1300 | 29692 | 27671 | 4750 | 0,9859 | 71 |
| 50 | 1100 | 28035 | 26566 | 4296 | 0,9881 | 84 |
| 50 | 900 | 29662 | 27490 | 4973 | 0,9903 | 103 |
| 50 | 700 | 28383 | 27154 | 5806 | 0,9925 | 133 |
| 50 | 500 | 29070 | 29042 | 4072 | 0,9946 | 185 |
| 50 | 300 | 30126 | 27513 | 3763 | 0,9968 | 312 |
| 50 | 100 | 3028420 | 3053878 | 1708773 | 0,9981 | 539 |
| 100 | 1500 | 30499 | 29808 | 5026 | 0,9968 | 312 |
| 100 | 1300 | 30525 | 28880 | 4376 | 0,9972 | 359 |
| 100 | 1100 | 28809 | 27217 | 4976 | 0,9976 | 425 |
| 100 | 900 | 28182 | 28693 | 5784 | 0,9981 | 519 |
| 100 | 700 | 1401174 | 1432268 | 793703 | 0,9981 | 539 |

**Table 1-2 Performance tests of NetFPGA – Packet_Out results**

Results of this test together with Packet_In test show responsiveness of the system for a new (unknown) flows. In case of 250, 500, and 1000Gbps (and partially for 100Mbps) the xDPd resets the TCP connection to ALIEN OFLOPS. This is because the performance of the xDPd for a NetFPGA platform is not optimized. The software part of the system has to analyze and store many packets. Probably here we have the same problem as it was described in results discussion for test of Packet_In messages. This occurrence will be investigated in details in future. At this moment we have some ideas to optimize software by reducing the level of logging and debugging information as well as dedicated options of compilation.

### 3) Data_path_delay

The Data_path_delay tests measure how much time packet requires to be transmitted via the NetFPGA device. Delay between tester and the NetFPGA platform is constant and negligible during the test (Figure 1-18Figure 1-18 ).

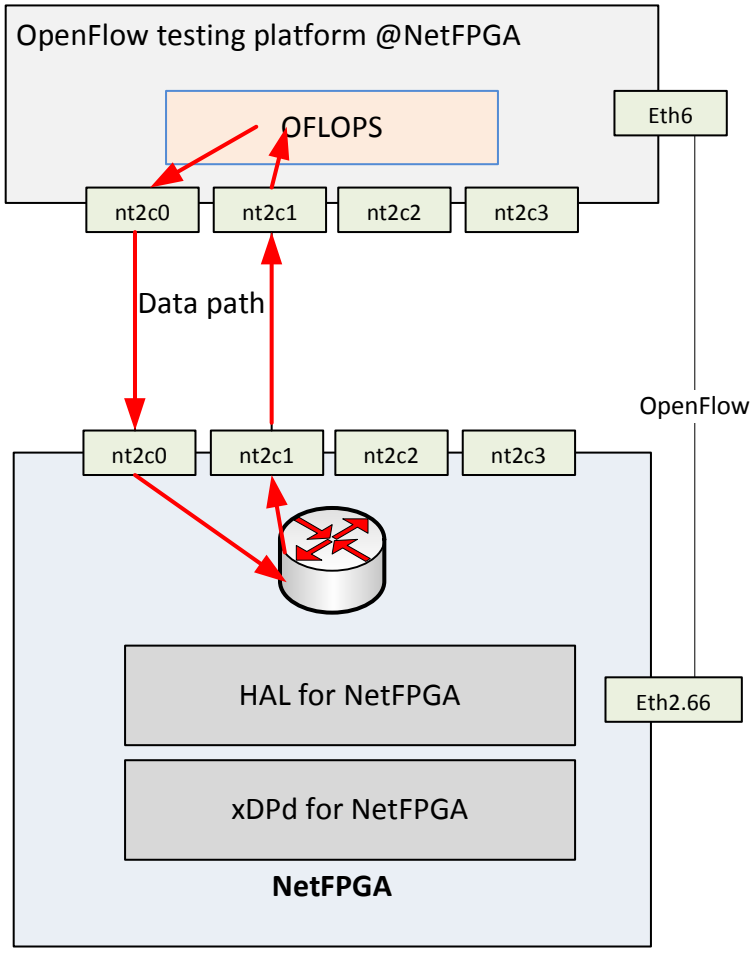


**Figure 1-18 Performance tests of NetFPGA – data_path_delay**

*Test conditions*

The test duration: 30 seconds.

*Test measurements*

During data_path_delay tests the following parameters were measured:

- Mean and median delay of data plane packet switching together with a standard deviation – measured in µs.
- Throughput – calculated as the number of packet_sent to the number of packet_received by ALIEN OFLOPS.
- Number of packets – number of packets generated by the ALIEN OFLOPS during test (duration time was 60 seconds).

The interval between UDP packets may be calculated in the following way:

data_snd_interval = (pkt_size * 8000000) / (datarate * 1024*1024);

The results of data_path_delay test were compared with OVS implementation.

*Test results*

On the three charts showed in Figure 1-19- Figure 1-21 results of data_path_delay test are presented for the following data rates: 10, 100, and 1000Mbps.

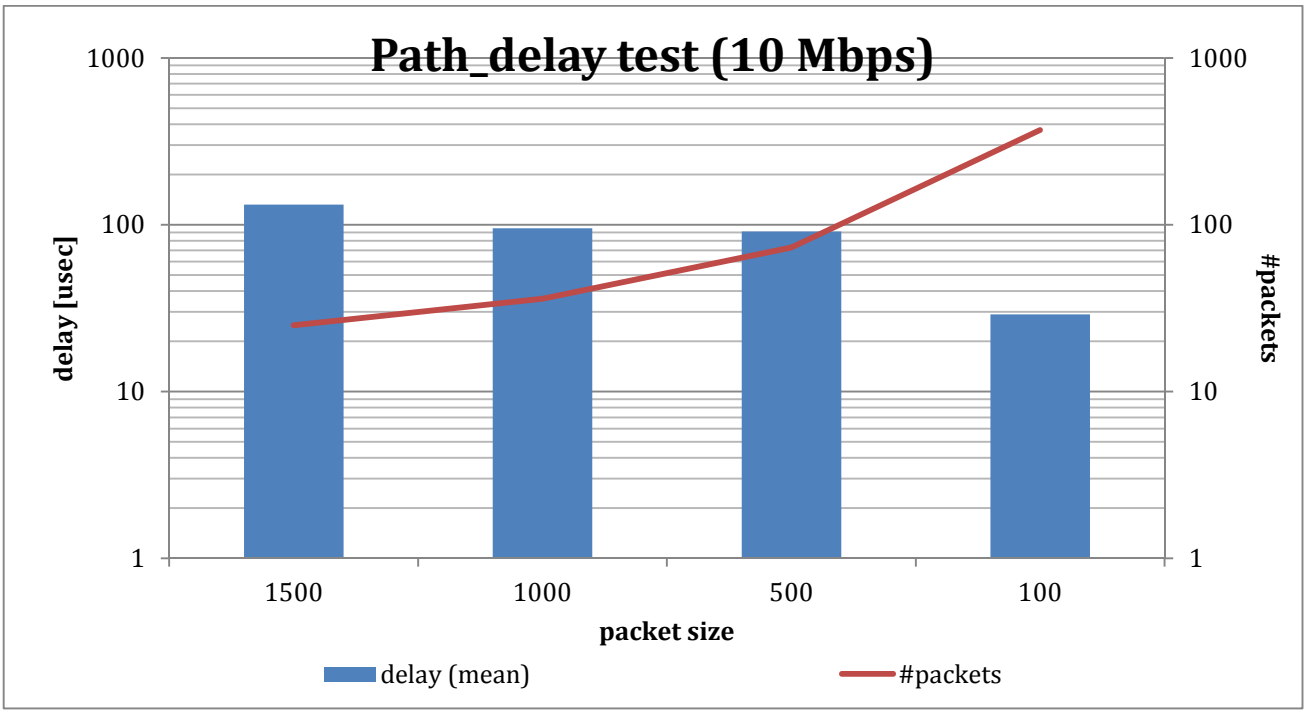


**Figure 1-19 Performance tests of NetFPGA (data_path_delay – 10Mbps)**

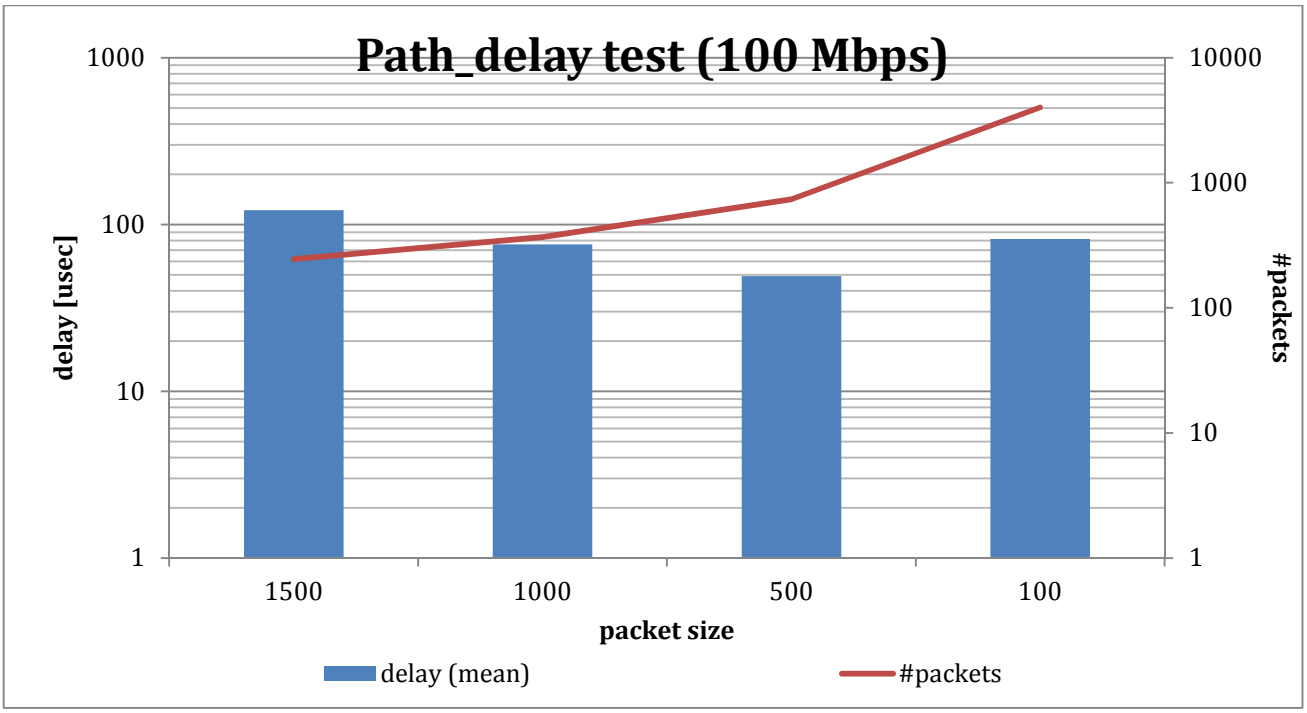


Figure 1-20 Performance tests of NetFPGA (data_path_delay – 100Mbps)

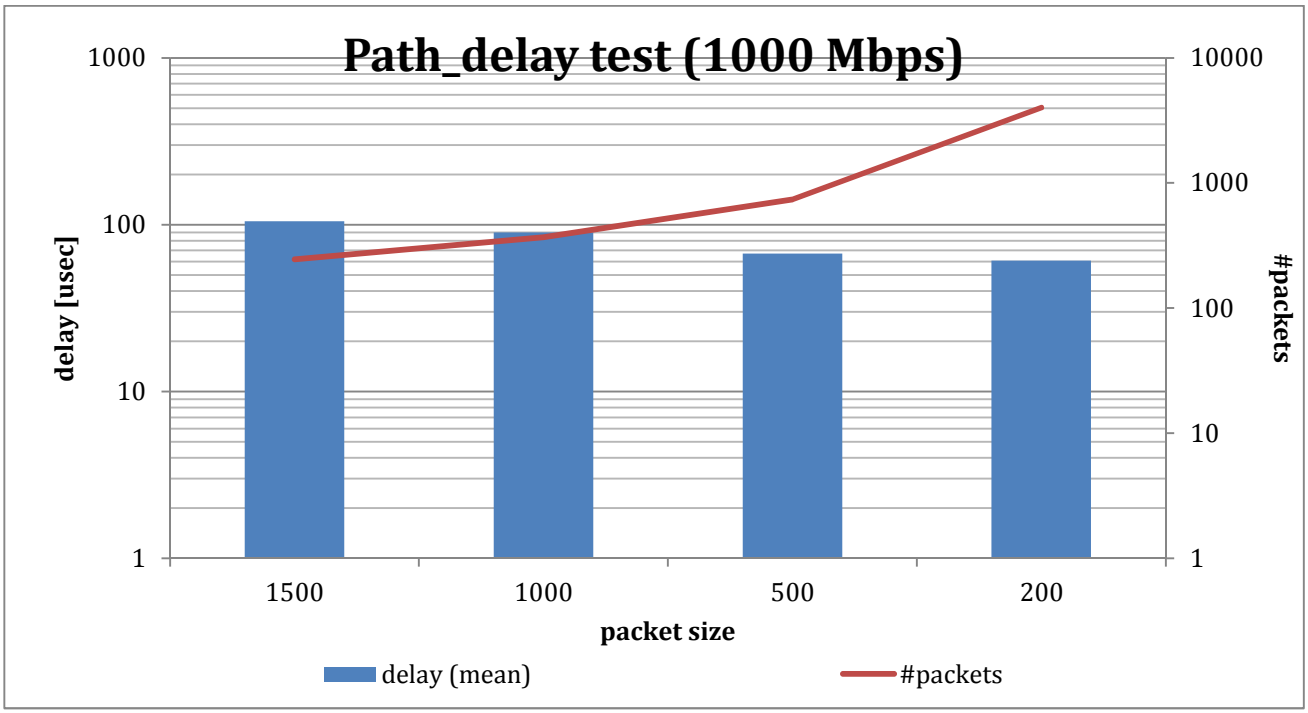


Figure 1-21 Performance tests of NetFPGA (data_path_delay – 1000Mbps)

| Probe rate (probe_rate) | Packet size (pkt_size) | mean | median | standard deviation | throughput | #packet send |
|---|---|---|---|---|---|---|
| [Mbps] | [B] | [µs] | [µs] | [µs] | [/] | [/] |
| 10 | 1500 | 132 | 115 | 31 | 1 | 25 |

| | | | | | | |
|---|---|---|---|---|---|---|
| 10 | 1100 | 95 | 97 | 6 | 1 | 36 |
| 10 | 500 | 91 | 94 | 8 | 1 | 73 |
| 10 | 100 | 29 | 30 | 0 | 1 | 369 |
| 100 | 1500 | 122 | 107 | 18 | 1 | 246 |
| 100 | 1000 | 76 | 77 | 0 | 1 | 368 |
| 100 | 500 | 49 | 49 | 1 | 1 | 737 |
| 100 | 100 | 82 | 88 | 13 | 1 | 4000 |
| 1000 | 1500 | 105 | 105 | 1 | 1 | 2546 |
| 1000 | 1000 | 90 | 95 | 8 | 1 | 3831 |
| 1000 | 500 | 67 | 58 | 16 | 1 | 9334 |
| 1000 | 200 | 61 | 59 | 26 | 1 | 28000 |

**Table 1-3 Performance tests of NetFPGA – Data path delay results**

The data_path_delay tests show that data plane packets switching time for NetFPGA device depends on frame length. The shortest measured frames (100 bytes) are switched in ~30 µs whereas the longest (1500 bytes) are processed in ~130 µs. Since in the flow table was installed only one flow (matches exact the generated traffic), so searching time is very low, presented values of delay represent only packet forwarding time. It shows the time of serving packets in the optimal (laboratory) conditions. More tests in more realistic cases can be also performed. Some details and assumptions for them are presented in Section XXX concluding this chapter.

### 4) The control_channel_delay test

In the control_channel_delay test the ALIEN OFLOPS tool sends OpenFlow Echo-requests to the NetFPGA device and measures delay receiving the OpenFlow Echo-reply messages (see Figure 1-22).

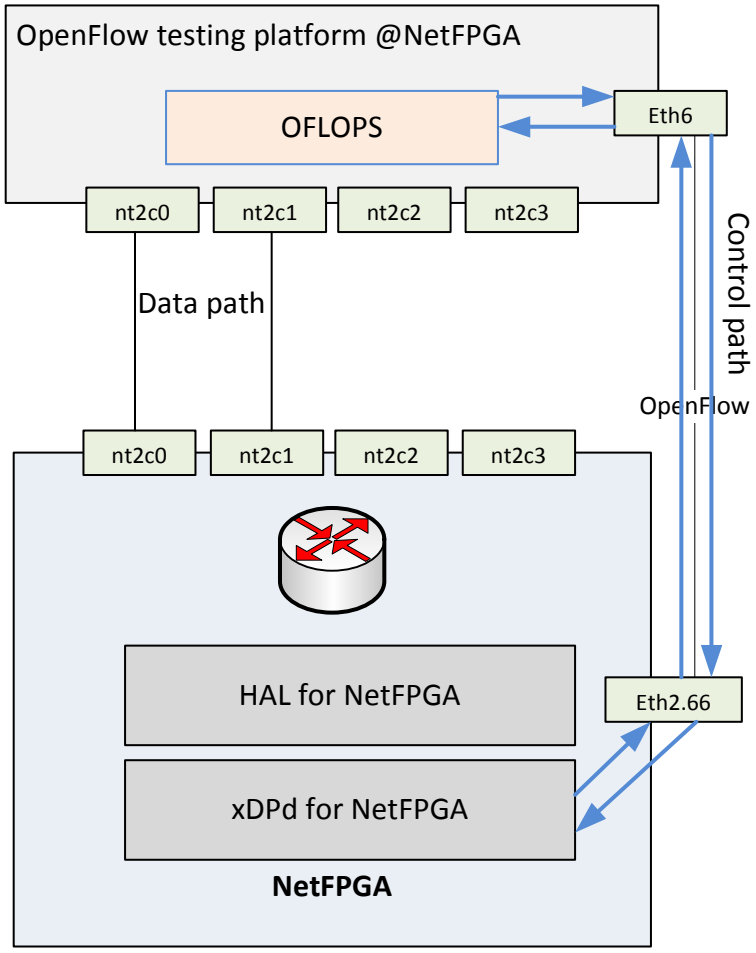


**Figure 1-22 Performance tests of NetFPGA – control_channel_delay**

*Test conditions*

The ALIEN OFLOPS was launched with parameter echo_rate=3.

*Test measurements*

During control_channel_delay tests the following parameters were measured:

- Mean and median delay of OpenFlow Echo-replay message with a standard deviation – measured in µs.
- Throughput – calculated as the number of packet_sent to the number of packet_received by ALIEN OFLOPS.
- Number of packets – number of packets generated by the ALIEN OFLOPS during the test (duration time 60 seconds).

The interval between UDP packets may be calculated in the following way:

data_snd_interval = (pkt_size * 8000000) / (datarate * 1024*1024);

The results of control_channel_delay test were compared with OVS implementation.

*Test results*

| | mean | median | standard deviation | throughput | #packet |
|---|---|---|---|---|---|
| Control channel delay | [µs] | [µs] | [µs] | [/] | [/] |
| | OpenVSwitch | | | | |
| | 354 | 341 | 6323 | 1 | 26 |
| | NetFPGA | | | | |
| | 1277 | 1365 | 217 | 1 | 25 |

**Table 1-4 Performance tests of NetFPGA – control_channel_delay results**

In our test, the measured control_channel_delay is equal to 1.28 ms which is the time required to the xDPd and the ROFL software to generate an OpenFlow replay. Presented results are worse than for the OVS. It is caused by longer and not optimized software processing chain as well as by using two different machines (in OVS all tests have been realized on virtual machines, all inter-machine communications was exactly inter-process communication in the same physical PC). However, in this case the scenario is very close to real ones - controllers are realized by different machines than switches.

### 5) The Add_flow test

The Add_flow test is prepared in order to determine the time required to install the given number of flow table entries (Figure 1-23). In this test, the ALIEN OFLOPS framework generates UDP packets with constant bit rate and sends out them to the configured data plane port of the NetFPGA card. After 10 seconds the ALIEN OFLOPS starts sending a set of OpenFlow flowadd commands to the NetFPGA platform with the new flow entries. All commands are encapsulated in one (or more if needed) TCP segments. Only the last flow entry matches generated UDP packets. When the first TCP segment is generated, time counter starts. This counter is stopped when the first packet is received on a data port, configured as OUT_PORT in the last installed flow. Experiment is repeated. Values of time counter are used in order to calculate mean, median, and standard deviation of time consumed by the flow install action.

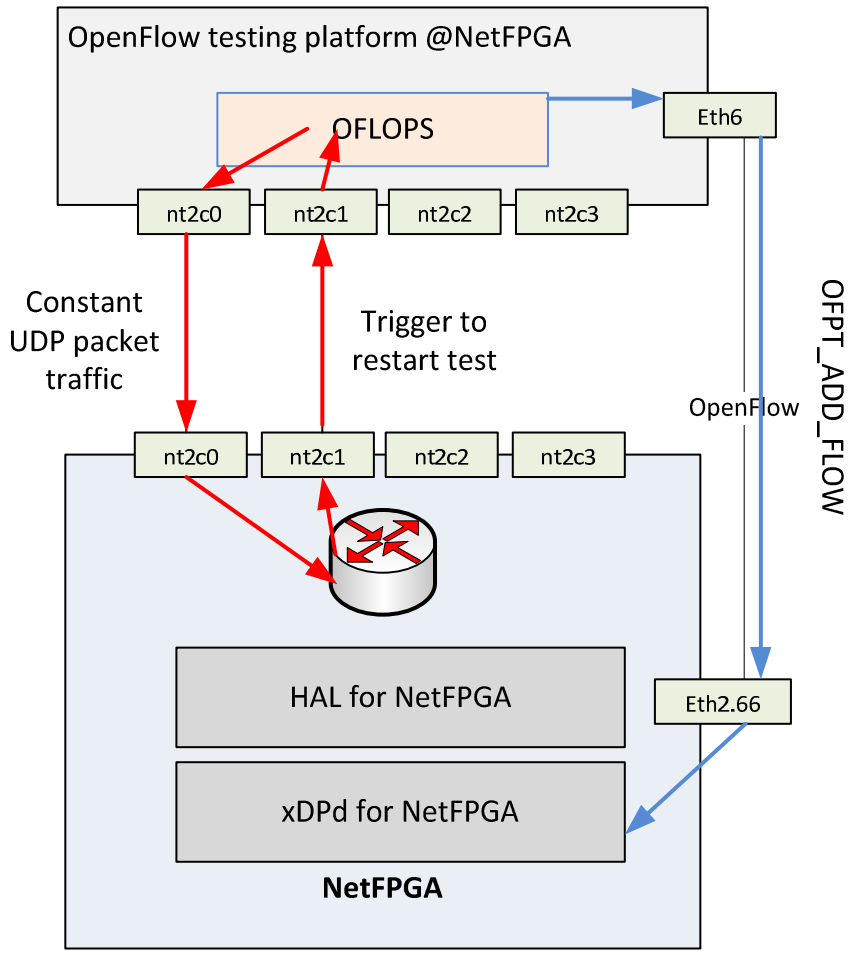


**Figure 1-23 Performance tests of NetFPGA – Add_flow delay**

*Test conditions*

The add_flow test was done for number of flows installed on the NetFPGA card: 1, 10, 20, 30, 40, 50, 60, 70, 80, 90, and 100.

The additional test was done to measure a single flow installation time for the number of flows: 1-1000 in flow table stored in the NetFPGA card.

The test duration was set to 120 seconds.

*Test measurements*

During add_flow test the following parameters were measured:

- Mean and median delay of the installation of a given number of flow entries in the NetFPGA cards.
- Delay per single flow – calculated from mean delay and the number of installed flows.

The interval between UDP packets may be calculated in the following way:

data_snd_interval = (pkt_size * 8000000) / (datarate * 1024*1024);

The results of add_flow test were compared with results obtained for the OVS implementation.

*Test results*

Results of add_flow test are presented on a single chart (see Figure 1-24).

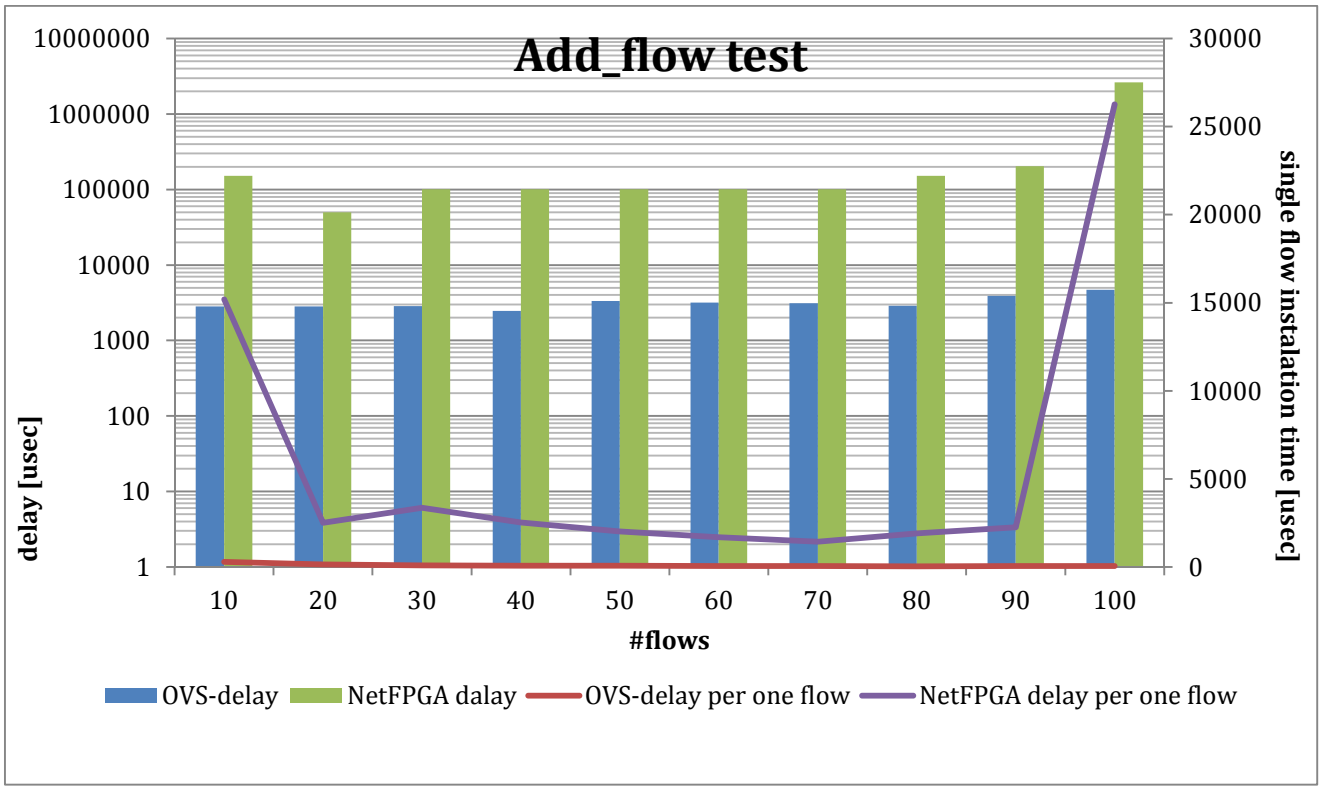


**Figure 1-24 Performance tests of NetFPGA (add_flow)**

| Flows [/] | mean [µsec] | median [µsec] | standard deviation [µsec] |
|---|---|---|---|
| 1 | 14922 | 14922 | 1 |
| 10 | 151938 | 151935 | 15 |
| 20 | 49970 | 49971 | 1 |
| 30 | 100910 | 100920 | 20 |
| 40 | 100902 | 100917 | 23 |
| 50 | 100914 | 100916 | 8 |
| 60 | 100925 | 100935 | 24 |
| 70 | 100906 | 100916 | 26 |
| 80 | 151908 | 151889 | 30 |

| | | | |
|---|---|---|---|
| 90 | 202791 | 202919 | 433 |
| 100 | 2625542 | 2625259 | 1554718 |

**Table 1-5 Performance tests of NetFPGA - add_flow results**

We can observe that flow installation in the NetFPGA card is slower (more than two orders of magnitude) in comparison to the OVS software switch. Such difference is caused by the fact that HAL implementation adds additional processing stage.

On the chart showed in Figure 1-25it is presented a single installation time into the NetFPGA flow table. The time consumed by flow installation process depends strongly on the number of installed flows; however, in wide range the flow number is constant. Only for a higher number of installed flows, required time grows. The irregularity is caused by a different length of packets used for testing particular number of flows. It was necessary to change some parameters to create usable tests (acceptable by switches)

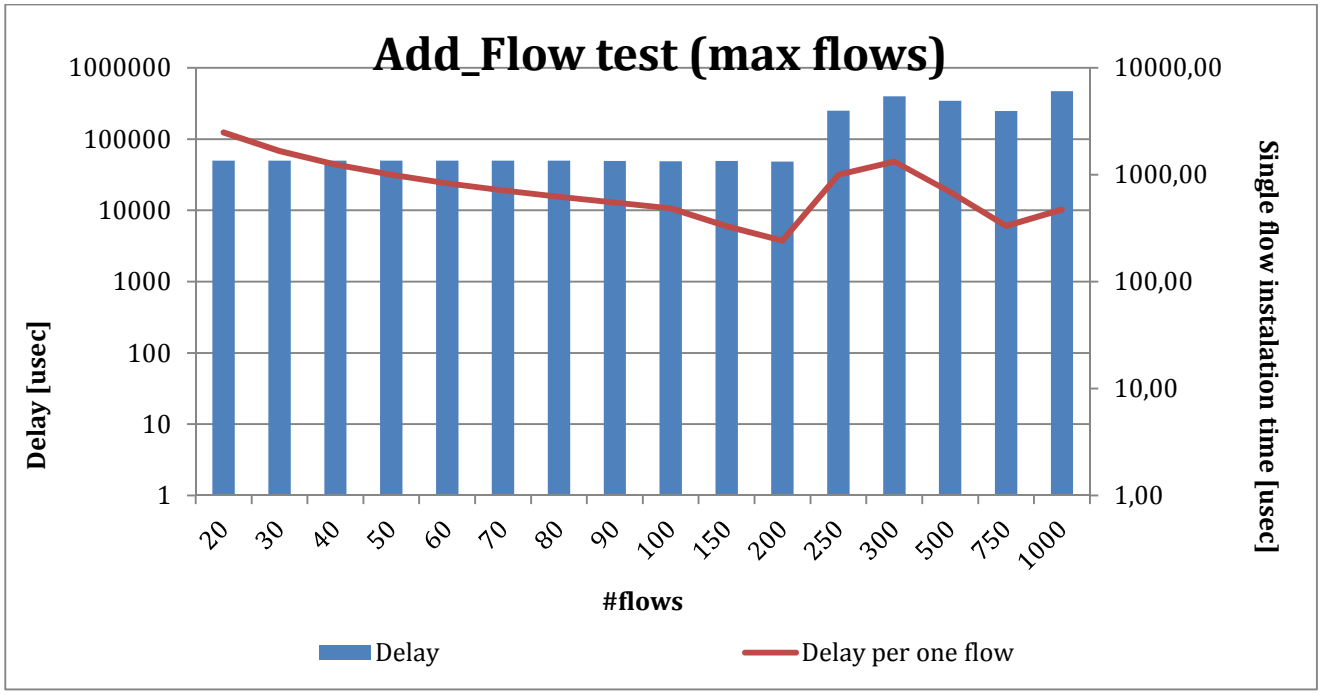


**Figure 1-25 Performance tests of NetFPGA (add_flow – max number)**

| Add flow - max number | Delay [usec] | Single flow installation time [usec] |
|---|---|---|
| 20 | 49899 | 2494,95 |
| 30 | 49996 | 1666,53 |
| 40 | 49933 | 1248,33 |
| 50 | 49927 | 998,54 |
| 60 | 49898 | 831,63 |
| 70 | 49972 | 713,89 |
| 80 | 49906 | 623,83 |
| 90 | 49615 | 551,28 |
| 100 | 48662 | 486,62 |

| | | |
|---|---|---|
| 150 | 49554 | 330,36 |
| 200 | 48489 | 242,45 |
| 250 | 249451 | 997,80 |
| 300 | 399138 | 1330,46 |
| 500 | 345702 | 691,40 |
| 750 | 248890 | 331,85 |
| 1000 | 472663 | 472,66 |

**Table 1-6 Performance tests of NetFPGA (add_flow – max number)**

### 1.2.2 EZappliance

Performance tests of HAL-enabled EZappliance platform was done using ALIEN OFLOPS framework described above. To test performance of OpenFlow-capable hardware based on EZchip NP-3 network processor two data plane ports (eth1 and eth2) of EZappliance was connected to OpenFlow testing platform (ports nt2c0 and nt2c1) based on NetFPGA 1G card with ALIEN OFLOPS installed. Communication channel between EZappliance and testing platform was established using OpenFlow v1.0 protocol as depicted on Figure 1-26.

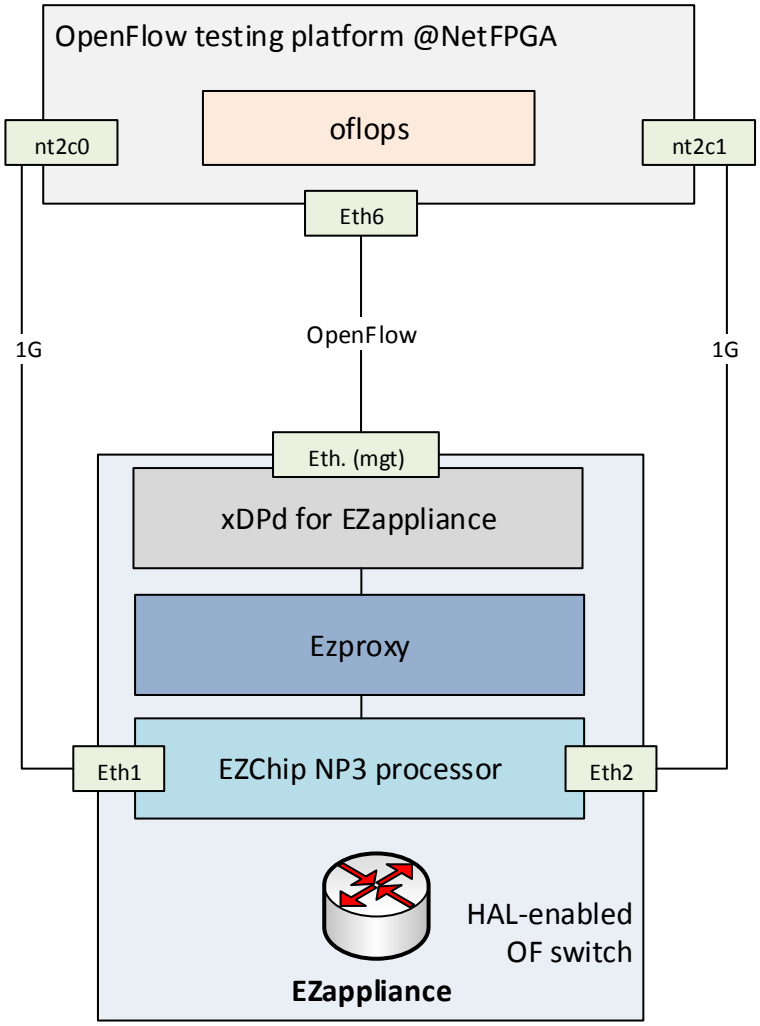


**Figure 1-26 Performance tests of EZappliance - setup topology**

xDPd was configured to work with ALIEN OFLOPS as OpenFlow controller (xDPd.conf):

```
Example of configuration single LSI

config:{

   openflow:{
     logical-switches:{
             #Name of the switch dp0
             dp0:{
                     #Most complex configuration
                     dpid = "0x11"; #Must be hexadecimal
                     version = 1.0;
                     description="This is a switch";

                     #Controller
```

```
                    mode="active"; #active, passive, (TODO: both)
                    master-controller-ip="10.216.66.190";
                                #master-controller-ip="localhost";
                    master-controller-port=6633;
                    reconnect-time=1; #seconds

                    #Tables and MA
                    num-of-tables=1;

                    #Physical ports attached to this logical switch. This is
   mandatory
                    #The order and position in the array dictates the number of
                    # 1 -> eth1, 2 -> eth2, 3 -> eth3
                    ports        =        ("eth1",        "eth2",        "eth3",
   "eth4","eth5","eth6","eth7","eth8","eth9","eth10","eth11","eth12","eth13","e
   th14","eth15","eth16");

              };
     };
   };
};
```

The following tests were done for EZappliance platform:

- Packet_in
- Packet_out
- Data_path_delay
- Control_channel_delay
- Add_flow

For all of these tests some statistics was captured. All abovementioned scenarios, test conditions, and results with charts are described in the following sub-chapters.

Test performance results for EZappliance platform was compared with OVS implementation and results are presented on charts and in the tables.

### 1) Packet_in

For Packet_in test ALIEN OFLOPS framework generates UDP packets that don't match any flow in the flow table into NPU. All unmatched packets are forwarded to queue dedicated for EZproxy. This module handles packets via Interrupt System Routines and then through TCP socket sent them to xDPd for EZappliance. xDPd configured to work with ALIEN OFLOPS as OpenFlow controller generates packet_in message and send it to ALIEN OFLOPS (Figure 1-27).

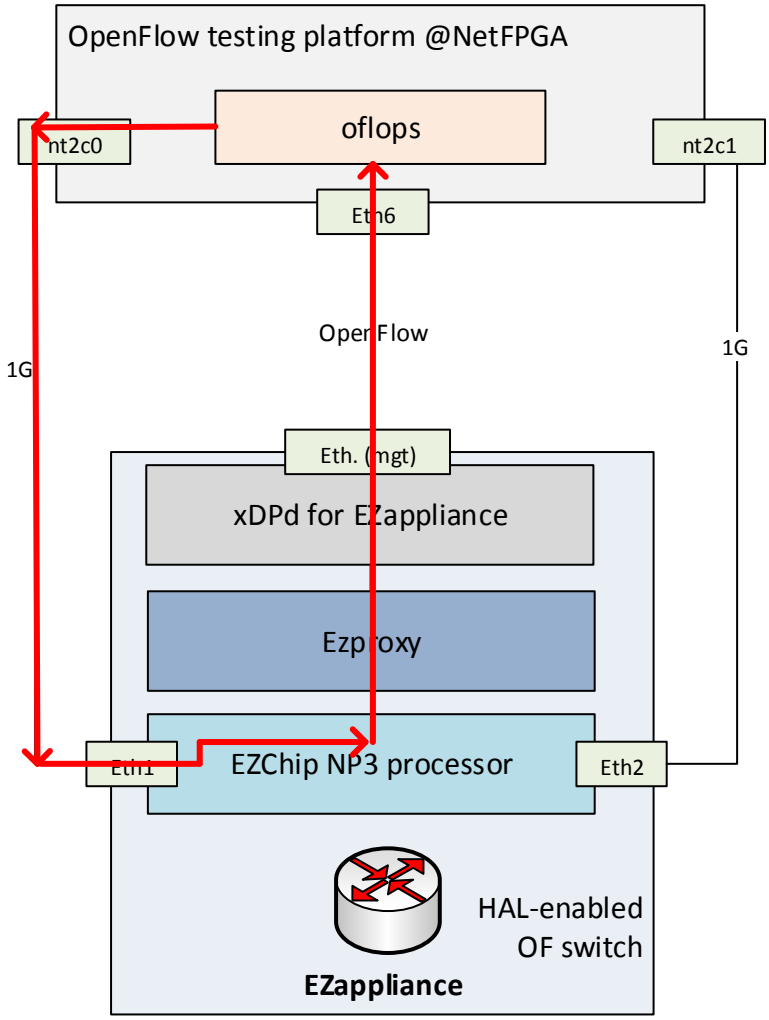


**Figure 1-27 Performance tests of EZappliance – packet_in**

*Test condition*

Packet_in test was done for probe rates: 10, 50, 100, 250, 500, 1000Mbps. This parameter controls the data rate of the measurement probe. For all of these rates the following packet size was tested: 100, 300, 500, 700, 900, 1100, 1300 and 1500B.

The test duration: 60 seconds.

*Test measurements*

During packet_in test the following parameters was measured:

- Mean and median delay of packet_in messages together with standard deviation – measured in µs. This delay includes also delay of data path (path between nt2c0 of NetFPGA and eth1 of EZappliance) but it is constant and negligible during the test.

- Throughput – calculated as number of packet_sent to number of packet_received by ALIEN OFLOPS. Throughput factor equal 1 means: “all UDP transmitted by ALIEN OFLOPS packets was then received”.

- Number of packets – number of packets generated by ALIEN OFLOPS during test (60 seconds).

Interval between UDP packet sent may be calculated in the following way:

data_snd_interval = (pkt_size * byte_to_bits * sec_to_µs) / (datarate * mbits_to_bits);

The results of packet_in test were compared with OVS implementation.

*Test results*

On the next six charts results of packet_in test are presented for the following data rates: 10, 100, 250, 500, 1000Mbps.

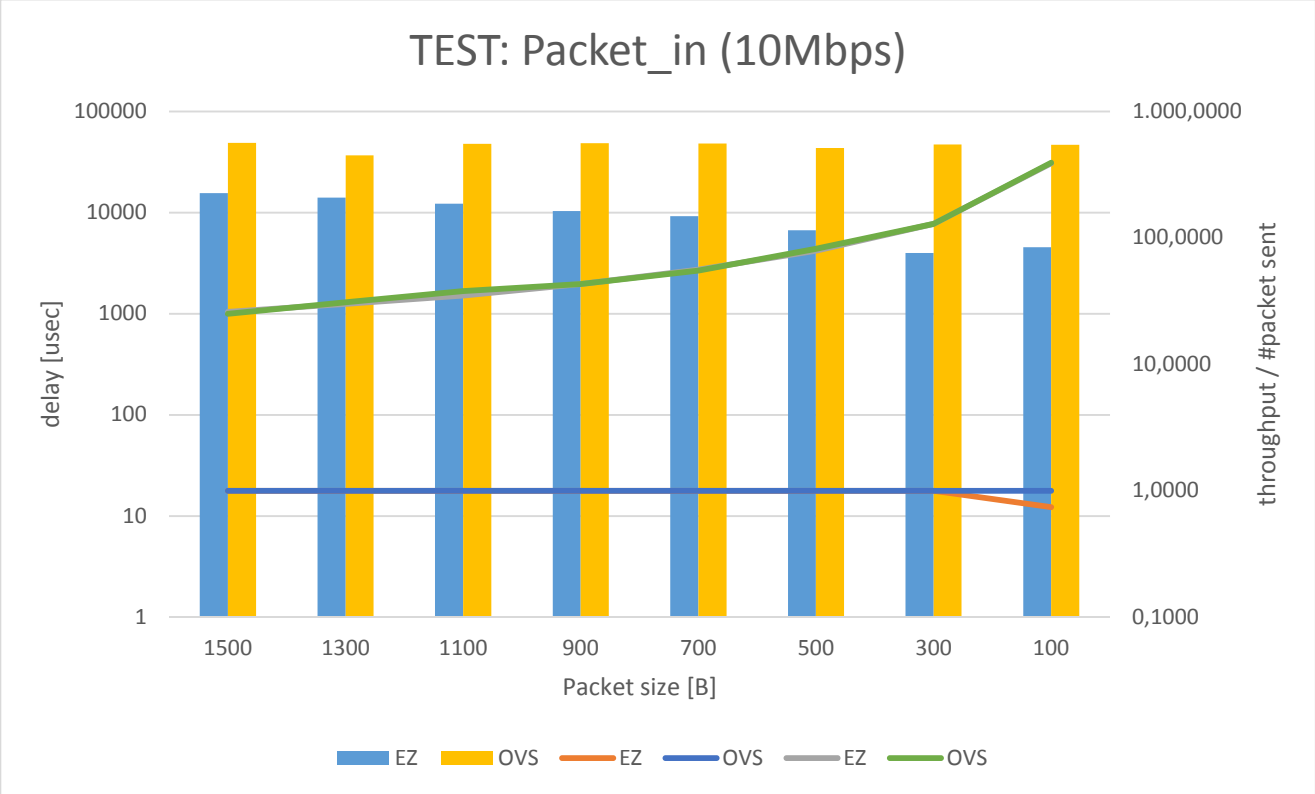


**Figure 1-28 Performance tests of EZappliance (packet_in – 10Mbps)**

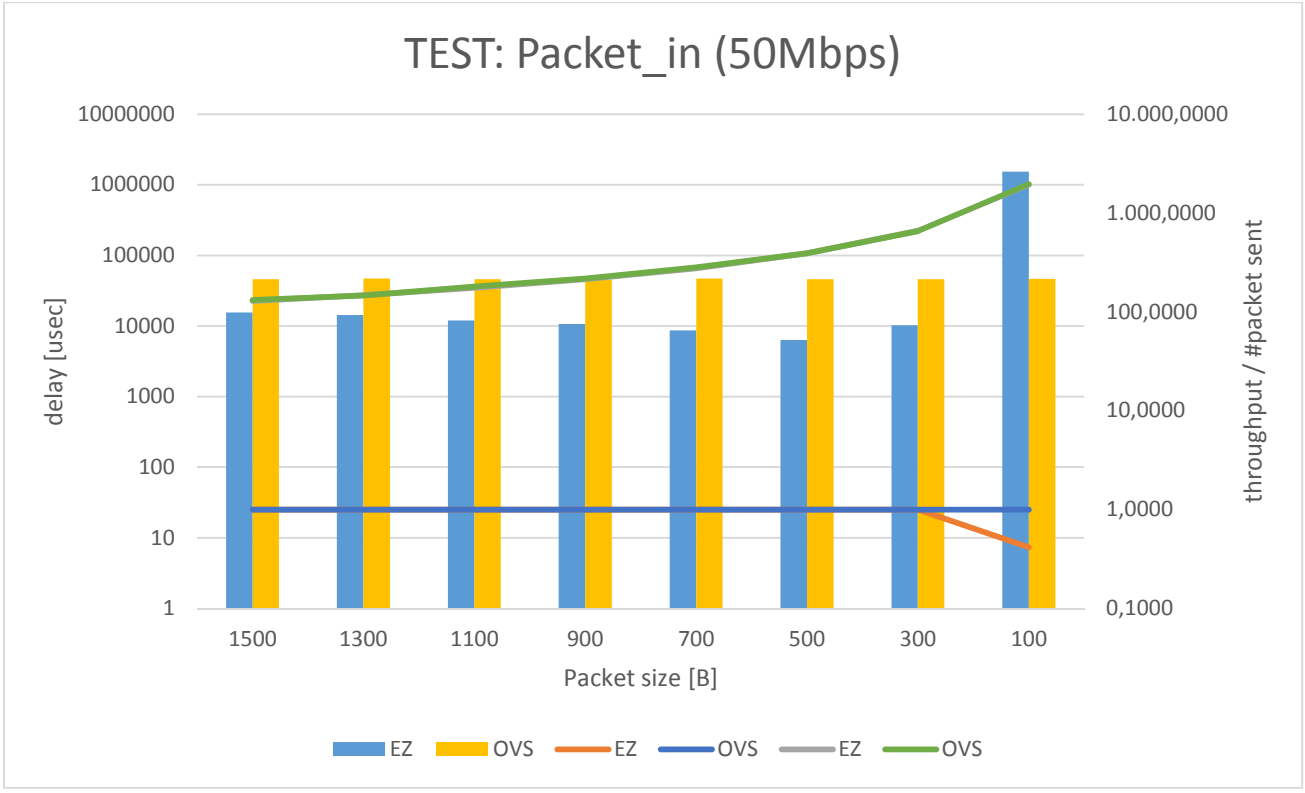


**Figure 1-29 Performance tests of EZappliance (packet_in – 50Mbps)**

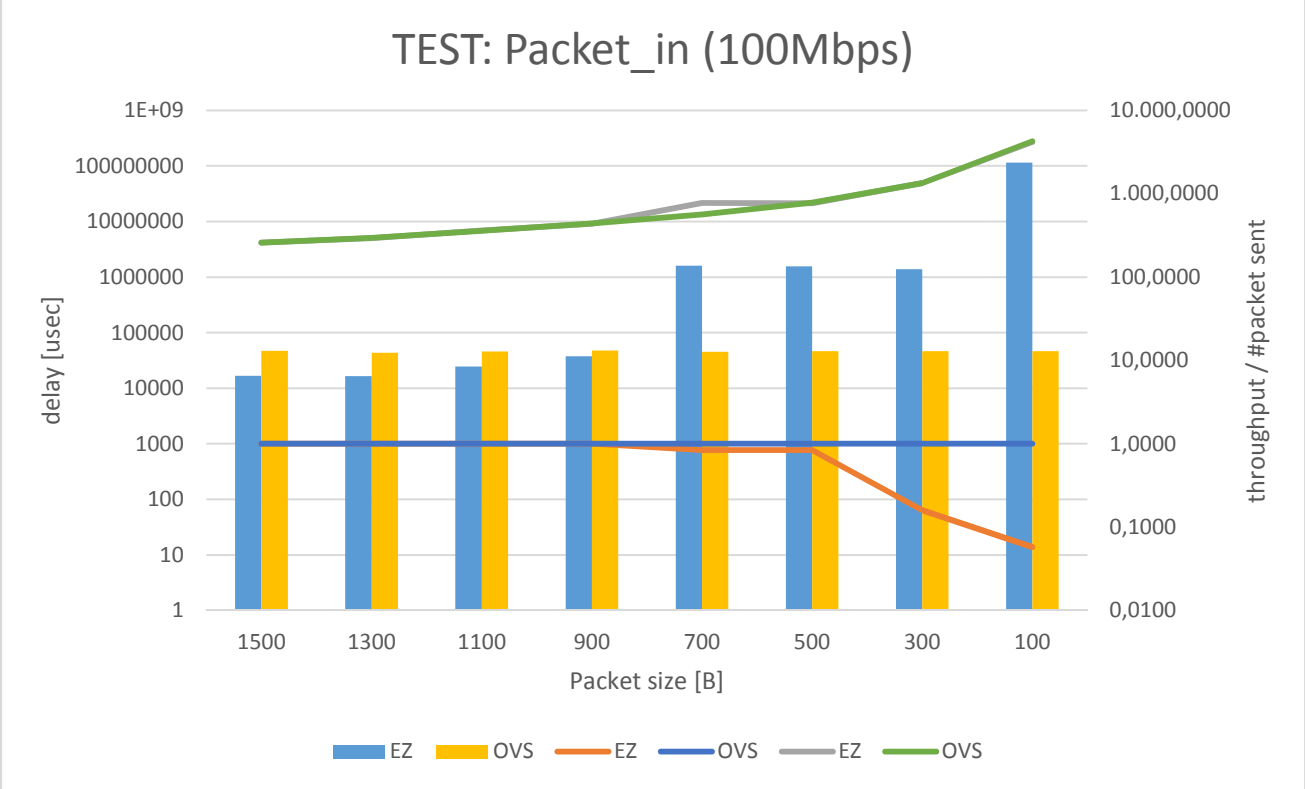


**Figure 1-30 Performance tests of EZappliance (packet_in – 100Mbps)**

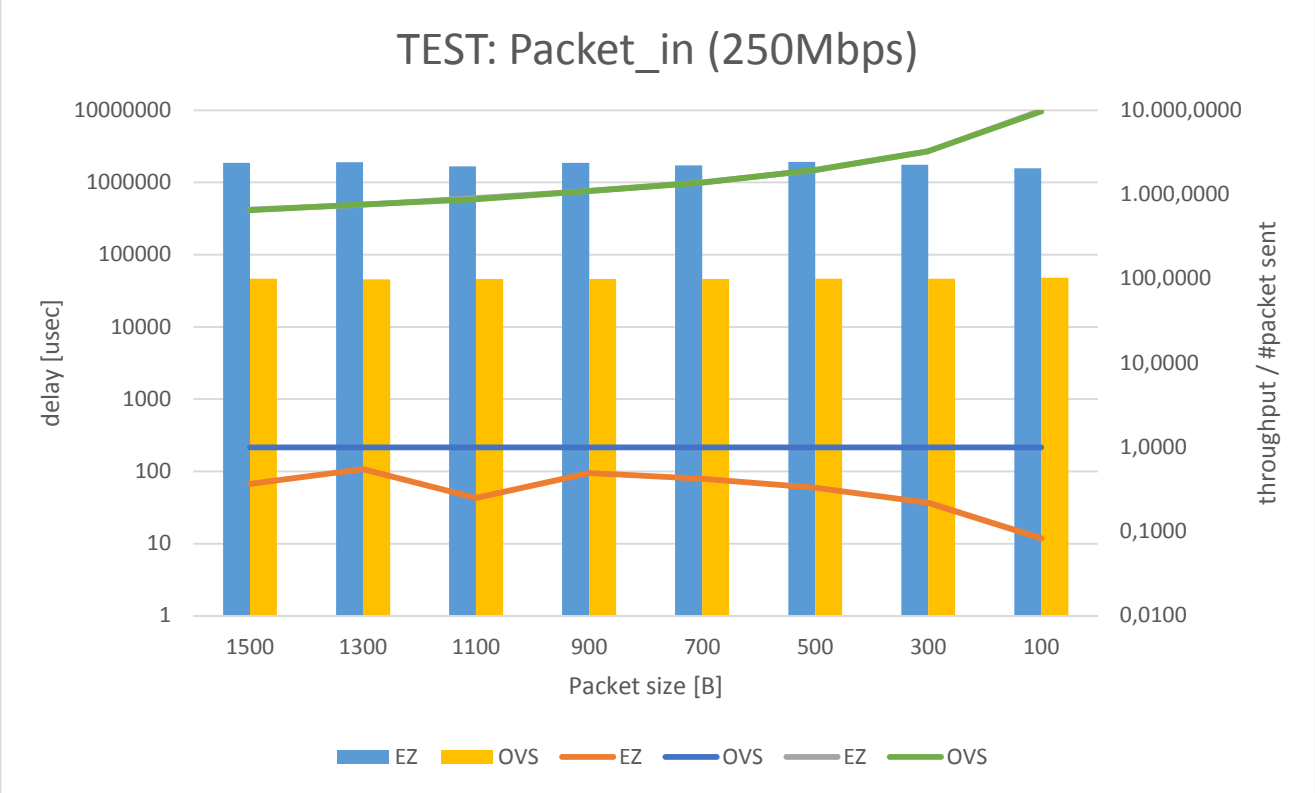


**Figure 1-31 Performance tests of EZappliance (packet_in – 250Mbps)**

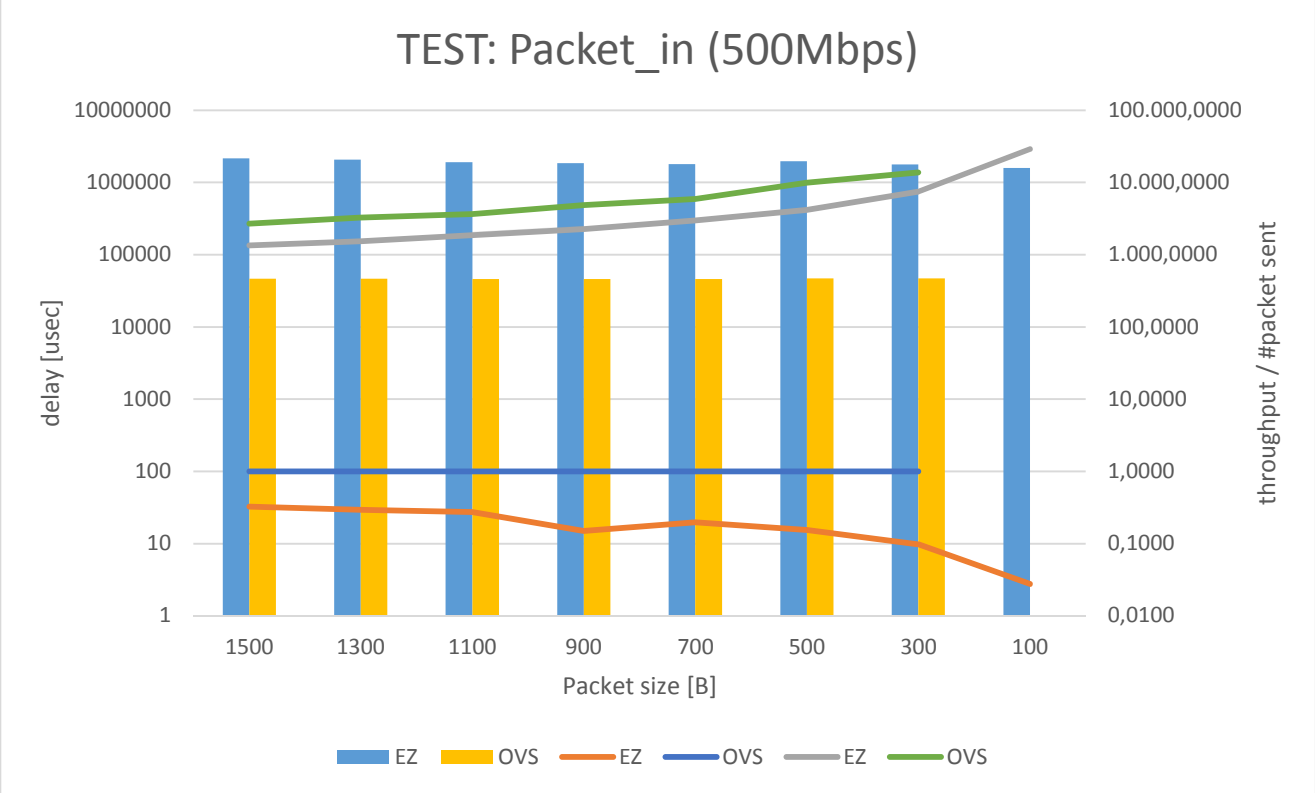


**Figure 1-32 Performance tests of EZappliance (packet_in – 500Mbps)**

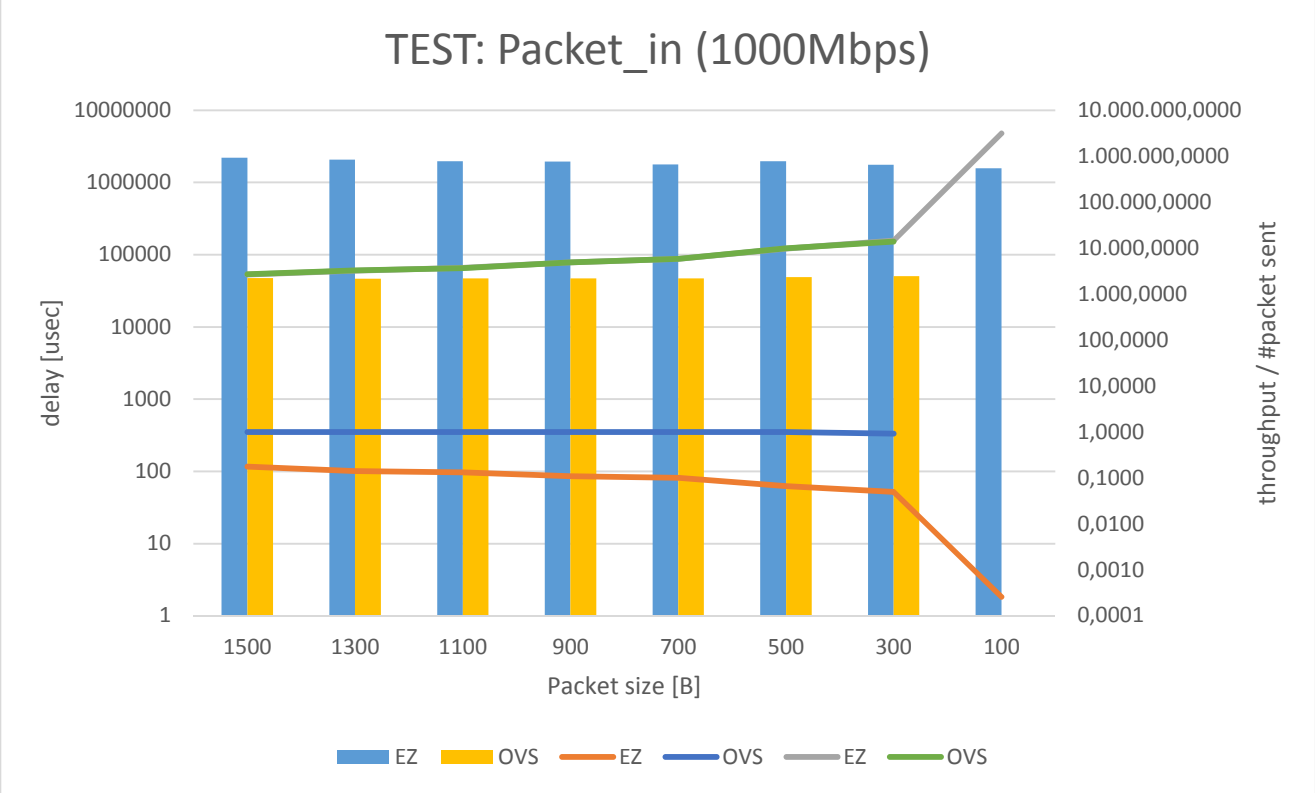


**Figure 1-33 Performance tests of EZappliance (packet_in – 1000Mbps)**

| Probe rate (probe_rate) | Packet size (pkt_size) | mean | median | standard deviation | throughput | #packet received | #packet send |
|---|---|---|---|---|---|---|---|
| [Mbps] | [B] | [µs] | [µs] | [µs] | [/] | [/] | [/] |
| 10 | 1500 | 15534 | 15341 | 1102 | 1,0000 | 26 | 26 |
| 10 | 1300 | 14072 | 13894 | 1009 | 1,0000 | 30 | 30 |
| 10 | 1100 | 12223 | 12103 | 1034 | 1,0000 | 35 | 35 |
| 10 | 900 | 10362 | 10255 | 801 | 1,0000 | 43 | 43 |
| 10 | 700 | 9235 | 8612 | 3792 | 1,0000 | 56 | 56 |
| 10 | 500 | 6674 | 6896 | 747 | 1,0000 | 79 | 79 |
| 10 | 300 | 3984 | 3816 | 700 | 1,0000 | 129 | 129 |
| 10 | 100 | 4535 | 2323 | 19349 | 0,7429 | 289 | 389 |
| 50 | 1500 | 15541 | 15482 | 5054 | 1,0000 | 129 | 129 |
| 50 | 1300 | 14259 | 13733 | 5754 | 1,0000 | 148 | 148 |
| 50 | 1100 | 12032 | 11987 | 699 | 1,0000 | 175 | 175 |
| 50 | 900 | 10739 | 10231 | 6586 | 1,0000 | 215 | 215 |
| 50 | 700 | 8673 | 8634 | 768 | 1,0000 | 276 | 276 |
| 50 | 500 | 6359 | 6144 | 1299 | 1,0000 | 391 | 391 |
| 50 | 300 | 10250 | 3811 | 26300 | 1,0000 | 660 | 660 |
| 50 | 100 | 1541228 | 1604160 | 282754 | 0,4137 | 800 | 1934 |
| 100 | 1500 | 16766 | 15237 | 13011 | 1,0000 | 260 | 260 |
| 100 | 1300 | 16569 | 13640 | 16637 | 1,0000 | 294 | 294 |
| 100 | 1100 | 24532 | 11943 | 50149 | 1,0000 | 361 | 361 |
| 100 | 900 | 37526 | 10187 | 79349 | 1,0000 | 435 | 435 |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 100 | 700 | 1597873 | 2003097 | 635419 | 0,8385 | 649 | 774 |
| 100 | 500 | 1572106 | 1998866 | 629196 | 0,8396 | 644 | 767 |
| 100 | 300 | 1385140 | 1786855 | 567947 | 0,1597 | 215 | 1346 |
| 100 | 100 | 115498487 | 115394325 | 1715982 | 0,0579 | 240 | 4142 |
| 250 | 1500 | 1873343 | 2267047 | 740970 | 0,3699 | 246 | 665 |
| 250 | 1300 | 1917489 | 2129081 | 521689 | 0,5503 | 416 | 756 |
| 250 | 1100 | 1679046 | 1980678 | 561956 | 0,2506 | 227 | 906 |
| 250 | 900 | 1874412 | 1999112 | 404125 | 0,4968 | 549 | 1105 |
| 250 | 700 | 1733803 | 1825408 | 331854 | 0,4235 | 592 | 1398 |
| 250 | 500 | 1924350 | 2027533 | 359675 | 0,3326 | 645 | 1939 |
| 250 | 300 | 1769380 | 1814656 | 311488 | 0,2201 | 715 | 3248 |
| 250 | 100 | 1580046 | 1613617 | 229694 | 0,0830 | 829 | 9982 |
| 500 | 1500 | 2164890 | 2294907 | 428905 | 0,3269 | 440 | 1346 |
| 500 | 1300 | 2076465 | 2143587 | 429624 | 0,2958 | 455 | 1538 |
| 500 | 1100 | 1905350 | 1994432 | 337124 | 0,2729 | 507 | 1858 |
| 500 | 900 | 1861236 | 2003723 | 415716 | 0,1505 | 340 | 2259 |
| 500 | 700 | 1794377 | 1831426 | 325259 | 0,1983 | 591 | 2981 |
| 500 | 500 | 1962652 | 2032041 | 332138 | 0,1541 | 645 | 4185 |
| 500 | 300 | 1779813 | 1812014 | 294726 | 0,0980 | 733 | 7481 |
| 500 | 100 | 1582667 | 1613236 | 229877 | 0,0277 | 808 | 29182 |
| 1000 | 1500 | 2207418 | 2299534 | 401273 | 0,1761 | 433 | 2666 |
| 1000 | 1300 | 2081058 | 2148332 | 371275 | 0,1414 | 457 | 3232 |
| 1000 | 1100 | 1965055 | 2002458 | 358257 | 0,1337 | 499 | 3731 |
| 1000 | 900 | 1955063 | 2007567 | 329973 | 0,1097 | 543 | 4948 |
| 1000 | 700 | 1787839 | 1834156 | 284612 | 0,1003 | 592 | 5902 |
| 1000 | 500 | 1962701 | 2033702 | 313863 | 0,0664 | 645 | 9709 |
| 1000 | 300 | 1754274 | 1811624 | 257843 | 0,0498 | 732 | 14696 |
| 1000 | 100 | 1573640 | 1614722 | 218358 | 0,0003 | 817 | 3140195 |

**Table 1-7 Performance tests of EZappliance – Packet_in results**

We can observe that packet_in message delay vary from milliseconds to hundreds of milliseconds depends on the packet size and probe rate. For the fixed probe rate (in Mbps) the smaller size of the packet means more UDP packets and packet_in messages generated during the test (60 seconds). For the system with throughput equal 1 the smaller size of the packets the smaller medium delay. For setup with many packet_in messages generated we observed packet losses (throughput<1). Although measured delay includes also data path delay (path from testing platform to EZappliance device) it is constant and negligible during the test.

Packet_in messages are generated by OpenFlow switch only when it does not find proper entry in the flow table for a frame/packet. Results of this test together with packet_out test presented bellow shows responsiveness of the system for a new (unknown) flow.

### 2) Packet_out

For Packet_out test ALIEN OFLOPS framework generates OFPT_PACKET_OUT messages that are received by xDPd for EZappliance OpenFlow endpoint. After that xDPd for EZappliance parse the message and send packet with 3B header through TCP socket between EZproxy and xDPd modules. 3B header contains out port number and packet size information. EZproxy uses EZChip API to NP-3 and send packets directly on the out port (see Figure 1-34).

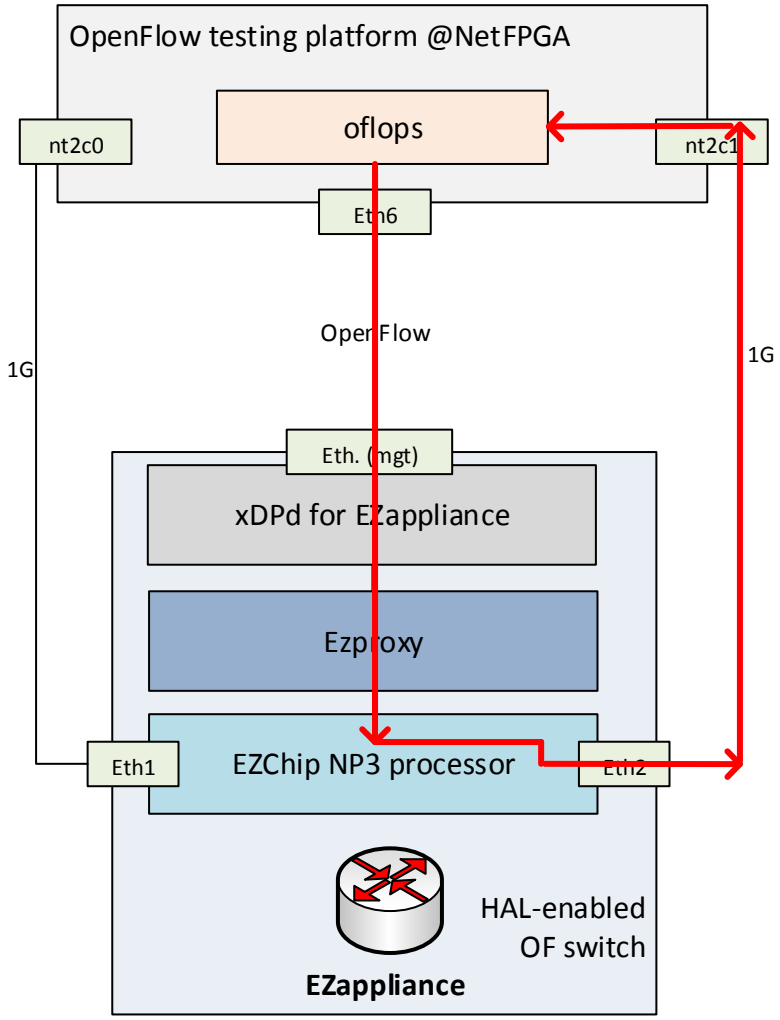


**Figure 1-34 Performance tests of EZappliance – packet_out**

*Test conditions*

Packet_out test was done for probe rates: 10, 50, 100 and 250 Mbps. This parameter controls the data rate of the measurement probe. For all of these rates the following packet size was tested: 100, 300, 500, 700, 900, 1100, 1300 and 1500B.

The test duration: 60 seconds.

*Test measurements*

During packet_out test the following parameters was measured:

- Mean and median delay of packet_out messages together with standard deviation – measured in µs. This delay includes also delay of data path (path between eth2 of EZappliance and nt2c1 of NetFPGA) but it is constant and negligible during the test.

- Throughput – calculated as number of packet_sent to number of packet_received by ALIEN OFLOPS. Throughput factor equal 1 means: “all UDP transmitted by ALIEN OFLOPS packets was received”.

- Number of packets – number of packets generated by ALIEN OFLOPS during test (60 seconds).

Interval between UDP packet sent may be calculated in the following way:

data_snd_interval = (pkt_size * byte_to_bits * sec_to_µs) / (datarate * mbits_to_bits);

The results of packet_out test were compared with OVS implementation.

*Test results*

On the next four charts results of packet_out test are presented for the following data rates: 10, 50, 100 and 250Mbps. In case of 500 and 1000Gbps xDPd reset TCP connection to ALIEN OFLOPS.

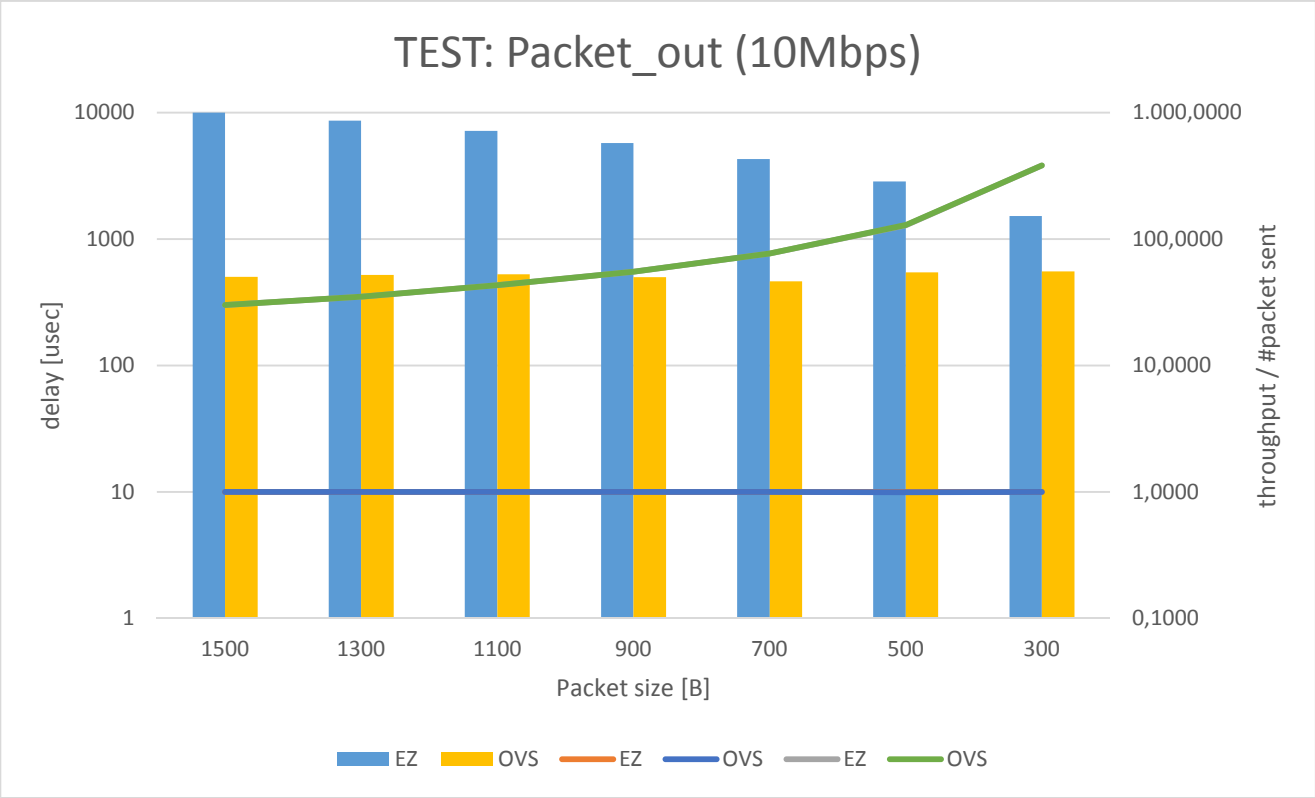


**Figure 1-35 Performance tests of EZappliance (packet_out – 10Mbps)**

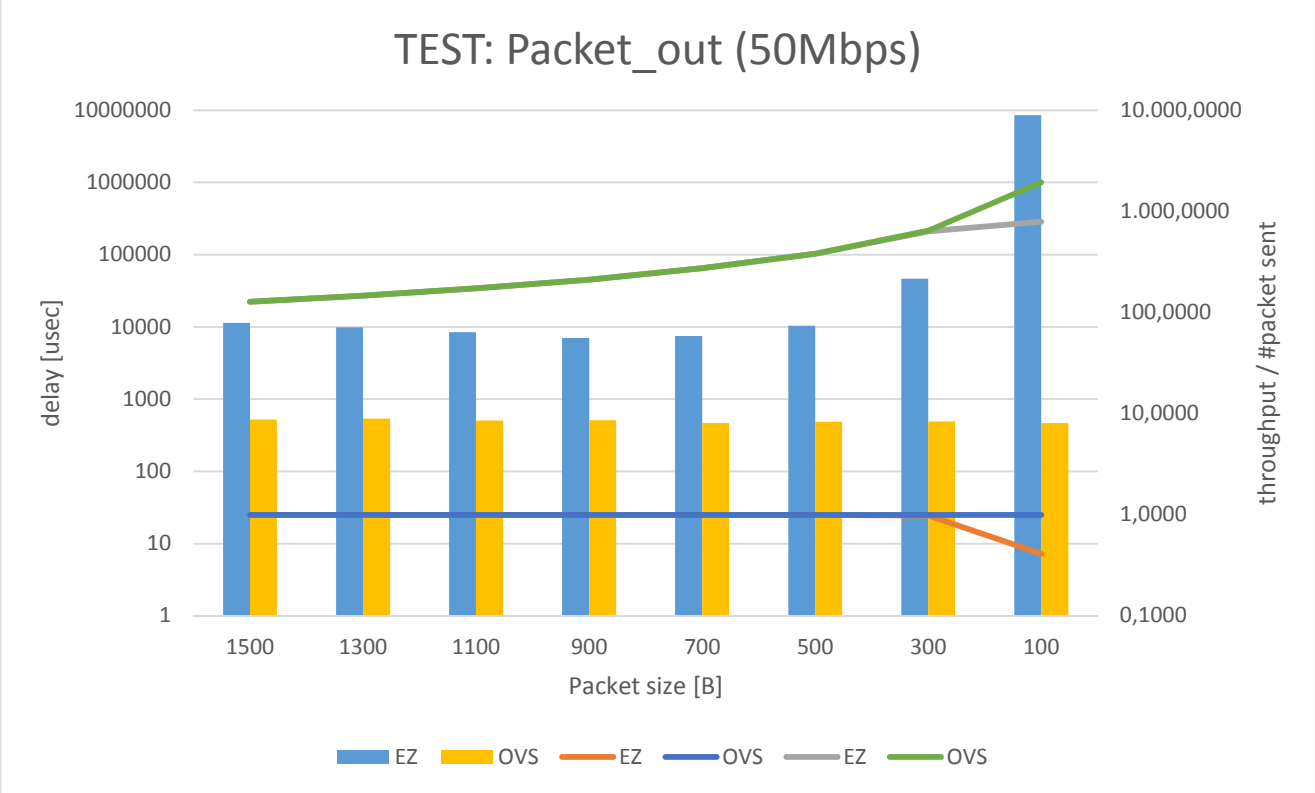


**Figure 1-36 Performance tests of EZappliance (packet_out – 50Mbps)**

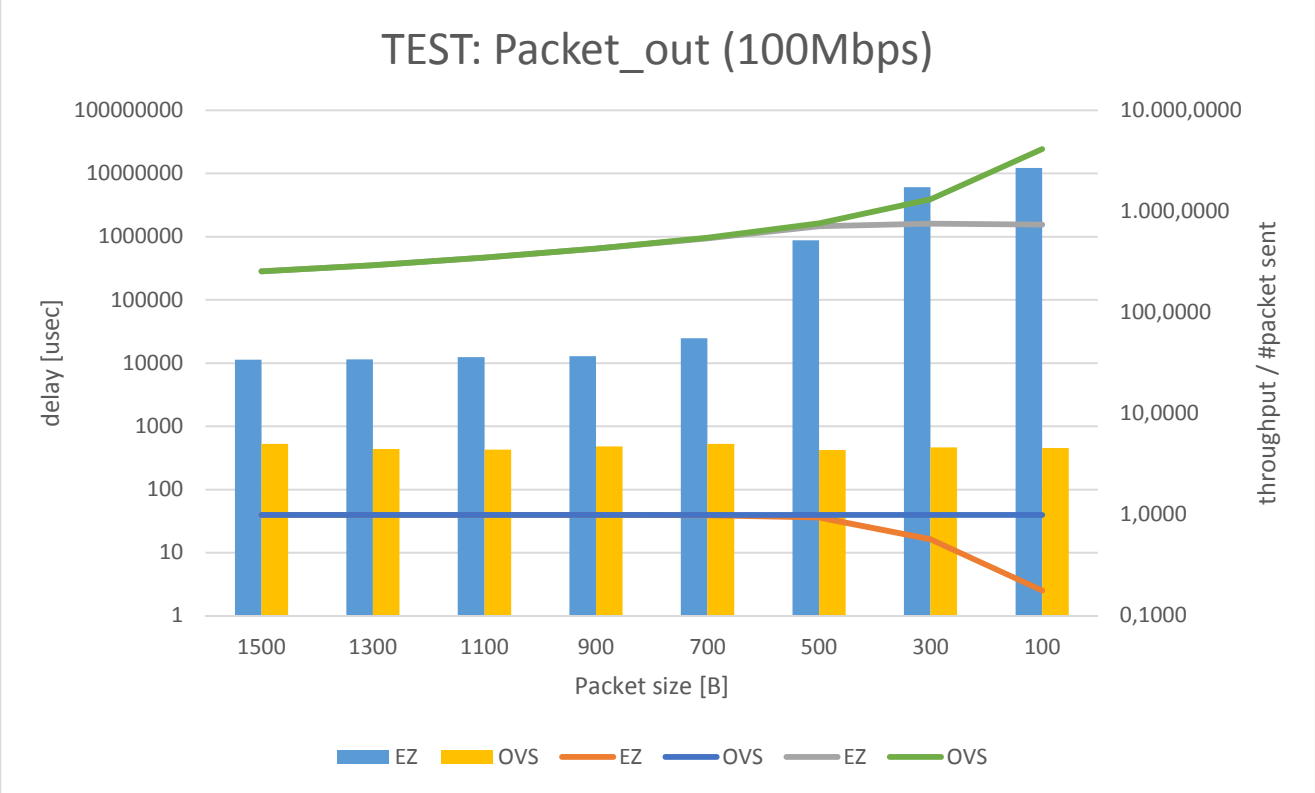


**Figure 1-37 Performance tests of EZappliance (packet_out – 100Mbps)**

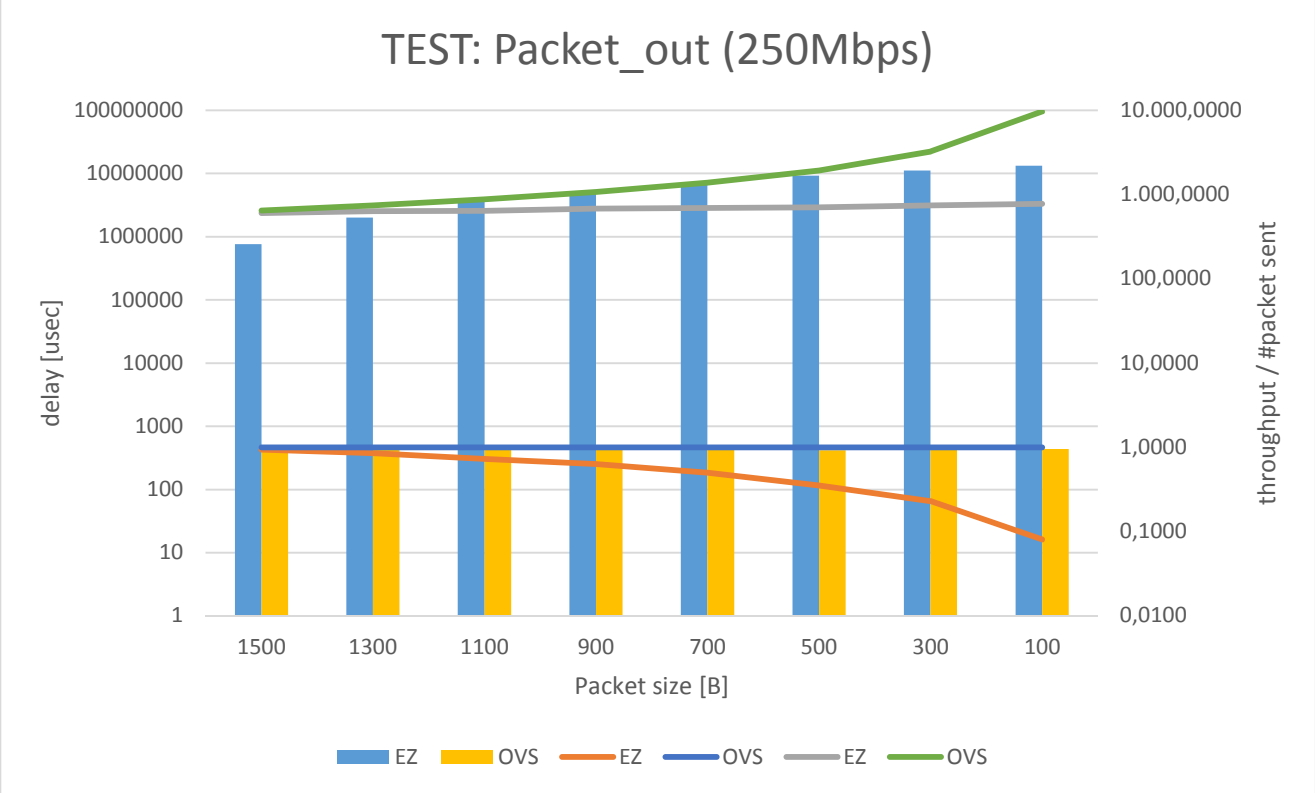


**Figure 1-38 Performance tests of EZappliance (packet_out – 250Mbps)**

| Probe rate (probe_rate) | Packet size (pkt_size) | mean | median | standard deviation | throughput | #packet |
|---|---|---|---|---|---|---|
| [Mbps] | [B] | [usec] | [usec] | [usec] | [/] | [/] |
| 10 | 1500 | 11338 | 11393 | 215 | 1,0000 | 26 |
| 10 | 1400 | 10785 | 10701 | 432 | 1,0000 | 28 |
| 10 | 1300 | 9999 | 10001 | 201 | 1,0000 | 30 |
| 10 | 1200 | 9234 | 9260 | 172 | 1,0000 | 32 |
| 10 | 1100 | 8644 | 8630 | 144 | 1,0000 | 35 |
| 10 | 1000 | 7547 | 7834 | 145 | 1,0000 | 39 |
| 10 | 900 | 7157 | 7140 | 171 | 1,0000 | 43 |
| 10 | 800 | 6403 | 6412 | 127 | 1,0000 | 48 |
| 10 | 700 | 5742 | 5742 | 135 | 1,0000 | 55 |
| 10 | 600 | 4998 | 504 | 112 | 1,0000 | 64 |
| 10 | 500 | 4284 | 4287 | 105 | 1,0000 | 77 |
| 10 | 400 | 3555 | 3550 | 91 | 1,0000 | 96 |
| 10 | 300 | 2852 | 2855 | 84 | 1,0000 | 128 |
| 10 | 200 | 2262 | 2140 | 1414 | 1,0000 | 191 |
| 10 | 100 | 1516 | 1491 | 421 | 1,0000 | 382 |
| 50 | 1500 | 11441 | 11407 | 205 | 1,0000 | 128 |
| 50 | 1400 | 10564 | 10581 | 287 | 1,0000 | 137 |
| 50 | 1300 | 9925 | 9921 | 269 | 1,0000 | 147 |
| 50 | 1200 | 9184 | 9151 | 346 | 1,0000 | 159 |

| | | | | | | |
|---|---|---|---|---|---|---|
| 50 | 1100 | 8462 | 8474 | 177 | 1,0000 | 174 |
| 50 | 1000 | 7788 | 7794 | 158 | 1,0000 | 191 |
| 50 | 900 | 7084 | 7084 | 242 | 1,0000 | 212 |
| 50 | 800 | 7457 | 6352 | 10719 | 1,0000 | 238 |
| 50 | 700 | 7534 | 5633 | 17082 | 1,0000 | 274 |
| 50 | 600 | 7063 | 4953 | 18309 | 1,0000 | 319 |
| 50 | 500 | 10432 | 4242 | 38923 | 1,0000 | 382 |
| 50 | 400 | 10435 | 3461 | 40249 | 1,0000 | 476 |
| 50 | 300 | 46576 | 2790 | 136583 | 0,9814 | 633 |
| 50 | 200 | 1716213 | 2746691 | 1582887 | 0,8128 | 786 |
| 50 | 100 | 8571935 | 8382978 | 5049271 | 0,4084 | 790 |
| 100 | 1500 | 11363 | 11377 | 274 | 1,0000 | 255 |
| 100 | 1400 | 10663 | 10600 | 187 | 1,0000 | 274 |
| 100 | 1300 | 11414 | 9969 | 13228 | 1,0000 | 293 |
| 100 | 1200 | 11659 | 9190 | 19979 | 1,0000 | 319 |
| 100 | 1100 | 12362 | 8512 | 28259 | 1,0000 | 350 |
| 100 | 1000 | 13749 | 4490 | 38605 | 1,0000 | 382 |
| 100 | 900 | 12840 | 7107 | 34980 | 1,0000 | 427 |
| 100 | 800 | 14912 | 6348 | 45841 | 1,0000 | 476 |
| 100 | 700 | 24660 | 5591 | 73158 | 0,9891 | 542 |
| 100 | 600 | 40783 | 4917 | 90984 | 1,0000 | 645 |
| 100 | 500 | 879273 | 854979 | 566871 | 0,9371 | 716 |
| 100 | 400 | 3300362 | 3205699 | 1998436 | 0,7704 | 745 |
| 100 | 300 | 6043827 | 5816196 | 3608185 | 0,5747 | 758 |
| 100 | 200 | 8699188 | 8525521 | 5194296 | 0,3940 | 762 |
| 100 | 100 | 12206298 | 12266553 | 6908672 | 0,1786 | 740 |
| 250 | 1500 | 762383 | 910128 | 404458 | 0,9380 | 605 |
| 250 | 1400 | 1770876 | 1721142 | 915869 | 0,8915 | 616 |
| 250 | 1300 | 2014220 | 1784674 | 1288105 | 0,8562 | 637 |
| 250 | 1200 | 2996630 | 3371355 | 1858057 | 0,7940 | 640 |
| 250 | 1100 | 3976712 | 3877266 | 2320993 | 0,7270 | 639 |
| 250 | 1000 | 4249668 | 4248928 | 2441689 | 0,7063 | 683 |
| 250 | 900 | 5184766 | 5060932 | 3099257 | 0,6335 | 681 |
| 250 | 800 | 6314866 | 6376938 | 3686550 | 0,5616 | 679 |
| 250 | 700 | 7149137 | 7286982 | 4062426 | 0,5025 | 694 |
| 250 | 500 | 9287929 | 9232064 | 5389083 | 0,3545 | 705 |
| 250 | 300 | 11192068 | 11262685 | 6523404 | 0,2308 | 744 |
| 250 | 100 | 13319713 | 13368067 | 7578452 | 0,0803 | 776 |

**Table 1-8 Performance tests of EZappliance – Packet_out results**

Results of this test together with packet_in test shows responsiveness of the system for a new (unknown) flows. In case of 500 and 1000Gbps xDPd reset TCP connection to ALIEN OFLOPS. This is because of efficiency of connection between xDPd and EZproxy (Corba technology) and between EZproxy and NP-3 (EZChip API) – for the 250Mbps and big number of packet_out events the mean delay is over 10 second.

### 3) Data path delay

The data path delay tests measure how much time packet requires to be transmitted via EZappliance device. Delay between tester and EZappliance is constant and negligible during the test.

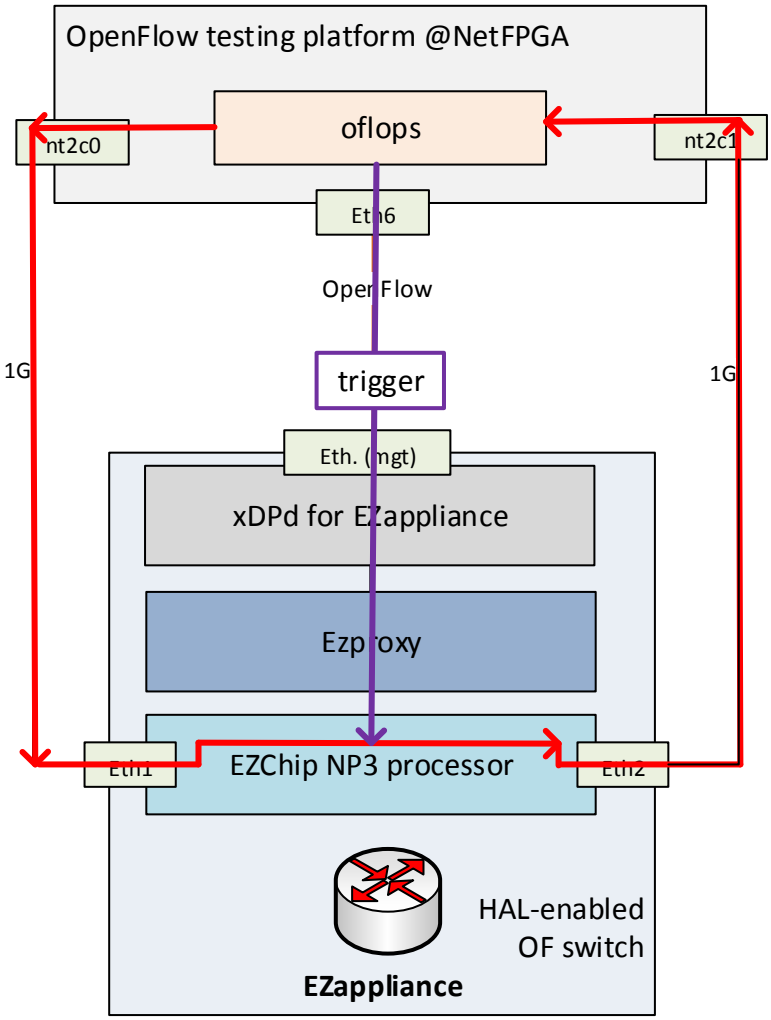


**Figure 1-39 Performance tests of EZappliance – data path delay**

*Test conditions*

The test duration: 30 seconds.

*Test measurements*

During data path delay tests the following parameters was measured:

- Mean and median delay of data plane packet switching together with standard deviation – measured in µs. This delay includes also delay of data path (path between nt2c0 of NetFPGA and eth1 of EZappliance and nt2c1 of NetFPGA and eth2 of EZappliance) but it is constant and negligible during the test.

- Throughput – calculated as number of packet_sent to number of packet_received by ALIEN OFLOPS. Throughput factor equal 1 means: “all UDP transmitted by ALIEN OFLOPS packets was then received”.

- Number of packets – number of packets generated by ALIEN OFLOPS during test (60 seconds).

Interval between UDP packet sent may be calculated in the following way:

data_snd_interval = (pkt_size * byte_to_bits * sec_to_µs) / (datarate * mbits_to_bits);

The results of path_delay test were compared with OVS implementation.

*Test results*

On the next three charts results of Data path delay test are presented for the following data rates: 10, 100, 1000Mbps.

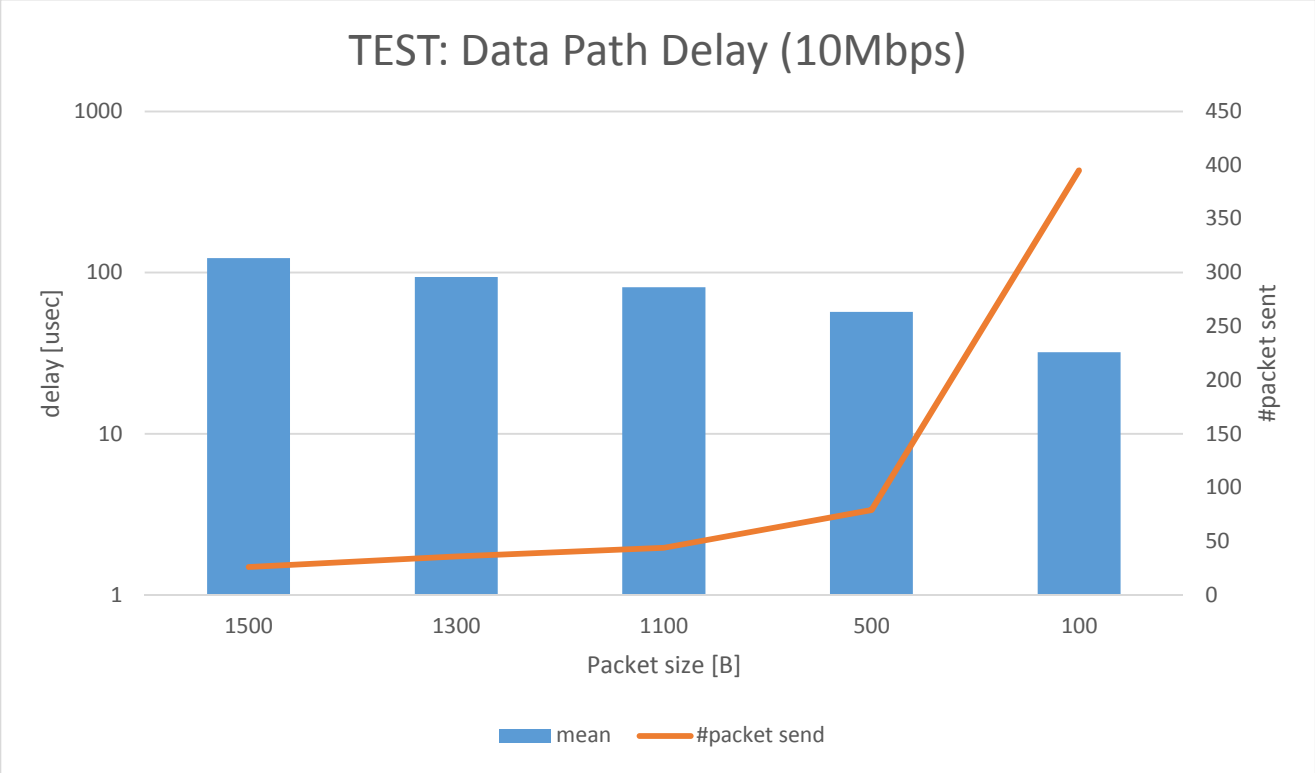


**Figure 1-40 Performance tests of EZappliance (Data path delay – 10Mbps)**

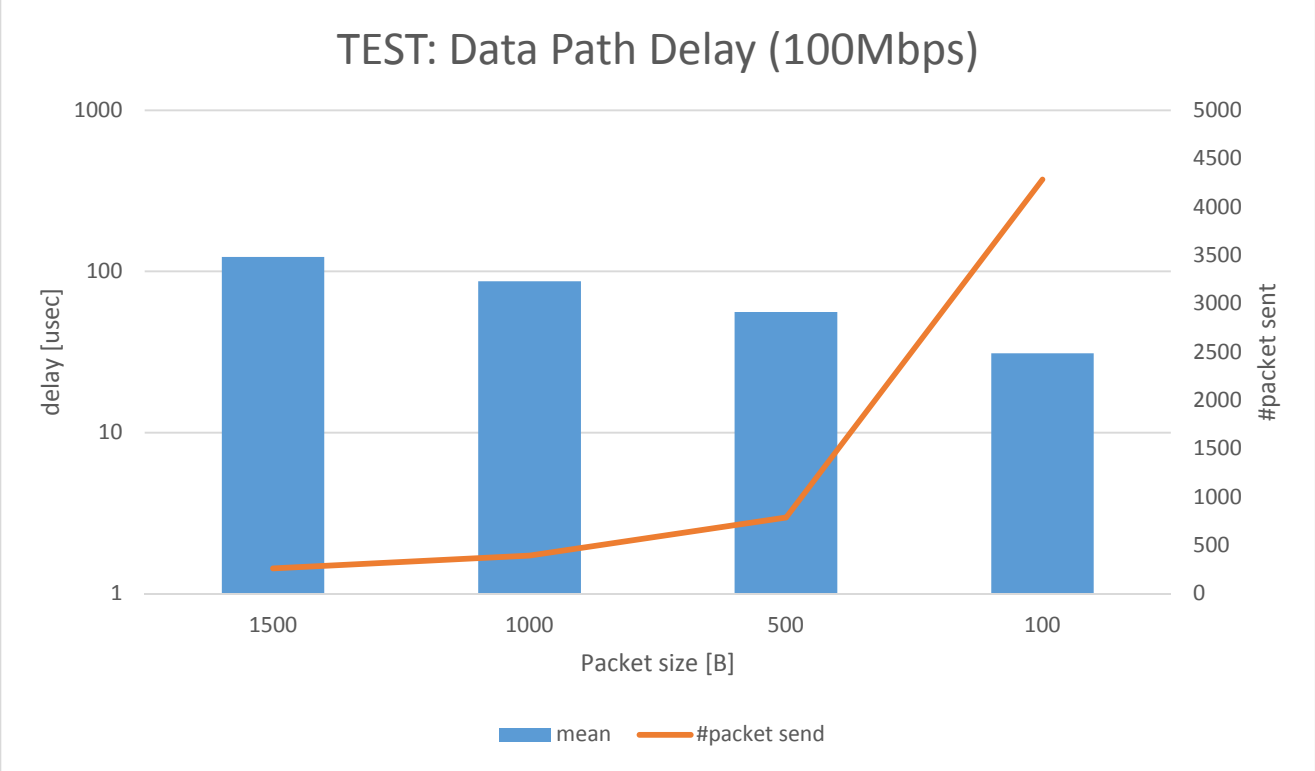


**Figure 1-41 Performance tests of EZappliance (Data path delay – 100Mbps)**

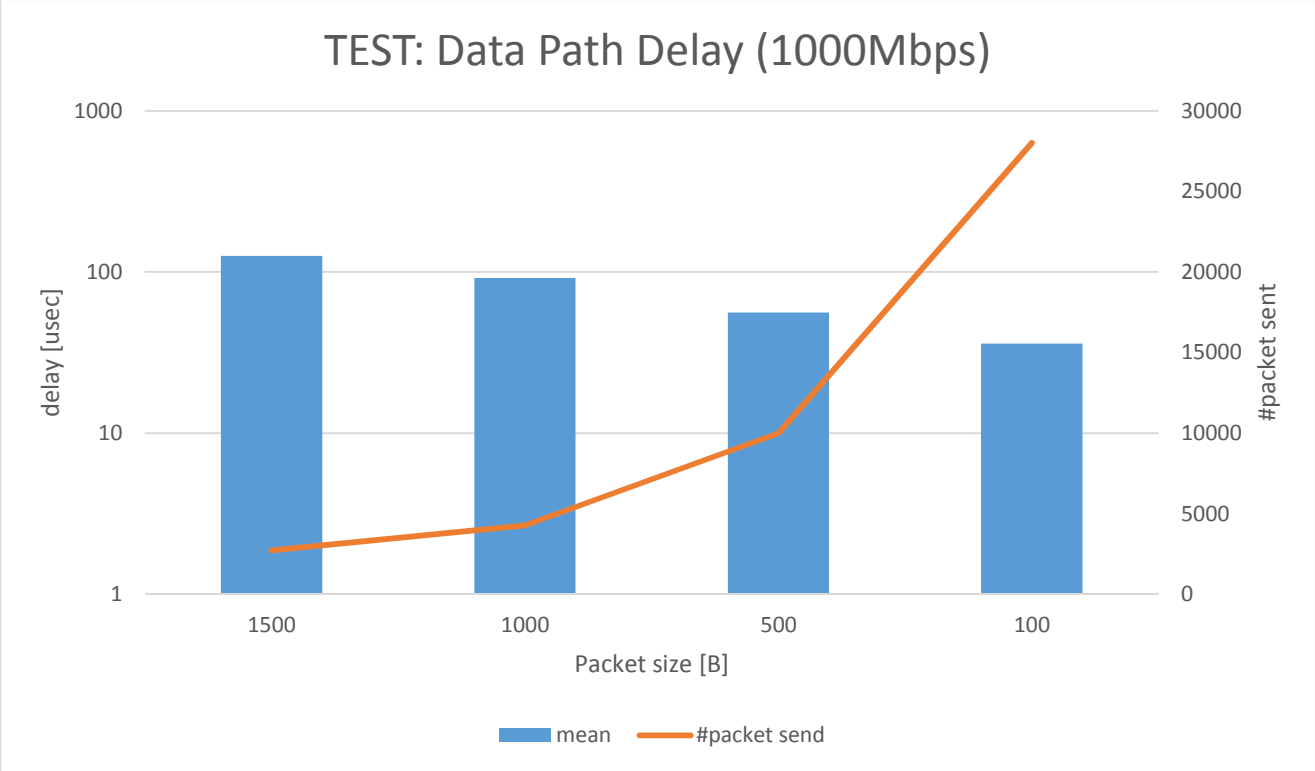


**Figure 1-42 Performance tests of EZappliance (Data path delay – 1000Mbps)**

| Probe rate (probe_rate) [Mbps] | Packet size (pkt_size) [B] | mean [µs] | median [µs] | standard deviation [µs] | throughput [/] | #packet send [/] |
|---|---|---|---|---|---|---|
| 10 | 1500 | 123 | 122 | 6 | 1 | 26 |
| 10 | 1300 | 94 | 95 | 0 | 1 | 36 |
| 10 | 1100 | 81 | 82 | 0 | 1 | 44 |
| 10 | 500 | 57 | 58 | 1 | 1 | 79 |
| 10 | 100 | 32 | 32 | 1 | 1 | 395 |
| 100 | 1500 | 123 | 117 | 11 | 1 | 263 |
| 100 | 1000 | 87 | 87 | 6 | 1 | 395 |
| 100 | 500 | 56 | 56 | 1 | 1 | 790 |
| 100 | 100 | 31 | 31 | 2 | 1 | 4286 |
| 1000 | 1500 | 126 | 120 | 14 | 1 | 2727 |
| 1000 | 1000 | 92 | 90 | 7 | 1 | 4286 |
| 1000 | 500 | 56 | 55 | 2 | 1 | 10000 |
| 1000 | 200 | 36 | 37 | 1 | 1 | 28000 |

**Table 1-9 Performance tests of EZappliance – Data path delay results**

The data path delay tests shows that data plane packets switching time for EZappliance device depends on frame length. The shortest measured frames (100 bytes) are switched in ~30 µs whereas the longest (1500 bytes) are processed in ~120 µs. That dependency is not linear. The measured values not depend on data rate.

### 4) Control channel delay

In the control channel delay test, the ALIEN OFLOPS tool sends OpenFlow Echo-requests to EZappliance device and measures delay in receiving the OpenFlow Echo-reply messages (see Figure 1-43).

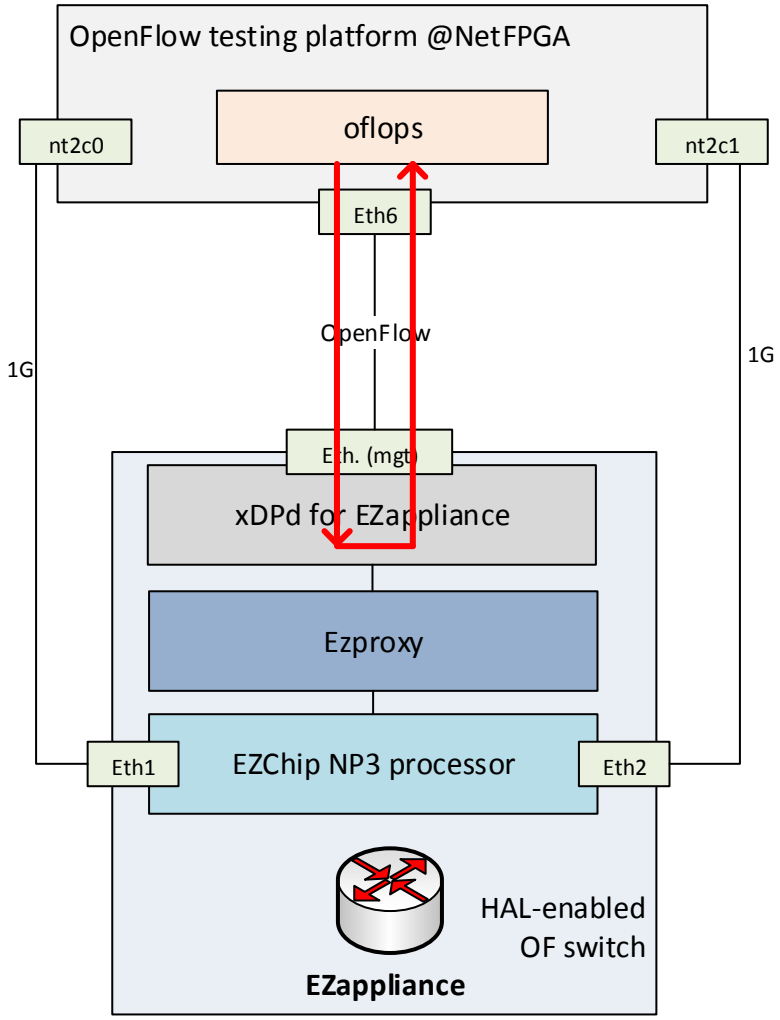


**Figure 1-43 Performance tests of EZappliance – control channel delay**

*Test conditions*

ALIEN OFLOPS was launch with parameter echo_rate =3.

*Test measurements*

During control channel delay tests the following parameters was measured:

- Mean and median delay of OpenFlow Echo-replay message with standard deviation – measured in µs. This delay includes also delay of connection (path between eth6 of NetFPGA and eth_mngt of EZappliance) but it is constant and negligible during the test.

- Throughput – calculated as number of packet_sent to number of packet_received by ALIEN OFLOPS .Throughput factor equal 1 means: "all UDP transmitted by ALIEN OFLOPS packets was received ".

- Number of packets – number of packets generated by ALIEN OFLOPS during test (60 seconds).

Interval between UDP packet sent may be calculated in the following way:

data_snd_interval = (pkt_size * byte_to_bits * sec_to_µs) / (datarate * mbits_to_bits);

The results of Control_channel_delay test were compared with OVS implementation.

*Test results*

| | mean | median | standard deviation | throughput | #packet |
|---|---|---|---|---|---|
| Control channel delay | [µs] | [µs] | [µs] | [/] | [/] |
| | OpenVSwitch | | | | |
| | 354 | 341 | 6323 | 1 | 26 |
| | EZappliance | | | | |
| | 394 | 391 | 45 | 1 | 9 |

**Table 1-10 Performance tests of EZappliance – Control channel delay results**

In our test, the measured control channel delay is 394 µs which is time required to xDPd and ROFL software to generate OpenFlow replay. This pure software processing make the results quite similar to result achieved for OVS.

### 5) Add_flow

Add_flow test measures latency between a new flow entry generation by OpenFlow controller and actual setup of flow rules in the datapath (i.e.: EZchip NP-3 processor). In this test, the ALIEN OFLOPS framework generates UDP packets and sends them to the first data plane port of EZappliance device. In mean time starts sending a set of OpenFlow flowmod messages to EZappliance with new flow entries. When the last flow entry generated by ALIEN OFLOPS is installed in EZchip network processor by EZproxy then UDP packets are started to be switched to second port of EZappliance and are received by ALIEN OFLOPS on second data plane port (see Figure 44).

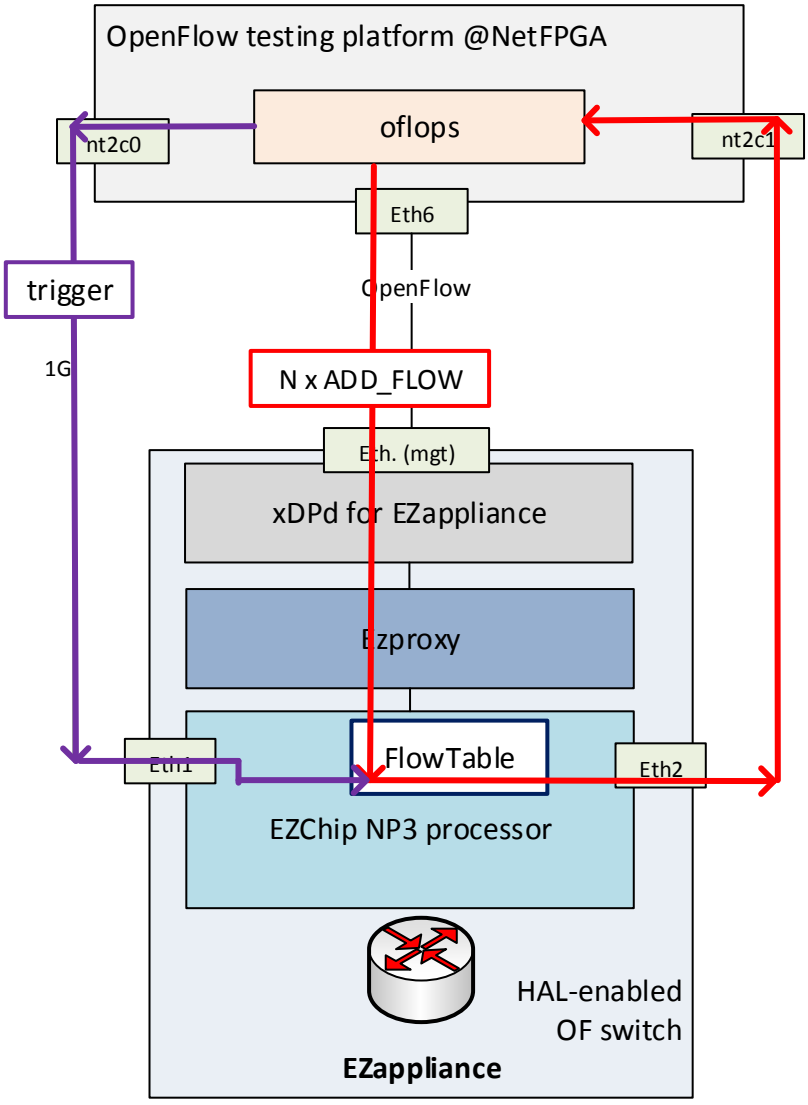


**Figure 1-44 Performance tests of EZappliance – add flow delay**

*Test conditions*

add_flow test was done for number of flows installed into EZappliance: 1, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100.

Additional test was done to measure single flow installation time for number of flows: 1-1000 into TCAM memory of EZappliance.

The test duration: 120 seconds.

*Test measurements*

During add_flow test the following parameters was measured:

- Mean and median delay of the installation of a given number of flow entries in NPU, together with standard deviation – measured in µs. This delay includes also delay of data path (path between nt2c0 of NetFPGA and eth1 of EZappliance) but it is constant and negligible during the test.

- Delay per single flow – calculated from mean delay and number of installed flows.

Interval between UDP packet sent may be calculated in the following way:

data_snd_interval = (pkt_size * byte_to_bits * sec_to_µs) / (datarate * mbits_to_bits);

The results of add_flow test were compared with OVS implementation.

*Test results*

Results of add_flow test are presented on a single chart.

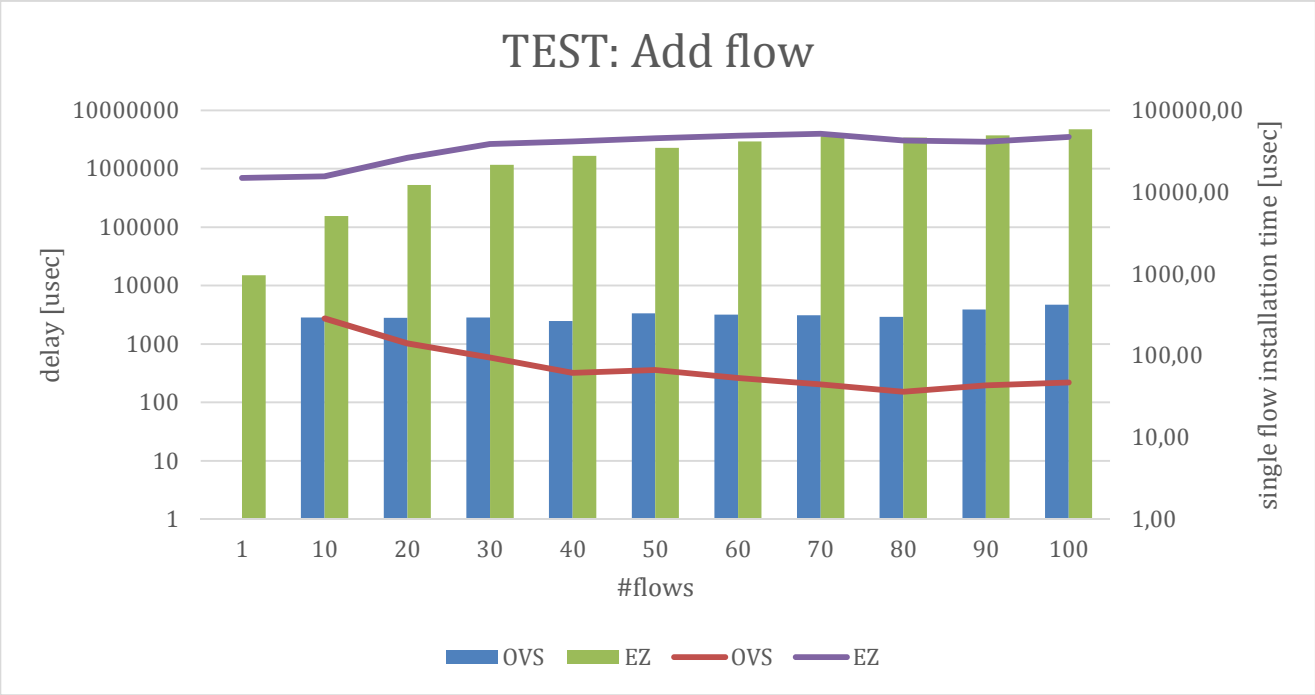


**Figure 1-45 Performance tests of EZappliance (Add flow)**

| Flows | mean | median | standard deviation | delay per flow | Conditions |
|---|---|---|---|---|---|
| [/] | [µs] | [µs] | [µs] | [msec] | |
| 1 | 14936 | 14934 | 7 | 14.9 | probe_rate=200, data_rate= 200, pkt_size=400 |
| 10 | 154885 | 146786 | 23078 | 15.5 | probe_rate=30, data_rate= 30, pkt_size=600 |
| 20 | 523291 | 549911 | 58061 | 26.2 | |

| | | | | | |
|---|---|---|---|---|---|
| 30 | 1162310 | 1009956 | 113335 | 38.7 | probe_rate=30, data_rate= 30, pkt_size=400 |
| 40 | 1658019 | 1615951 | 107035 | 41.4 | |
| 50 | 2274842 | 2221950 | 208504 | 45.4 | |
| 60 | 2940155 | 2878452 | 237216 | 49.0 | |
| 70 | 3623311 | 3534939 | 294745 | 51.8 | |
| 80 | 3401253 | 3334935 | 205869 | 42.5 | probe_rate=30, data_rate= 30, pkt_size=600 |
| 90 | 3704700 | 3653947 | 216114 | 41.2 | probe_rate=30, data_rate= 30, pkt_size=800 |
| 100 | 4715910 | 4576924 | 421603 | 47.1 | probe_rate=2, data_rate= 2, pkt_size=1200 |

**Table 1-11 Performance tests of EZappliance - Add flow results**

We can observe that flow installation in EZappliance is much slower (more than two orders of magnitude) in comparison to OVS software switch. Such difference is caused by fact that EZappliance is storing flow entries in TCAM memory, which is very slow for writing operations whereas OVS is using fast DRAM memory. The single flow installation delay in EZappliance is increasing when more flows have to be installed for less than 70 flows and then this value fluctuating between 40-50 msec per flow. In [1], the some measurements for NEC IP8800 device showed that a single flow installation delay is between 2.5 to 10 msec.

On the following chart it is presented single installation time into TCAM memory of EZappliance. The more flows are requested to be installed the longer delay in installation of single flow. It is because of abovementioned speed of TCAM memory but also it is related to implementation of adding new entries into search structure of NP-3 processor – it is implemented using Corba technology which causes delays between xDPd and EZproxy as it was presented in deliverable D3.3. Also modular prototype deployment of HAL modules on separate VMs causes additional delays.

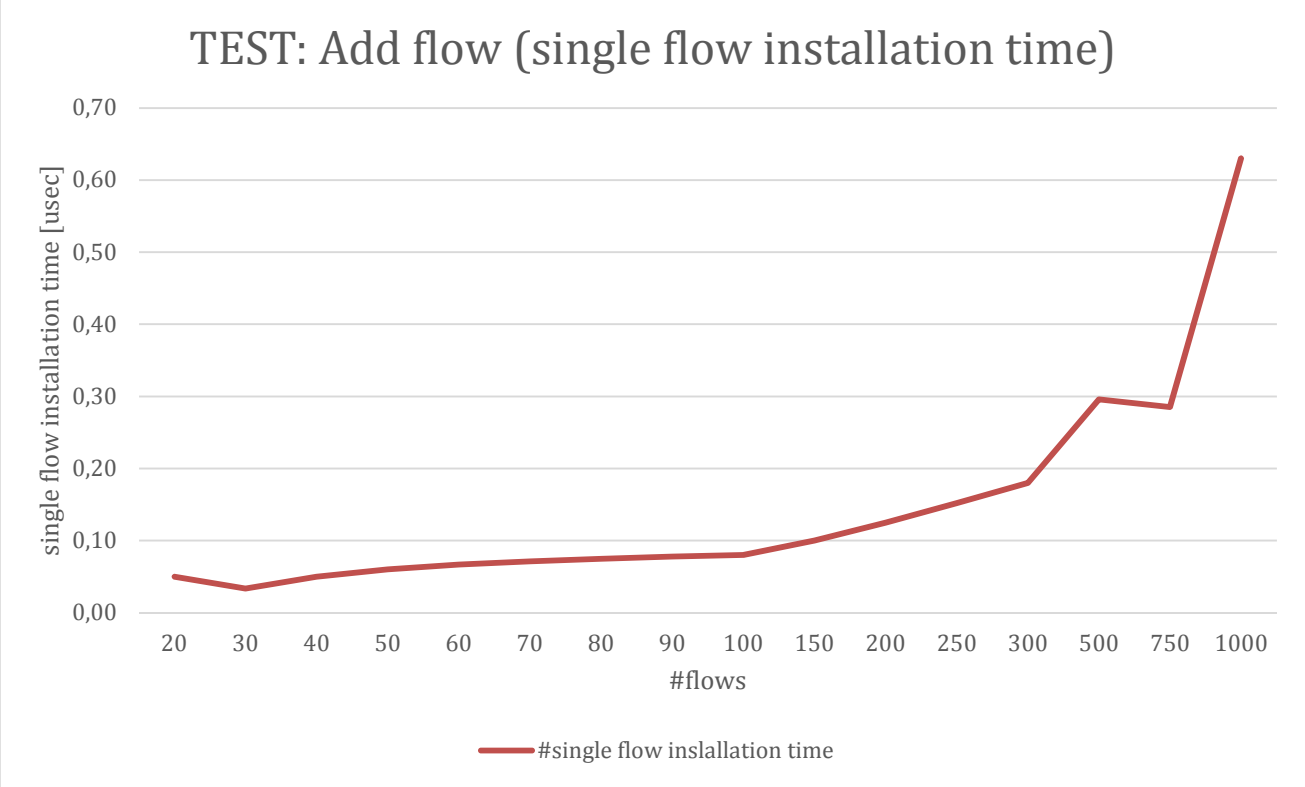


**Figure 1-46 Performance tests of EZappliance (Add flow – max number)**

| Add flow - max number | Result | Delay [sec] | Single flow installation time [sec] |
|---|---|---|---|
| 20 | success | 1 sec | 0,05 |
| 30 | success | 1 sec | 0,03 |
| 40 | success | 1 sec | 0,05 |
| 50 | success | 3 sec | 0,06 |
| 60 | success | 4 sec | 0,07 |
| 70 | success | 5 sec | 0,07 |
| 80 | success | 6 sec | 0,08 |
| 90 | success | 7 sec | 0,08 |
| 100 | success | 8 sec | 0,08 |
| 150 | success | 15 sec | 0,10 |
| 200 | success | 25 sec | 0,13 |
| 250 | success | 38 sec | 0,15 |
| 300 | success | 54 sec | 0,18 |
| 500 | success | 148 sec | 0,30 |
| 750 | success | 214 sec | 0,29 |
| 1000 | success | 630 sec | 0,63 |

**Table 1-12 Performance tests of EZappliance (Add flow – max number)**