Corigine

PERFORMANCE REPORT

Agilio CX 2x40GbE with OVS-TC

OVS-TC WITH AN AGILIO CX SMARTNIC CAN IMPROVE A SIMPLE L2 FORWARDING USE CASE AT LEAST 2X.

WHEN SCALED TOREAL LIFE USE CASES WITH COMPLEX RULES TUNNELING AND USING MORE FLOWS, THE CORIGINE SOLUTION PROVIDES AN AVERAGE IMPROVEMENT OF 16X OVER THE COMPETITION.

CONTENTS

1.0 LIST OF ACRONYMS

Acronym	Description
DPDK	Intel's Data Plane Developer Kit
Gb/s	Throughput unit: Gigabit per second
Mb/s	Throughput unit: Million bits per second
Mpps	Frame rate unit: Million packets per second
NIC	Network Interface Card
OVS	Open vSwitch
PF	PCIe Physical Function
PMD	DPDK Poll-Mode Driver
SR-IOV	PCIe Single Root Input/Output Virtualization
TC	Traffic Control
VF	PCIe Virtual Function
VM	Guest Virtual Machine
XVIO	Corigine's Express Virtio
VNF	Virtual Network Function

2.0 OVERVIEW

OVS is the most commonly used virtual switch datapath. Kernel-OVS is implemented as a match/action forwarding engine based on flows that are inserted, modified or removed by user space. OVS was further enhanced with another user space datapath based on DPDK. The addition of OVS-DPDK improved performance but also created some new challenges. OVS-DPDK bypasses the Linux kernel networking stack, requires third party modules and implements its own security model for user space access to networking hardware. DPDK applications are more difficult to configure optimally; while OVS-DPDK management solutions exist, debugging can become a challenge without access to all the tools generally available for the Linux kernel networking stack. As more and more networking engineers realize that DPDK is complex, there is a need for continuously improved solutions.

OVS using TC is the newest kernel-based approach, and improves upon Kernel-OVS and OVS-DPDK by providing a standard upstream interface for hardware acceleration. This report will discuss how an offloaded OVS-TC solution performs against software-based OVS-DPDK.

OVS-TC with an Agilio CX SmartNIC can improve a relatively simple Layer 2 (L2) forwarding use case at least 2X. When scaled to real life use cases with complex rules (see Chapter 6), tunneling and using more flows, the Corigine solution provides an average improvementof 16X over the competition.

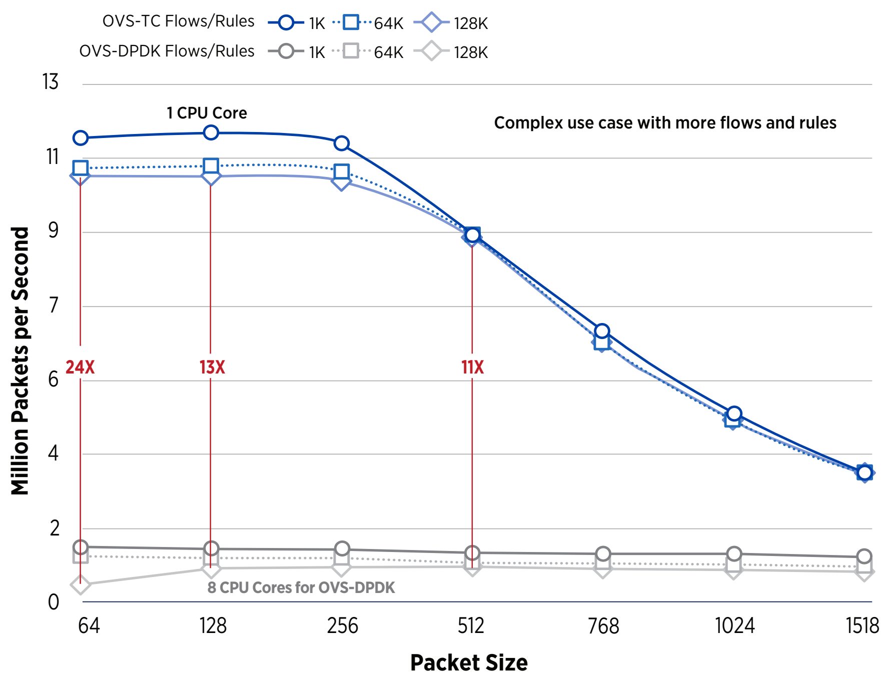


Figure 2.1: *1-port Mpps performance comparison | PHY-VM-PHY VXLAN | Agilio CX - Intel XL710 | 2x40GbE*

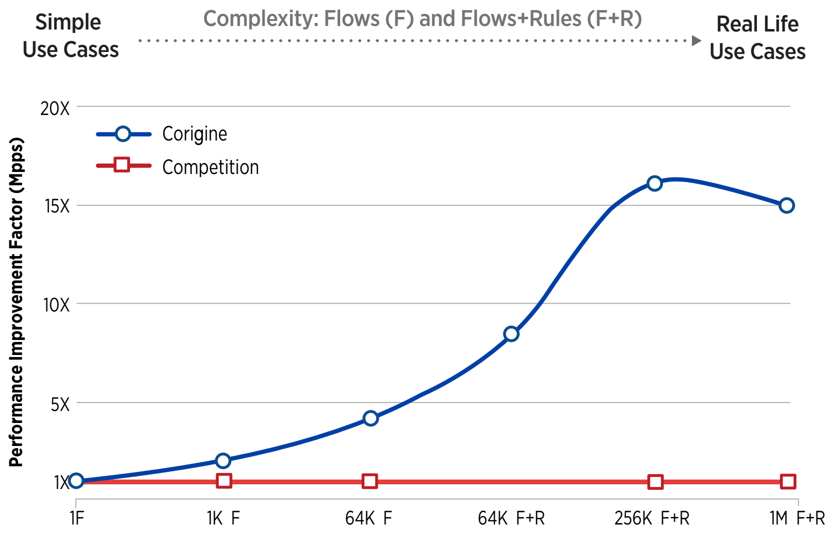


Figure 2.2: *Corigine performance advantage for average packet sizes between 64B and 512B*

3.0 AGILIO OVS-TC ARCHITECTURE OVERVIEW

The figure below depicts the OVS-TC data plane being offloaded to the Corigine SmartNIC.

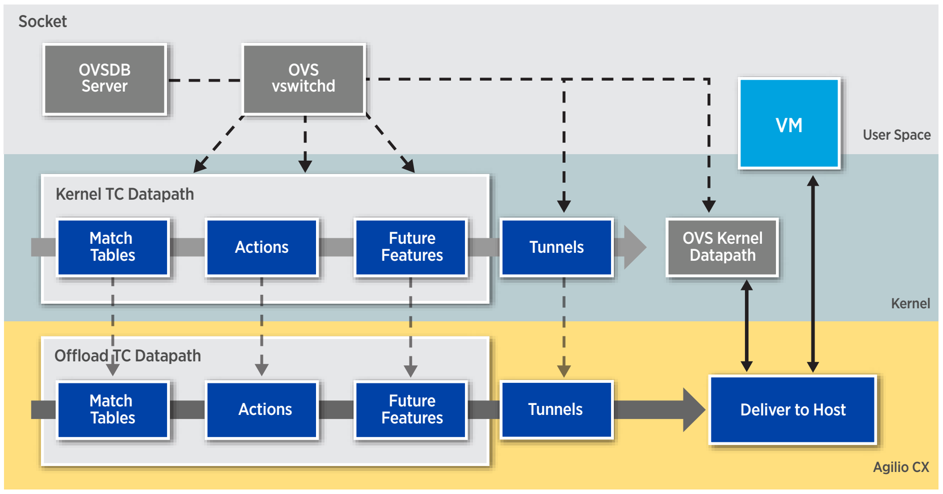


Figure 3.1: *Agilio OVS-TC architectural overview*

THE KEY GOAL OF THE BENCHMARKING IS TO QUANTIFY OVS DATAPATH PERFORMANCE AND THE ABILITY TO DELIVER DATA TO AND FROM HOST OR GUEST APPLICATIONS.

By running the OVS-TC data functions on the Agilio SmartNIC, the TC datapath is dramatically accelerated while leaving higher-level functionality and features under software control. Thus, retaining the leverage and benefits derived from a sizeable open source development community.

OVS contains a user space-based agent and a kernel-based datapath. The user space agent is responsible for switch configuration and flow table population. As observed in Figure 3.1, it is broken up into two fundamental components: ovs-vswitchd and ovsdb-server. The user space agent can accept configuration via a local command line (e.g. ovs-vsctl) as well as from a remote controller. When using a remote controller it is common to use OVSDB to interact with OVSDB-server for OVS configuration. The kernel TC datapath is where the Agilio offload hooks are inserted. With this solution, the OVS software still runs on the server, but the OVS-TC datapath match/action modules are synchronized down to the Agilio SmartNIC via hooks provided in the Linux kernel.

4.0 TESTING METHODOLOGY

Repeatability is essential when testing complex systems and this is often difficult to achieve.

Running performance tests and reviewing results are not trivial tasks as expert knowledge is required in multiple technology fields and host configurations, such as BIOS optimization, Linux host optimization, Kernel-based virtual machine (KVM)/Quick Eumulator (QEMU) optimization, CPU pinning and process affinity, OVS configuration, etc. The key goal of the benchmarking is to quantify OVS datapath performance and the ability to deliver data to and from host or guest applications. The tests are designed to make use of the DPDK to ensure the offloaded OVS datapath performance capabilities are evaluated and not the Linux IP networking stack – more details are provided in the *Infrastructure Specifications* section. Ixia was used as an external hardware traffic generator (IxNetwork 8.41) which can transmit and receive traffic at line rate. An example DPDK application, testpmd, was used as a VNF to forward traffic.

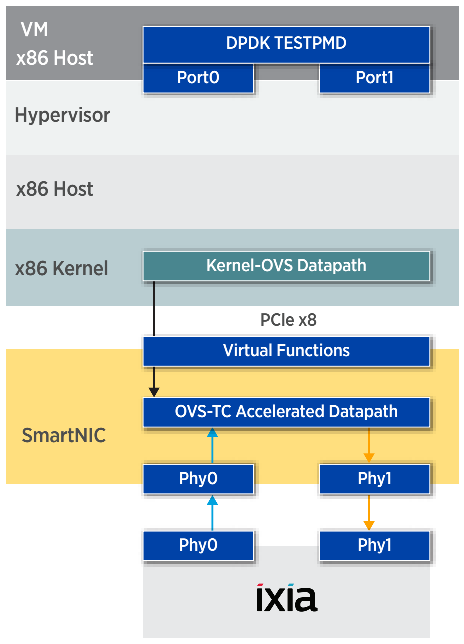


Figure 4.1.1: *PHY-OVS-PHY topology*

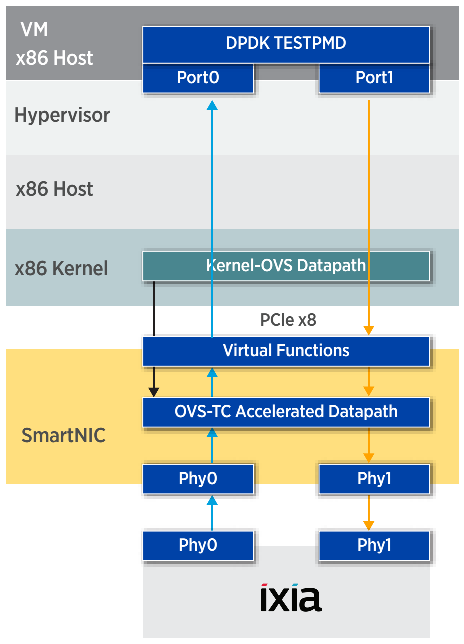


Figure 4.2.1: *PHY-VM-PHY topology*

4.1. PHY-OVS-PHY

The PHY-OVS-PHY benchmarking topology measures the capabilities of the SmartNIC to forward traffic between physical ports. An Ixia is used to generate and sync traffic for the PHY-OVS-PHY topology. OVS OpenFlow rules were setup to forward traffic between the two physical ports. *Figure 4.2.1. PHY-OVS-PHY topology* shows the data flow in a purely accelerated fashion.

4.2. PHY-VM-PHY

The PHY-VM-PHY benchmarking topology simulates the scenario of a VNF - incoming traffic is received by a guest application, which is then transmitted back over a different interface. In a production environment the VNF could perform routing, deep packet inspection, NAT, firewall, caching, etc. In the PHY-VM-PHY testing topology traffic is transmitted without any modification to dedicate resources to packet forwarding.

In this test case, the traffic crosses over the PCI bus and is forwarded in a guest using testpmd as the VNF - testpmd is an example application that is part of DPDK. The OVS OpenFlow rules are accelerated and offloaded to the Agilio SmartNIC. *Figure 4.1.1. PHY-VM-PHY topology* shows the data flow tested for the PHY-VM-PHY topology.

5.0 FLOW SCALABILITY

The flow scalability test implements simple in/out OVS OpenFlow rules and executes the same topology with different number of flow counts. For this report, 1K and 1M flow traffic profiles have been highlighted and used to demonstrate the difference in performance between a small number of flows and a larger number of flows.

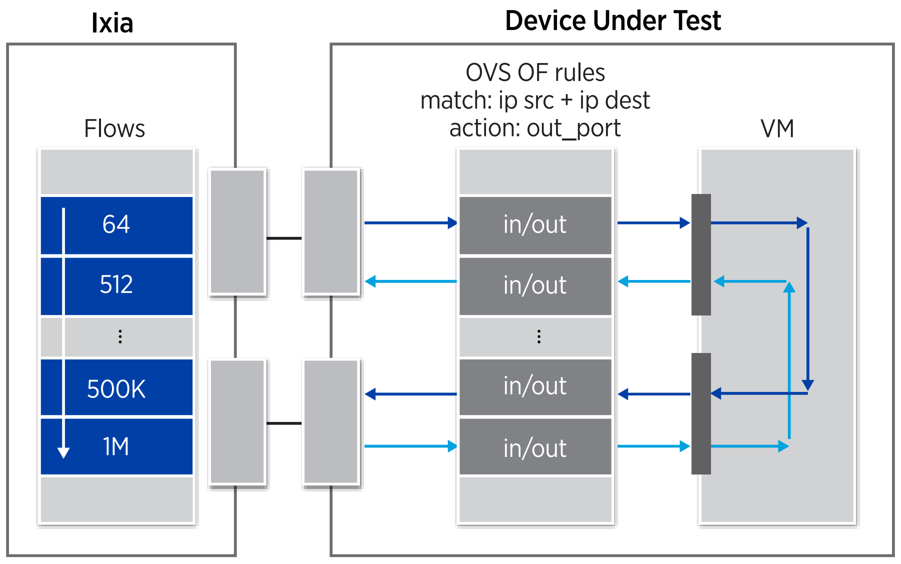
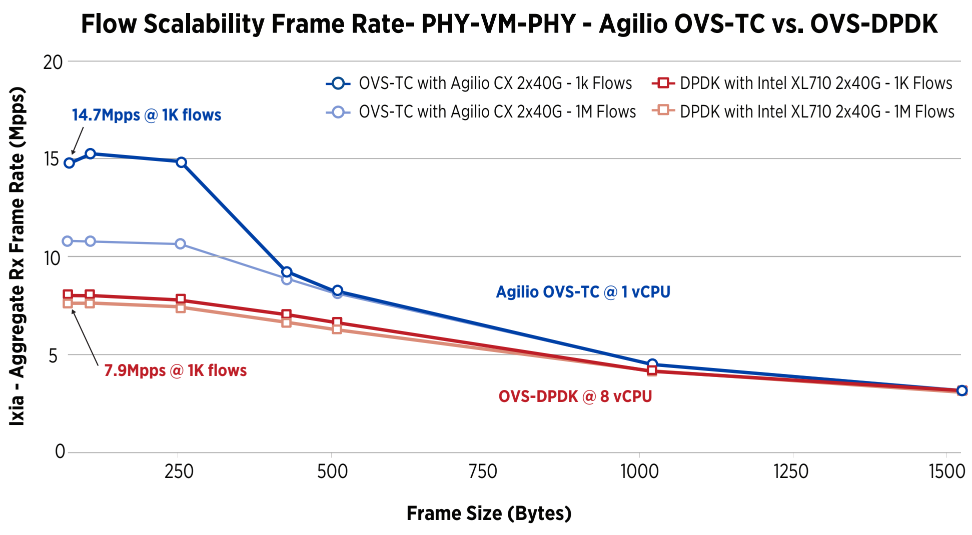


Figure 5: *Flow scalability test topology*

5.1. PHY-VM-PHY — GUEST.TESTPMD.VF

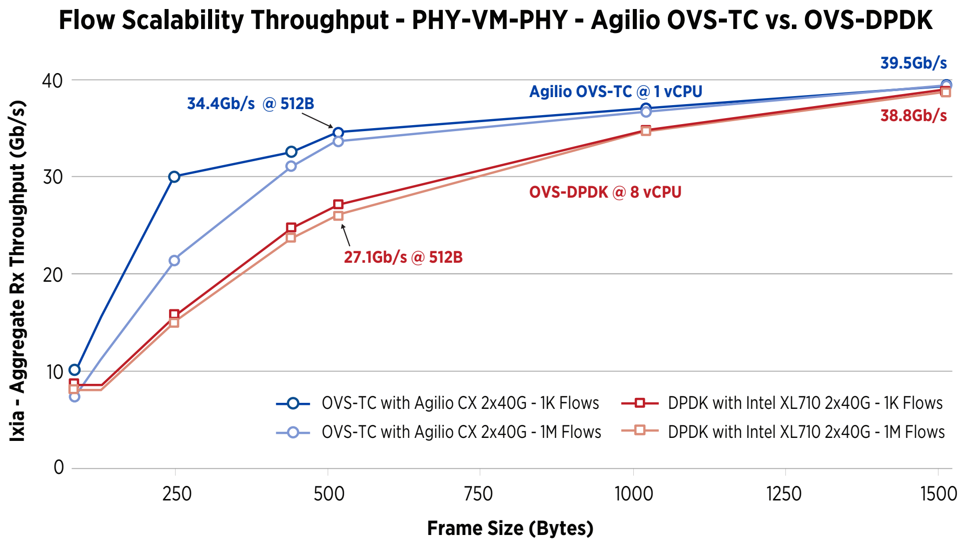
5.1.1. FRAME RATE



Graph 5.1.1: *Frame rate | PHY-VM-PHY | Agilio OVS-TC vs. OVS-DPDK (Flow scalability)*

Agilio OVS-TC achieves 85% higher frame rate than OVS-DPDK at 66B for 1,000 flows, providing a clear advantage for the Agilio CX 2x40GbE SmartNIC. OVS-TC utilizes one CPU core and delivers 14.7Mpps. Intel XL710 with OVS-DPDK uses eight CPU cores to deliver 7.9Mpps.

5.1.2. THROUGHPUT

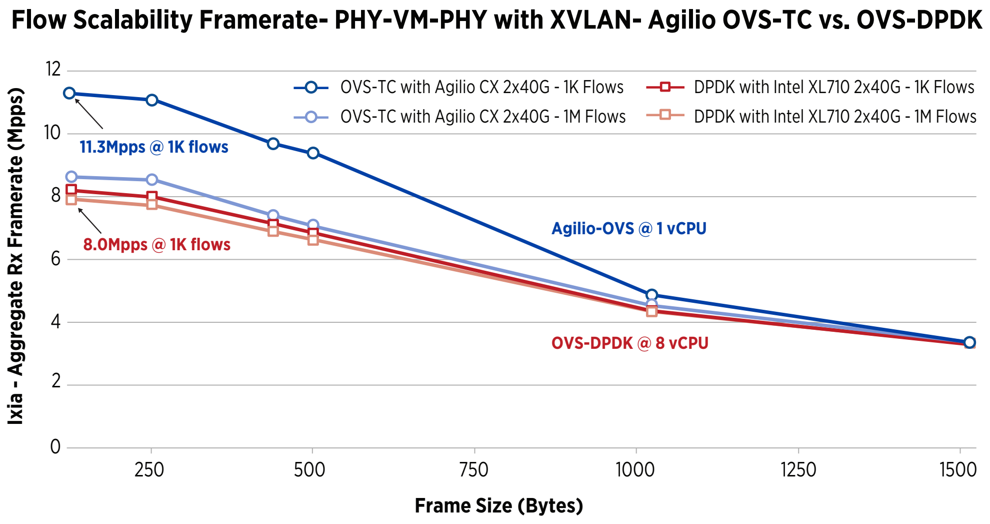


Graph 5.1.2: *Throughput PHY-VM-PHY | Agilio OVS-TC vs. OVS-DPDK (Flow scalability)*

Agilio OVS-TC outperforms OVS-DPDK for all frame sizes. The most substantial performance delta appears at 256B and 512B frame sizes. At 512B Agilio CX 2x40GbE provides 34.4Gb/s of throughput compared to 27.1Gb/s for the XL710 2x40GbE OVS-DPDK solution.

5.2. PHY-VM-PHY — VXLAN - GUEST.TESTPMD.VF

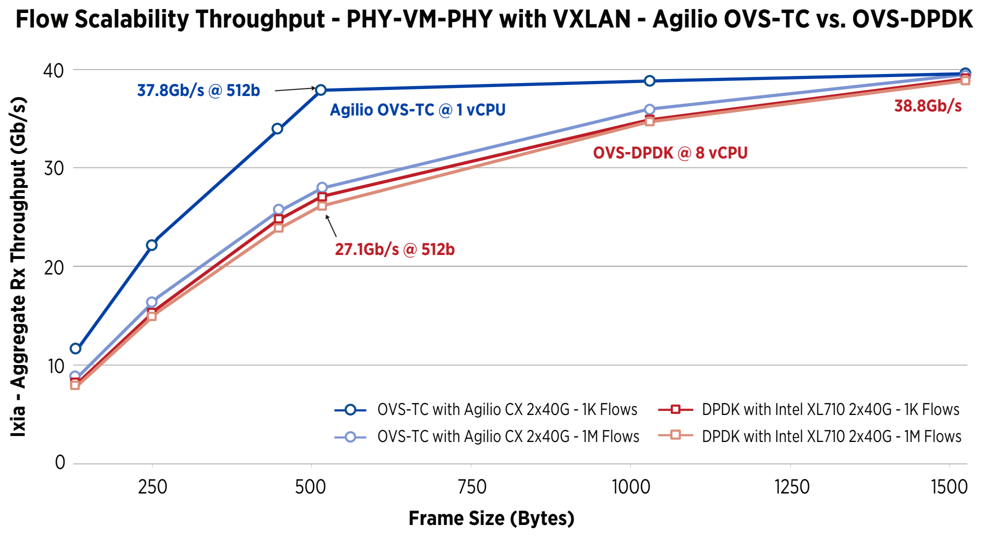
5.2.1. FRAME RATE



Graph 5.2.1: *Frame rate PHY-VM-PHY | VXLAN | Agilio OVS-TC vs. OVS-DPDK (Flow scalability)*

On this PHY-VM-PHY with VXLAN test, Agilio OVS-TC achieves 41.3% higher frame rate than OVS-DPDK for VXLAN traffic at 130B, and it maintains similar performance delta up to 512B of frame size. At 1,000 flows, the measurement was 11.3Mpps performance for Agilio CX 2x40G-bE and 8Mpps for XL710 at 64B.

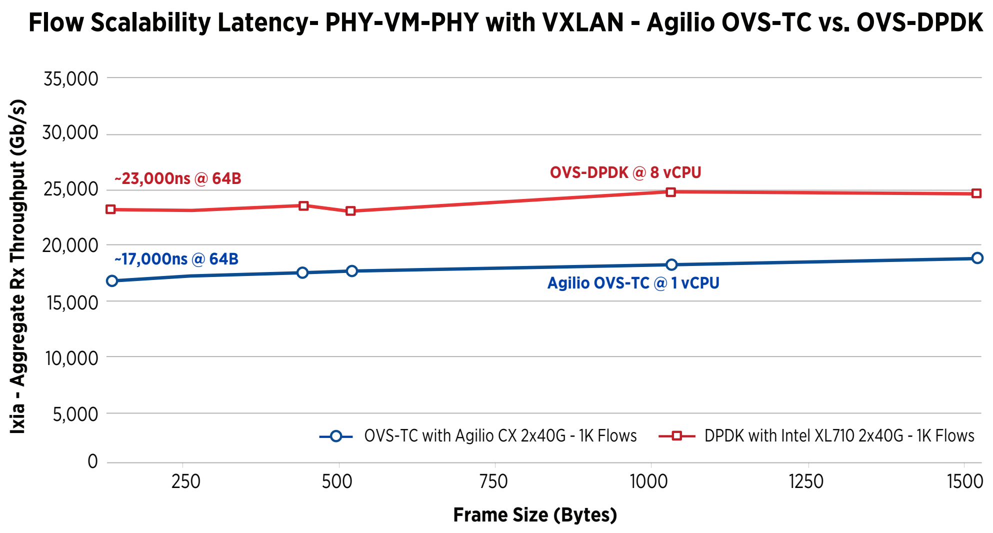
5.2.2. THROUGHPUT



Graph 5.2.2: *Throughput PHY-VM-PHY | VXLAN | Agilio OVS-TC vs. OVS-DPDK (Flow scalability)*

With this test, one observes a 37.8Gb/s of throughput with 1,000 flows on Agilio CX 2x40G-bE which translates to 95% of line rate for VXLAN traffic at 512B. At the same test instance, OVS-DPDK performance is below line rate at 27.1Gb/s.

5.2.3. LATENCY



Graph 5.2.3: *Latency PHY-VM-PHY | VXLAN | Agilio OVS-TC vs. OVS-DPDK (Flow scalability)*

To complete this test case the latency performance was measured for both solutions and Agilio OVS-TC achieves an average of 25% lower latency for VXLAN tunneling than OVS-DP-DK using only ⅛th of the CPU.

6.0 RULE COMPLEXITY

The rule complexity case is designed to scale the number of OVS OpenFlow rules (matching on source and destination IP addresses) and execute the same topology using the identical number of flows (1:1 rules/flows). This test allow us to observe granular performance characteristics of both solutions on a large number of flows and rules. Each installed rule gets a uniform distribution of traffic. For this report 64, 1K, 8K, 64K, 128K and 1M traffic flow profiles have been highlighted and is used to highlight the difference in performance between a small number of flows and a larger number of flows.

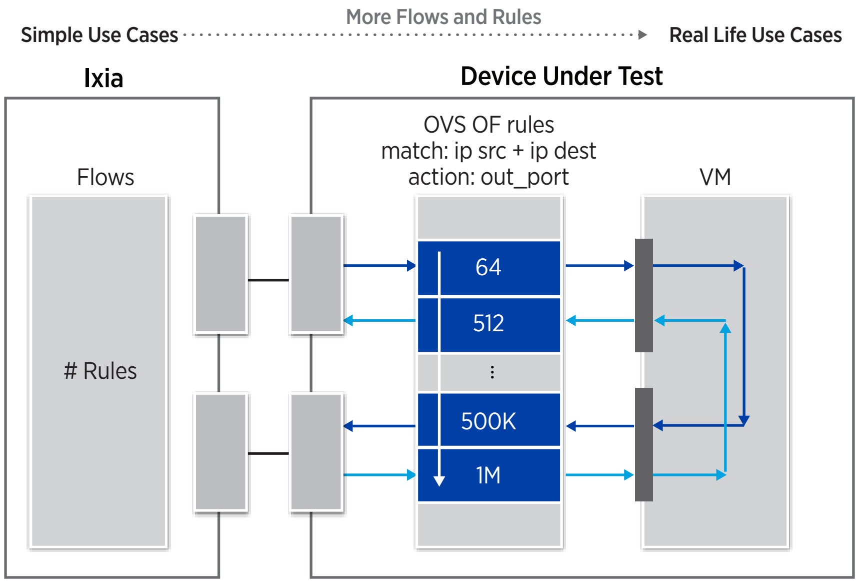
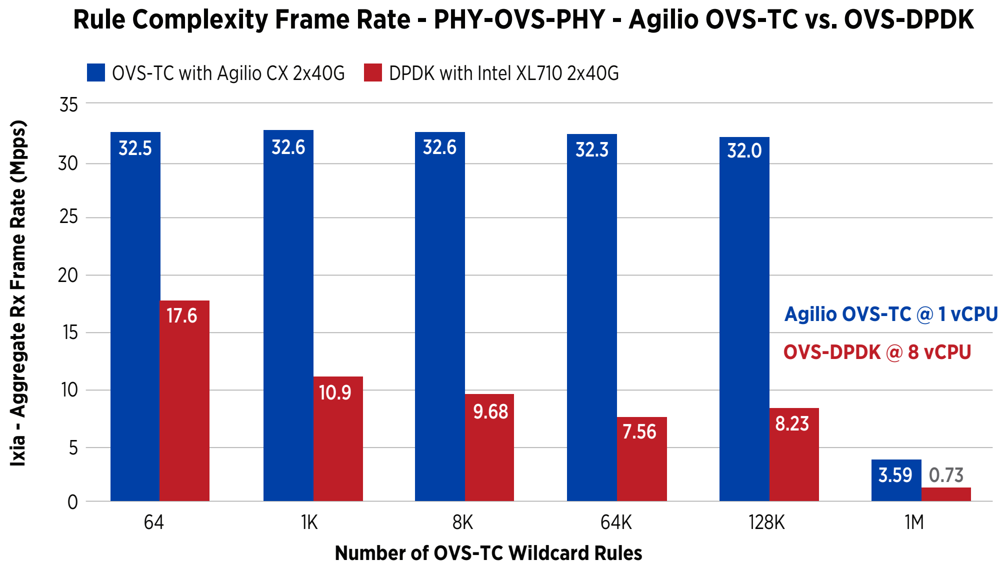


Figure 6: Rule complexity test topology

6.1 PHY-OVS-PHY

6.1.1. FRAME RATE @ 86B

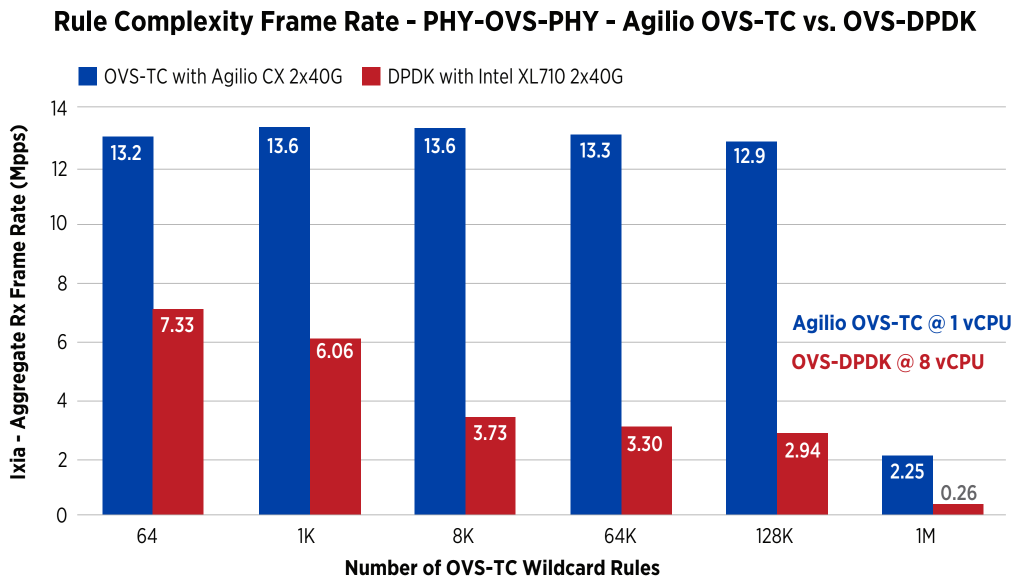


Graph 6.1.1: Frame rate @ 86B | PHY-OVS-PHY Agilio OVS-TC vs. OVS-DPDK (1:1 Flows/Rules)

The results of the standard OVS-DPDK test highlights that an increased number of flows/rules has a negative impact on the throughput. As the flows/rules increase, the frame rate drops. At 64B packet size, OVS-DPDK delivers 17.6Mpps on 64 flows/rules, but when scaled to 1M flows/rules the performance drops by 24X to approximately 0.73Mpps. As the number of flows and rules increase, the OVS-DPDK performance degrades dramatically.

6.2 PHY-VM-PHY — GUEST.TESTPMD.VF

6.2.1. FRAME RATE @ 86B

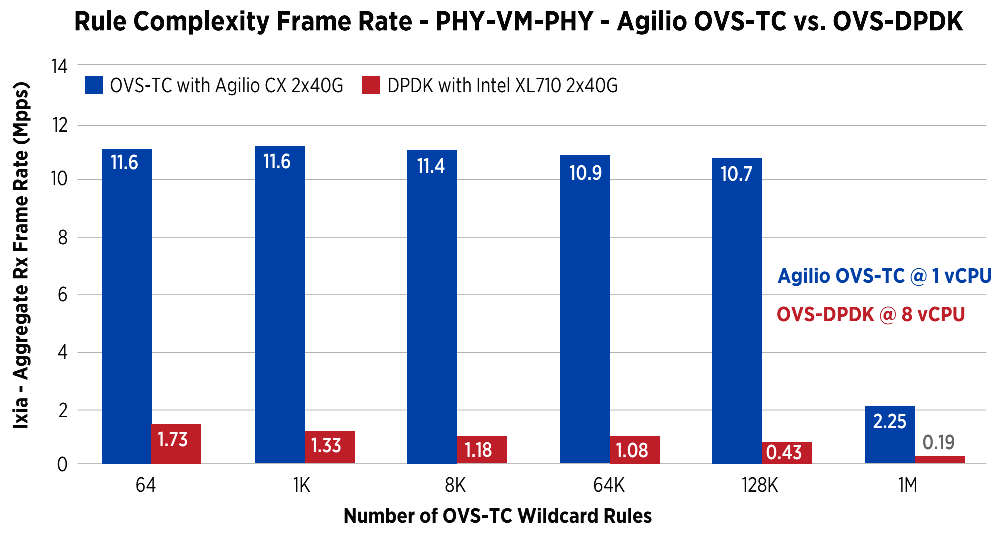


Graph 6.2.1: *Frame rate @ 86B | PHY-VM-PHY Agilio OVS-TC vs. OVS-DPDK (1:1 Flows/Rules)*

At a 86B packet size, OVS-TC delivers 13Mpps for 64K flows/rules while OVS-DPDK achieves 3.3Mpps. When compared to OVS-DPDK performance for this test, OVS-TC running on the Agilio SmartNIC performs an average of 5X better than the software only solution.

6.3 PHY-VM-PHY — VXLAN - GUEST.TESTPMD.VF

6.3.1. FRAME RATE @ 130B



Graph 6.3.1: *Frame rate @130B | PHY-VM-PHY | VXLAN | AOVS-TC vs. OVS-DPDK (1:1 Flows/Rules)*

On this PHY-VM-PHY with VXLAN test the average packet-per-second throughput of hardware-accelerated OVS-TC with one CPU core allocated is 10X higher than OVS-DPDK with eight CPU cores allocated.

7.0 SUMMARY

Agilio CX SmartNICs with OVS-TC significantly outperform OVS-DPDK with Intel NICs. Even when four server CPU cores with hyperthread partners are allocated to OVS-DPDK, the packets-per-second throughput of Agilio with OVS-TC delivers an up to 24X performance increase over OVS-DPDK.

OVS-TC leads the way in virtual switching at scale by accelerating the Linux networking stack. This benchmarking confirms that Agilio SmartNICs with OVS-TC can deliver more data to more applications running on a given server. This translates directly to improved server efficiency and a dramatic reduction in TCO, as fewer servers and less data center infrastructure (such as switches, racks and cabling) are needed to perform a given application workload.

8.0 APPENDIX

8.1. INFRASTRUCTURE SPECIFICATIONS

The benchmarking setup was performed using the following servers:

NIC	Hardware	Software
Agilio CX 2x40GbE	**Motherboard** Supermicro X10DRi **CPU** Intel(R) Xeon(R) CPU E5-2630 v4 40 CPUs @ 2.20GHz 2 NUMA nodes **Memory** 4x 16GB DIMM DDR4 @ 2400 MHz	**OS** Ubuntu 16.04.4 LTS (Xenial Xerus) **Kernel** 4.15.0-041500-generic **Virtualization (Qemu)** Compiled against library: libvirt 3.6.0 Using library: libvirt 3.6.0 Using API: QEMU 2.10.1 Running hypervisor: QEMU 2.10.1 **OVS** 2.8.1
Intel XL710 2x40GbE	**Motherboard** Supermicro X10DRi **CPU** Intel(R) Xeon(R) CPU E5-2630 v4 40 CPUs @ 2.20GHz 2 NUMA nodes **Memory** 4x 16GB DIMM DDR4 @ 2400 MHz	**OS** Ubuntu 16.04.4 LTS (Xenial Xerus) **Kernel** 4.15.0-041500-generic **Virtualization (Qemu)** Compiled against library: libvirt 3.6.0 Using library: libvirt 3.6.0 Using API: QEMU 2.10.1 Running hypervisor: QEMU 2.10.1 **OVS** OVS 2.8.1 DPDK enabled (v17.05.2) 8X Hyperthreaded CPUs pairs (4x physical CPUs) 4X RX queues per physical port 4X TX queues per physical port 2X RX queues per vhostuser port

Table 1: *Host machine specifications*

8.1.1. DPDK BUILD SPECIFICATIONS

DPDK app	Specifications
DPDK	Tag: v17.05.2
testpmd-dpdk	**Build variables:** DEFAULT_PKT_BURST = 64 RTE_TEST_RX_DESC_DEFAULT = 1024 RTE_TEST_TX_DESC_DEFAULT = 1024 **VNF variables:** 8X Hyperthreaded CPUs pairs (4x physical CPUs) 2x RX queues per virtual port 2x TX queues per virtual port

Table 2: *DPDK build parameters*

8.1.2. BOOT SETTINGS

- BIOS settings: NUMA enabled
  - Enhanced Intel SpeedStep technology disabled
  - C-state, P-state - disabled
  - Hyper-threading enabled
  - Intel Virtualization Technology for Directed I/O enabled
  - CPU Power and Performance Policy: Performance
  - Memory Power Optimization: Performance Optimized
- Host Boot settings:
  - Hugepage size = 2MB; No. of Hugepages = 16,384 (32GB total; 16GB per NUMA node). Isolated CPUs: 0-9, 20-29

8.1.3. OPERATING SYSTEM SETTINGS

- Linux OS Services Settings
  - Disable: NetworkManager, firewalld, iptables, irqbalance

Corigine

Email: sales@corigine.com
www.corigine.com.cn