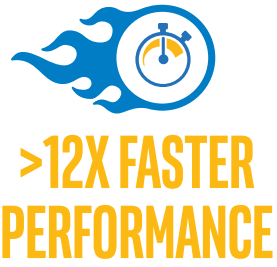
WHITE PAPER

Open vSwitch* Enables SDN and NFV Transformation

Intel's collaboration with the Open vSwitch community on DPDK boosts Open vSwitch performance by more than 12x to meet the demanding needs of SDN and NFV.



Executive Overview

Open source software (OSS) and industry open standards will continue to play a critical role in creating greater choice, faster time to market, and interoperability for software-defined networking (SDN) and Network Functions Virtualization (NFV) commercial solutions. For example, the Open vSwitch* (OvS*), which was first released in 2009, has evolved over the last six years to become a production-quality, multilayer virtual switch designed to enable network automation through standard management interfaces and protocols. Now universally available in the Linux* kernel since version 3.3 and widely used within the SDN and NFV developer communities, OvS is poised to play a key role as an open source reference point for the industry to adopt, measure, and test multilayer packet processing in a virtual switching environment in a number of ways:

- Packet processing performance
- Virtual switching functionality
- Interoperability
- Manageability

Until recently, it was widely believed within the networking industry and by Communication Service Providers (CSPs) that single root input/output virtualization (SR-IOV) and PCIe* pass-through were the only paths to achieve near bare metal line rate of networking workloads in virtual machines (VMs). SR-IOV technology allows simple, yet inflexible fixed configuration of the datapath between the VM and the physical I/O. For example, features such as network virtualization overlay need to be implemented in the hardware. In addition, the traffic steering capability of SR-IOV is limited by the hardware so, for example, if a 12-tuple lookup is required and the hardware does not support it then a vSwitch is required.

The openvswitch.org project recently included Data Plane Development Kit (DPDK) support as a user space option that helps accelerate datapaths across physical and virtual interfaces. Through improvements introduced in OvS with DPDK OSS, a significant boost in performance on Intel® architecture is made possible. Performance results of OvS with DPDK reach 40 Gbps and scale nearly linearly to higher throughputs. This is nearly a 12x improvement for aggregate switching between physical interfaces and a 7x improvement for switching between virtual functions. Increasing throughput is only one key metric for performance improvements. There is further optimization work ongoing within Intel and the community that is expected to deliver not only higher throughput, but also improved functionality and latency and jitter characteristics.

Table of Contents

While it may have previously been valid to conclude that a virtual switch is unable to meet the needs of CSP data plane workloads, recent advances in the use of DPDK coupled with virtual switch software provides a compelling solution that meets NFV goals and ensures flexibility and manageability in the future. This can be demonstrated with OvS as an open source solution, but there are also commercial offerings available to meet a variety of deployment scenarios.

Intel continues to work with the community and across industries to help accelerate SDN and NFV deployments. In addition to contributions to OvS with DPDK, Intel developers contribute to Open Platform for NFV* projects to advance open network platforms for NFV. The Intel® Open Network Platform (Intel® ONP) reference architecture is an example of such a platform designed to showcase the performance and functionality achievable through open platforms. OvS with DPDK is integral to the Intel ONP. Intel plans to continue to invest in OvS improvements to achieve improved performance, more throughput, lower latency and jitter, and additional functionality.

Introduction

With software-defined networking (SDN) and Network Functions Virtualization (NFV) gaining traction, CSPs, enterprises, and cloud providers are seeking solutions based on open source software (OSS) and off-the-shelf hardware platforms. Using OSS, standard high-volume servers (SHVS), and open platforms, those providers are offered greater choice, lower cost, and no single-source supplier lock-in, which is typical of dedicated hardware solutions. Additionally, high-performance software-based solutions enable faster delivery of flexible, innovative services using SDN and NFV to support the experiences users are looking for.

Intel is enabling network transformation by working on open source contributions to enable OSS to meet the demanding performance required in network virtualization. One of these focused endeavors is with Open vSwitch* (OvS*), a software switch layer used in virtualized environments. Intel's focus with OvS software has been to target bottlenecks in performance and enhance functionality to meet CSP, enterprise, and cloud needs by using the Data Plane Development Kit (DPDK) OSS libraries. OvS can be configured to support two software architectures, both discussed in this document:

- **Native OvS.** OvS with an in-kernel datapath
- **OvS with DPDK.** OvS compiled with the DPDK datapath

Performance and functionality requirements differ from one market to another. OvS provides adequate performance in a number of cloud and enterprise use cases, but the performance seen so far has not met the needs of many Telecom NFV use cases. By integrating the DPDK into OvS, virtual switching performance is accelerated across virtual and physical connections, creating a virtual switch layer targeting the performance required to meet Telecom and enterprise use case deployment models.

Native OvS and OvS with DPDK are critical components of open source-based NFV solutions on open platforms. Advancements to OvS software, test tools, and integration by Intel are made through direct contributions to the OvS project and through the Open Platform for NFV* (OPNFV*) community project.

Intel® Open Network Platform (Intel® ONP) is a reference architecture for SDN and NFV deployments that provides updates through quarterly public releases. The Intel ONP includes reference software that showcases the progress made in OvS with DPDK, offering service providers an accelerated time to solution delivery.

This paper describes OvS, the enhancements enabled with DPDK, the software performance gains achieved, and the work of Intel in OPNFV to continue advancing OvS with DPDK.

Virtual Switching and Open vSwitch*

This section explains what virtual switching is and provides details about OvS and OvS with DPDK.

The Software Virtual Switch

SDN and NFV are transforming how new products for data centers and networks are designed and built. A critical part of Network Functions Virtualization Infrastructure (NFVI) is the virtual switch, or vSwitch. A vSwitch is a software layer in a server. The server also hosts virtual machines (VMs) or containers with virtual Ethernet ports (vNICs). These ports connect to the vSwitch through a virtual interface (vIF), and the vSwitch routes traffic between the VMs, both resident on the same server and across the rack or between data centers. The vSwitch may serve as the ingress point into overlay networks running on top of physical networks. Figure 1 illustrates a typical physical and virtual switching infrastructure.

The vSwitch is used in enterprise, cloud, and Telecom use cases. For example, vSwitch enables forwarding traffic between VMs in a multitenant network virtualization setting, particularly with multiserver virtualization deployments. With network overlays, encapsulated packets are transferred across multiple vSwitches that forward packets on top of a physical network. In Service Function Chaining, the vSwitch's role is to switch packets between VMs while supporting dynamic service chaining configurations (see Figure 2 on the next page).

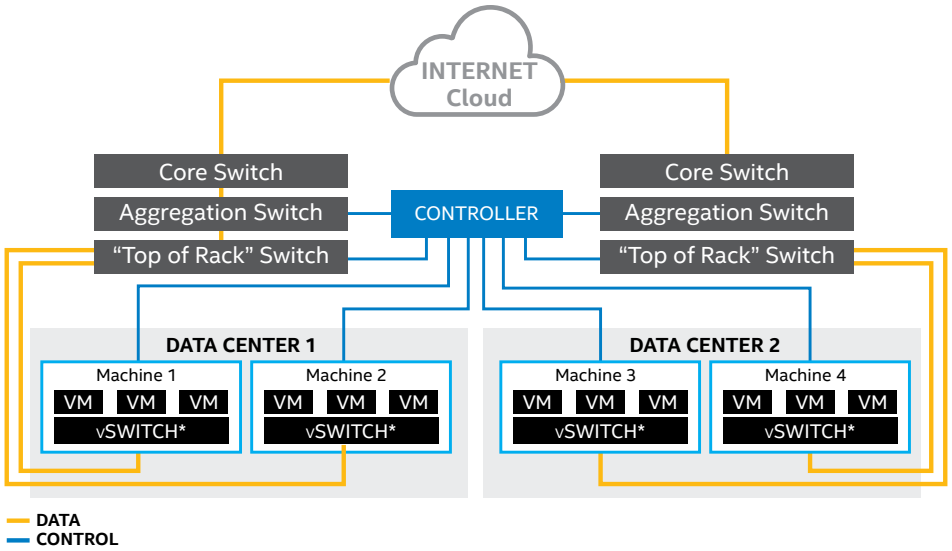


Figure 1. A typical service and switching infrastructure used in a data center.

The vSwitch plays a critical role in the NFV architecture. In particular, the vSwitch enables a flexible, robust, and time-performance-centric NFVI. Figure 3 illustrates the reference architecture defined by the European Telecommunications Standards Institute (ETSI) for NFV deployment (ETSI NFV Reference Architecture).

The ETSI NFV group has specified many use cases for NFV deployment, which cover both telecommunication and data communication domains. Examples of use cases include the following:

- **Virtual 4G Enhanced Packet Core (vEPC).** Hosting a 4G EPC framework that enables secure mobile services in a virtual environment.
- **Virtual Customer Premise Equipment (vCPE) solutions.** Moving high-level services away from the customer site and provisioning them in the virtualized cloud.
- **Virtual Provider Edge (vPE) solutions.** Completely virtualizing the provider edge router functionalities using both a monolithic model and a service chaining model.

In all these use cases, if the deployment model assumes an infrastructure as shown in Figure 1, the role of the vSwitch is as follows:

- Complete isolation of the underlying hardware infrastructure for network I/O for the virtualized network functions (VNFs) that are hosted, enabling greater flexibility for the VNF in the event of its migration.
- Perform as a high-throughput logical layer-2 switch for traffic between the VNFs that are hosted (either on the same platform or on a remote platform).
- Perform overlay termination, origination, and other intelligent functions for delivering high-throughput network traffic to and from the VNFs.

Today, a wide range of software-based switches is available, both open source and proprietary. Open source vSwitch solutions include Open vSwitch, Snabb Switch*, and Lagopus*. Commercial offerings include VMware ESXi*, Wind River Titanium Edition*, and 6Wind's 6WINDGate*.

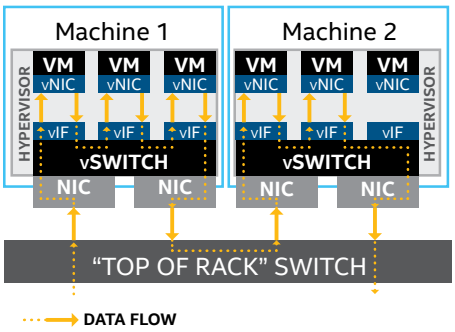


Figure 2. The vSwitch as part of a virtualized environment. Example for illustration purposes only.

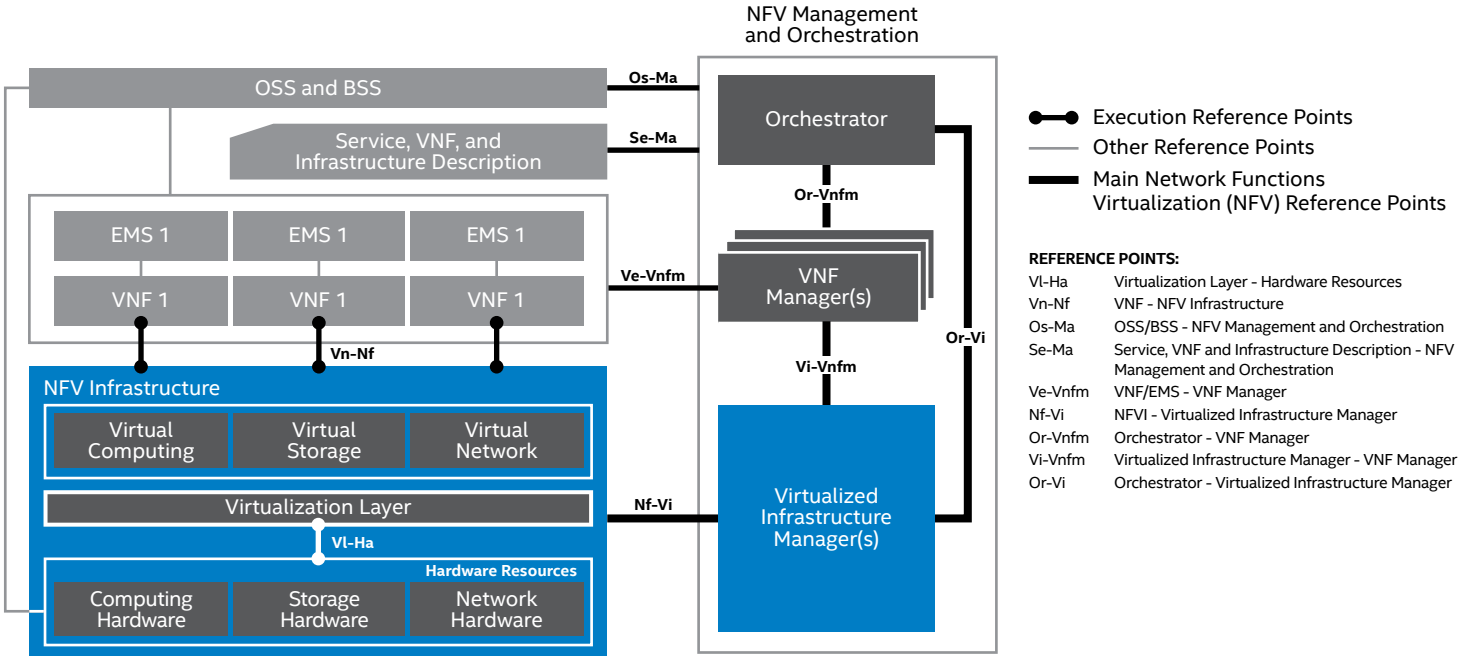


Figure 3. European Telecommunications Standards Institute (ETSI) NFV reference architecture.

Open vSwitch

OvS is an open source vSwitch software stack project that can run as a virtual switch in virtual environments, provide switching to host-based applications, and run as the control stack of hardware switches. OvS is offered under the Apache 2.0 license.

Among the available virtual switches, OvS stands out as one that is widely known and understood and, with the enhancements in progress, can meet a wide range of deployment needs. OvS plays a vital role in several SDN/NFV open source projects, including OpenStack*, OpenNebula*, and OpenDaylight*. OvS is the vSwitch that is most widely deployed in OpenStack installations, according to the OpenStack.org 2014 survey.[1] OvS has been ported into multiple commercial virtualization platforms and added to switching chipsets. The kernel datapath software is included in the Linux* kernel, making it widely available with Linux distributions. It is largely deployed in cloud enterprise networks and is also being evaluated by leading service providers and CSPs as the open source vSwitch of choice for NFV. For these reasons, Intel has adopted OvS as the vSwitch in its Intel ONP reference architecture in order to illustrate data plane performance on a multilayer, open source vSwitch within an open platform.

Open vSwitch Architecture

The OvS software is designed for Linux-based hypervisors and intended to address the open source community's needs for feature-rich virtual switching capabilities. These capabilities include L2/L3 forwarding, VxLAN, Access Control Lists (ACLs), Quality of Service (QoS) policy and traffic shaping, and monitoring and configuration. OvS enables these features using standard network protocols, such as NetFlow*, IPFIX*, SPAN*, and sFlow*. The OvS Database Management protocol (OVSDB), an OpenFlow* configuration protocol, enables automated control and remote monitoring triggers.

As mentioned earlier, there are two salient configurations of OvS software: native OvS (OvS with the datapath in the kernel) and OvS with DPDK (OvS with the datapath in the user space). Both architectures are discussed in this document.

Native Open vSwitch Architecture

The native OvS architecture (see Figure 4) comprises two principle software components: a kernel space fast path (openvswitch.ko) and a user space daemon (ovs-vswitchd). The main purpose of the kernel space fast path is to process received frames based on a lookup table of recently used flow-table entries that ovs-vswitchd programs.

The native OvS architecture's main elements include the following:

- **openvswitch.ko**. This is the fast path through OvS (openvswitch.ko) and consists of the lookup table, with each entry made up of a number of match fields and actions.
  - Match fields define a set of fields in the packet's headers that can be used to identify the type of packet received.
  - Actions define what can be done with the packet, such as push a VLAN tag, modify the packet header, and output to a port.
- **ovs-vswitchd**. This is a set of several major libraries:
  - **ofproto.** Implements an OpenFlow switch.
  - **netdev.** Abstracts the interaction with network devices (physical and virtual).
  - **dpif.** Abstracts a simple single-table forwarding path.

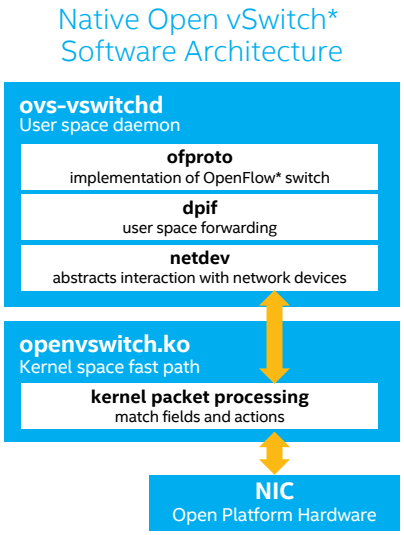


Figure 4. Native Open vSwitch* software architecture.

When a frame is received, the fast path in the kernel space uses match fields to determine which flow table entry to match on and, correspondingly, which set of actions to execute. If the frame does not match an entry in the lookup table, it is sent to ovs-vswitchd in the user space, which requires a costly context switch. The user space component then determines how to handle frames of this type before sending it back to the kernel space component, again with a costly context switch. ovs-vswitchd also instructs openvswitch.ko how to handle future frames of this type by caching a flow table entry in the fast path lookup table in the kernel.

While OvS performs reasonably well as a vSwitch, there is a need and an opportunity for performance enhancements and added capabilities useful for NFV. Thus, Intel has contributed to the open source community and the OvS project to develop and enhance OvS with DPDK, boosting software performance across several fronts to meet today's NFV demands.

Open vSwitch Architecture with DPDK

The DPDK is a set of data plane libraries and network interface controller drivers that enable high-throughput and low-latency packet processing. The openvswitch.org project recently included DPDK support as a compile-time option for the user space fast path that helps accelerate datapaths across physical and virtual interfaces.

DPDK uses Poll Mode Drivers (PMDs) for user space packet processing. It also uses advanced Intel® instruction sets, among other techniques, to meet the throughput and latency requirements that virtualized network functions need in today's demanding services. PMDs allow OvS to poll physical network interfaces, and other virtualized or non-virtualized device types. This avoids costly interrupt-driven network device models, while allowing OvS to take advantage of advanced Intel instruction sets, such as SIMD (single instruction, multiple data) instructions.

Figure 5 shows the OvS with DPDK architectures and highlights the areas on which Intel's contributions focus. The following list explains the various components shown in the figure:

- **dpif-netdev** is the user space implementation of the fast path. It implements the dpif API using netdev devices. Implementing all packet processing in the user space eliminates the overhead associated with legacy kernel space code.
- **ofproto-dpif** implements the ofproto API using the dpif layer.
- **netdev-dpdk** enables a variety of DPDK vSwitch ports, implementing the netdev API using DPDK libraries to implement each type of interface.
  - **Physical ports (PMDs)** are implemented using vfio (or igb_uio).
  - **dpdkvhostuser and dpdkvhostcuse** ports are implemented using librte_vhost. These ports allow a user to add vHost User and vHost Cuse ports to the user space datapath. These ports enable fast communication with a VM by implementing a VirtIO user space vHost provider. VirtIO/vHost is a fast, secure, and standard interface for communicating with VMs.
  - **dpdkr** interfaces are implemented using librte_ring. These ports allow a user to add DPDK ring ports to the user space datapath. This enables fast zero-copy communication with a VM using IVSHMEM or another process running on the host OS.

All of these architectural changes to OvS enable a wide variety of VNFs. OvS with DPDK allows service providers to achieve significant performance improvements on Intel® architecture when switching traffic from a physical network to a virtual network and between VMs on a virtual network. Intel has benchmarked the performance achievable from OvS with DPDK, as discussed in the next section.

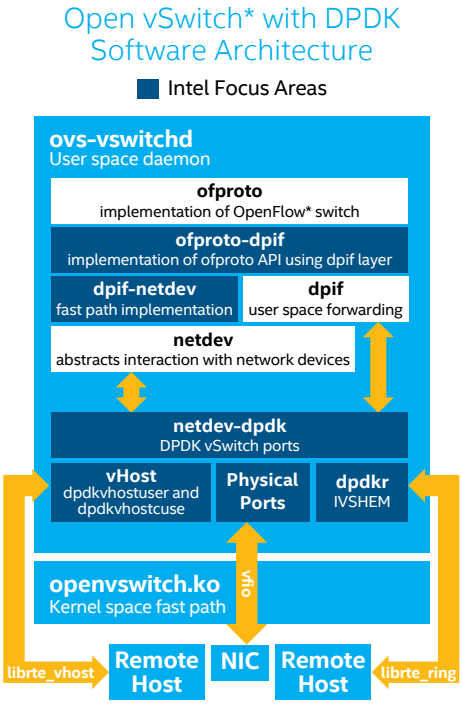


Figure 5. Open vSwitch* with Data Plane Development Kit (DPDK) software architecture.

Open vSwitch with DPDK Performance Benchmarks

Intel has compared OvS to DPDK through both physical-to-physical and physical-to-virtual-to-physical switching tests to produce benchmarks. These tests can easily be duplicated for testing in the reader's own environment.

The benchmarks achieved significant performance improvements with DPDK. As seen if Figure 6, physical-to-physical performance tests with DPDK ran nearly 12x faster with DPDK than without DPDK, illustrating the aggregate switching performance through OvS with DPDK. As seen if Figure 7, physical-to-virtual-to-physical switching through a VM with OvS resulted in over a 7x improvement with DPDK over native OvS. Both tests were done with 256-byte packets to focus on the performance for small packet processing by OvS.

Table 1 shows the test platform setup for both physical-to-physical and physical-to-virtual-to-physical tests.

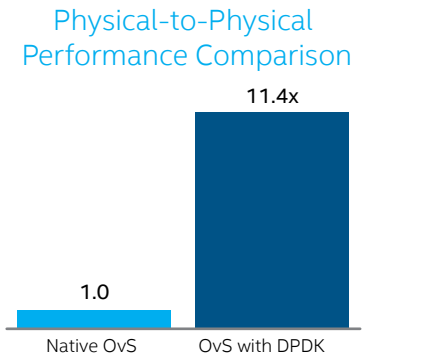


Figure 6. Physical-to-physical test results show that Open vSwitch* (OvS) with Data Plane Development Kit (DPDK) is almost 12x faster than native OvS. Intel internal measurements, July 2015.

Table 1. Test Platform Setup

HARDWARE COMPONENT	
Platform	Intel® WorkStation Board W2600CR
Processors	2x Intel® Xeon® processors E5-2680 v2 @ 2.80 GHz
Memory	8x 8 GB DDR3-1867 MHz DIMM
NICs	2x Intel® 82599 10 Gigabit Ethernet Controller
BIOS	Revision: 04/19/2014 SE5C600.86B.02.03.0003.041920141333 • Intel® Virtualization Technology enabled. • Hyper-threading enabled/disabled as indicated by test. • Intel SpeedStep® technology disabled. • Intel® Turbo Boost Technology disabled. • CPU power and performance policy: Performance.
SOFTWARE COMPONENT	
Host OS	Fedora* 21 x86_64 3.17.4-301.fc21.x86_64
Virtual machine	Fedora 21 x86_64 3.17.4-301.fc21.x86_64
Virtualization technology	QEMU-kvm v2.2.0
DPDK	DPDK 2.0.0
vSwitch	Open vSwitch* v2.4.0 (pre-release) `git commit 44dbb3e4bd085588a1ebc70d9a25d2ed6b63e18b`
IXIA	IxNetwork 7.40.929.15 EA
gcc	gcc (GCC) 4.9.2 20141101 (Red Hat 4.9.2-1)
CONFIGURATION	
CPU isolation	All cores isolated except core 0
Core affinitization	Standard Open vSwitch: QEMU*, vHost processes core affinitized OvS with DPDK: PMDs core affinitized through OvS pmd-cpu-mask
DPDK	vhost-user disabled
vSwitch	`ovs-vswitchd --dpdk -c 0x8 -n 4 --socket-mem 1024,0 -- unix:/usr/local/var/run/openvswitch/db.sock –pidfile`
QEMU	`VHost=on, mrg_rxbuff=off`

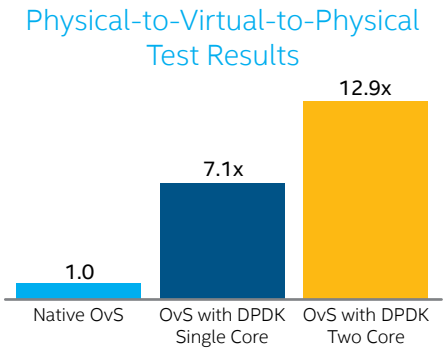


Figure 7. Physical-to-virtual-to-physical test results show that Open vSwitch* (OvS) with Data Plane Development Kit (DPDK) performs 7x faster than native OvS. Intel internal measurements, July 2015.

Physical-to-Physical Performance

We provide physical-to-physical results as a reference baseline for comparing performance. Physical-to-physical testing is uncomplicated but indicates the aggregate switching performance through a simple OvS software datapath. It uses two physical 10 Gbps NIC ports with the data routed bidirectionally through OvS (first in one direction and then in the other).

Figure 8, on the left, shows the steps for the packet path in one direction (it is identical for the opposite direction).

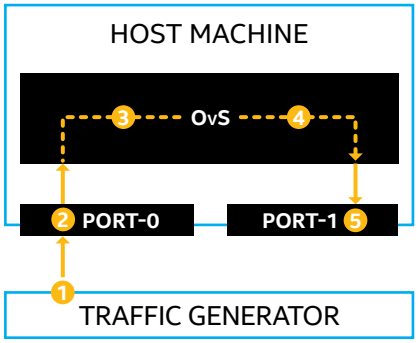
As shown in Figure 6, OvS with DPDK compared to native OvS delivered an almost 12x speedup with a single PMD thread. This test was performed with 256-byte packets with zero packet loss. These results are important, because they show typical packet performance through OvS. Table 1 provides full test setup and configuration details.

Physical-To-Virtual-To-Physical Performance

Loopback testing through a VM shows the performance for virtual switches and VMs. The test uses two 10 Gbps NIC ports with data switched through virtual ports in OvS to a VM and back again. Figure 8, on the right, shows the test setup for the packet path in one direction (it is identical for the opposite direction).

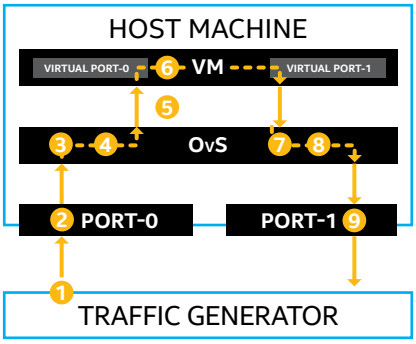
As shown in Figure 7, OvS with DPDK provides over a 7x speedup. Note again that this test uses typical 256-byte packets with zero packet loss. Table 1 provides the full test setup and configuration details.

Physical-to-Physical Test Setup



1. The match data is initially loaded into the OvS with DPDK fast path.
2. The packet arrives in physical port 0.
3. The packet is received directly into the OvS user space through the DPDK PMD for the NIC, bypassing the Linux* kernel.
4. A lookup is made to match the packet with an action.
5. The action in this case is to forward the packet to physical port 1. The packet is sent directly from the OvS user space to physical port 1 using the DPDK PMD.

Physical-to-Virtual-to-Physical Test Setup



1. The match data is initially loaded into the OvS flow tables.
2. The packet arrives in physical port 0.
3. The packet is received directly into the OvS user space using the DPDK PMD for the NIC.
4. A lookup is made to match the packet with an action.
5. The action in this case is to forward the packet to virtual port 0. The packet is sent to the VM through the DPDK vHost library.
6. The packet is received in the VM and forwarded back to the host using the DPDK VirtIO PMD.
7. The packet is received in the vSwitch from virtual port 1 using the DPDK vHost library.
8. A lookup is made to match the packet with an action.
9. The action in this case is to forward the packet to physical port 1. The packet is sent directly from the OvS user space to physical port 1 using the DPDK PMD.

Figure 8. The steps for the physical-to-physical (left) and physical-to-virtual-to-physical (right) test setup describe the packet path in one direction (it is identical in the opposite direction).

Open vSwitch with DPDK, SR-IOV, and PCIe Pass-Through: A Continuum of Solutions

Problem Statement

CSPs have demanding data plane performance needs. Some have concluded that the only way to achieve their goals is to bypass any software in the NFVI and pass through direct hardware access using mechanisms such as SR-IOV or PCIe pass-through. However, this approach does not satisfy many of the goals of NFV, as described in the ETSI NFV Architectural Framework, including key ones, such as standardized interfaces and flexibility in assigning VMs to different infrastructure.

The NFVI provides the hardware and software upon which the VNFs run (see Figure 10).

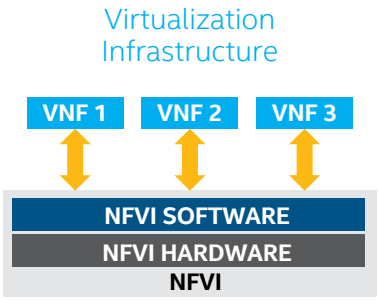


Figure 10. Network Functions Virtualization Infrastructure (NFVI).

SR-IOV and PCIe Pass-Through Approach

SR-IOV is a standard that makes a single PCI (peripheral component interconnect) hardware device appear as multiple virtual PCI devices. The standard is defined by the PCI Special Interest Group*, and it is supported on a variety of Ethernet controllers.

Unlike in the various vSwitch proposals, where a vIF is used between the VNF and NFVI, SR-IOV bypasses the NFVI software and exposes the hardware directly to the VNF.

The driver for the hardware thus resides in the VM, rather than the NFVI. The virtual functions (VFs) have drivers that are often a subset of the Physical Function (PF) drivers, and thus not all features may be available in the VF. For example, the controller might be set up in promiscuous mode, which for security reasons may not be enabled in a VF driver. There are other constraints as well, such as configuration parameters that could potentially affect all VFs.

PCIe pass-through is a capability that allows the PF PCIe device to be allocated in its entirety to the VM. The VM must have the relevant device drivers for the hardware PF. Intel® Virtualization Technology for Directed I/O (Intel® VT-d) supports PCIe pass-through. Note that many of the same characteristics (advantages and disadvantages) of SR-IOV also apply to PCIe pass-through.

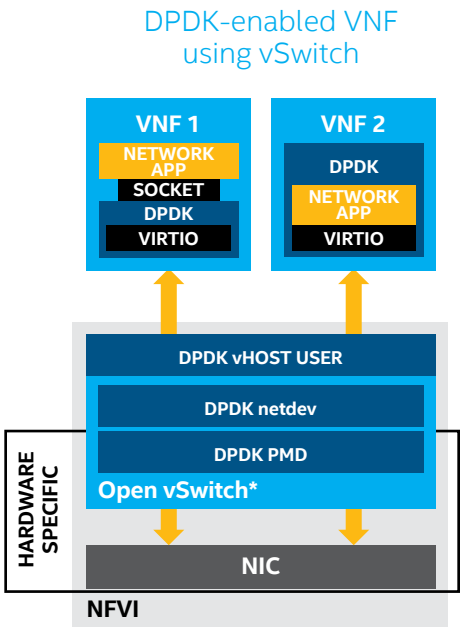


Figure 11. DPDK-enabled Virtual Network Functions (VNF) using vSwitch.

VNF Interface Options and Constraints

Figures 11 and 12 show a high-level comparison between a vSwitch approach and the SR-IOV approach. In the vSwitch approach (using OvS as an example), the hardware is abstracted by the DPDK PMD in the NFVI. The interface, as seen by the VNF, uses the widely supported VirtIO interface. In contrast, in the SR-IOV model the VFs of the NIC are exposed directly to the VM. In this case the VF PMD resides within the VM itself.

Table 2, on the following page, lists the advantages and disadvantages of both approaches. Note that PCIe pass-through is equivalent to the model shown for VNF 3 and VNF 4, with the VF replaced by a PF.

Another factor to consider is how the traffic is steered toward the VNF. In the case of a vSwitch, this can be done in a variety of ways, such as by the 12-tuple defined by OpenFlow rules. Steering traffic with SR-IOV, however, depends on the capabilities of the specific hardware. For example, some hardware may only support steering traffic by the VLAN or possibly the outer IP (Internet protocol) header. Other hardware may allow a more flexible pipeline, although the number of flows supported will be less than is possible in a software solution.

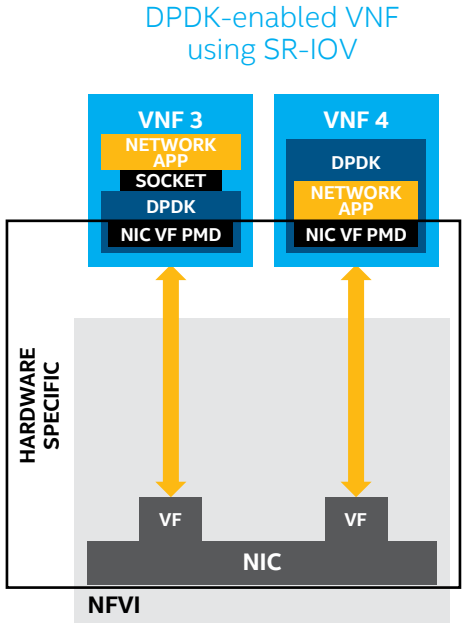


Figure 12. DPDK-enabled Virtual Network Functions (VNF) using SR-IOV.

Table 2. Advantages and Disadvantages of Virtual Switch, SR-IOV, and PCIe* Pass-Through Solutions

REQUIREMENT	SR-IOV	PCIe* PASS-THROUGH	NATIVE OvS	OvS WITH DPDK
Maximum throughput to an individual VM	Near native throughput. I/O delivered directly to the VM	Same as SR-IOV	< 1M pps	>12x compared to native vSwitch approach
Maximum throughput from VM to VM	Limited by NIC internal switching and PCIe bandwidth	Limited by PCIe bandwidth and ToR switch	< 1M pps	Scalable with additional cores for switching
Latency/Jitter	Lowest	Same as SR-IOV	Not a focus requirement	Tuning in progress
Network virtualization (programmability/SDN)	Overlay networks possible with NIC or ToR switch support	Overlay networks possible with ToR switch support	Core of enterprise and cloud network virtualization and SDN	Continued feature growth
VM live migration	VM tied to specific hardware port (workaround available)	Same as SR-IOV	Key value of vSwitch	Link bonding, vHost migration
VM abstracted from hardware	VM will need to include specific driver	Same as SR-IOV	Key value of vSwitch	Yes when using standard I/O API (VirtIO); if using DPDK, requires a driver on the guest VM
VNF longevity	Additional validation and certification with I/O hardware updates	Same as SR-IOV	Same binary will run across generations of servers (x86)	Same binary will run across generations of servers (x86)
Network monitoring	Will need to copy additional VM or be extracted from I/O hardware	Same as SR-IOV	Available in vSwitch	Will eventually be available in vSwitch
Hardware pool utilization	Possible underutilization due to VM being limited to specific hardware mapping	Same as SR-IOV	Full flexibility for VNF assignment	Flexible under orchestration configuration
Industry maturity and adoption	Used by the Telecom industry	Same as SR-IOV	Appropriate for limited Telecom workloads that require low packet processing	Telecom is waiting for it; Intel is working with suppliers to enable adoption

DPDK – Data Plane Development Kit; OvS – Open vSwitch*; pps – packets per second; SDN – software-defined networking; SR-IOV – Single Root I/O Virtualization; ToR – top-of-rack; VNF – virtualized network functions; VM –Virtual Machine

Open vSwitch, SR-IOV, and PCIe Pass-Through Conclusions

While it may have been valid in the past to conclude that a vSwitch is unable to meet the needs of CSP data plane workloads, and thus SR-IOV or PCIe pass-through were required, recent advances in the use of DPDK, coupled with vSwitch software, provides a compelling solution that meets the NFV goals and ensures flexibility in the future. This capability can be demonstrated with OvS as an open source solution, but there are also commercial offerings available to meet a variety of deployment scenarios.

Potential Future Enhancements

As has been shown, there are significant performance gains when using OvS with DPDK. There are possibilities for further improvements using DPDK techniques, such as bulk processing, prefetching, and advanced Intel instruction sets. There are also possibilities to directly offload some of the processing to highly optimized DPDK libraries, as well as take advantage of new DPDK features (for example, multi-queue vHost User).

Intel continues to explore such techniques and to work on OvS with the community in collaboration with the OPNFV project in order to enhance OvS performance and capabilities even further. Intel's goals for OvS are to help make it the open source vSwitch of choice for SDN and NFV.

OPNFV is an open source project focused on accelerating NFV's evolution through an integrated, open platform. As an open source project, OPNFV is uniquely positioned to bring together the work of standards bodies, open source communities and commercial suppliers to deliver a de facto standard open source NFV platform for the industry.

www.opnfv.org

OPNFV: A Continuing Commitment to Open Platforms

Successful deployments of NFV services and appliances require platforms with performance and capabilities to support the different virtualized functions. The Intel ONP reference architecture is one such design, on which Intel has based much of its work and evaluation for OvS with DPDK. The industry has joined the collaborative OPNFV project. Intel is a Platinum member of the project, helping to advance the development of open platforms.

In addition to its work with OvS, Intel is leading other vSwitch projects in OPNFV.org as part of its continuing commitment to advance SDN and NFV on open platforms. See the Intel's vSwitch Projects sidebar.

Features of Open vSwitch 2.4 Release Enabled by Intel

Intel has enabled many features that are part of OvS release 2.4. Intel is focused on enhancing the functionality of the OvS user space to meet CSP, cloud, and enterprise requirements. Contributions to date include DPDK support, vHost User, IVSHMEM, and more. The following provides an overview of the new capabilities introduced in OvS release 2.4.

Data Plane Development Kit (DPDK) Support

OvS supports DPDK versions up to 2.0. Allows OvS to execute using the DPDK 2.0 library.

vHost Cuse

User space vHost enables fast packet transfer between host and guest. This is achieved by offloading the servicing of VirtIO-net devices to a user space device, using KVM*/QEMU*. This offloading reduces context switching and packet copying involved in traditional VirtIO-net, which in turn provides a significant performance boost for VM communications (compared to traditional methods).

There are two implementations of vHost, both of which have been enabled in OvS: vHost Cuse, where VirtIO-net devices are offloaded to a character device, and vHost User, where VirtIO-net devices are offloaded to a socket server device.

The vHost Cuse feature adds support for a new port type to the user space datapath called dpdkvhost. This allows KVM/QEMU to offload the servicing of VirtIO-net devices to its associated dpdkvhost port.

vHost User

vHost User is the standard QEMU-adopted user space vHost (VirtIO is the front end in the VM). This feature adds support for a new port type to the user space datapath called dpdkvhostuser. It adds to the existing infrastructure of vhost-cuse; however, it disables vhost-cuse ports as the default port type in favor of vhost-user ports. vhost-cuse 'dpdkvhost' ports are still available; they can be enabled using a configure flag.

A new dpdkvhostuser port will create a Unix* domain socket, which, when provided to QEMU, is used to facilitate communication between the VirtIO-net device on the VM and the OvS port on the host.

Intel's vSwitch Projects

Intel is a leader of several vSwitch projects in OPNFV.org and is committed to the advancement of SDN and NFV on open platforms. These projects include the following.

Characterizing vSwitch Performance for Telecom NFV Use Cases
This project develops a generic and architecture-agnostic vSwitch testing framework with associated tests that serve as a basis for validating the suitability of different vSwitch implementations in a Telecom NFV deployment environment.

The project implements a modular test framework, combining traffic generator, vSwitch, virtualized network functions, and characterization test cases. It generates a performance benchmark report for any of the supported test cases. Each of the modules (for example, the traffic generator) is pluggable: a test setup using any traffic generator can be supported within this framework.

Open vSwitch for NFV
This project proposes modifying the Open Platform for NFV* (OPNFV*) build to include a deployment option for the OvS build with a software-accelerated user space design. Changes will significantly improve the performance of the Network Functions Virtualization Infrastructure for network I/O.

Future work will encompass collaborative development within the OvS project to add functionality and improve the performance of the OvS version with a software-accelerated user space fast path and increase its suitability for Telecom NFV deployments.

For more information, see the **OPNFV Wiki.**

OpenDaylight*/OpenStack* Detection of DPDK Ports

This feature enables an ovsdb client (for example, OpenDaylight or OpenStack) to query the datapath to determine whether certain datapath and port types exist. In particular, an ovsdb client can determine whether an instance of ovs-vswitchd was compiled with DPDK support.

User Space Link Bonding

DPDK link bonding implements active/backup mode bonding. This enables transparent failover between a pair of redundant Ethernet links. If the active link fails, traffic is transferred to the backup link, and the previously active link is designated as the backup. Active/backup mode bonding is only one of several link bonding modes that are supported.

IVSHMEM

This feature enables a different method for sending packets to the VM than vHost, which copies between separate memory spaces. Shared memory transfer is a more efficient transfer method, because it is copying only packet pointers and not packet data. However, it lacks security, since the VM can see all the packet buffer memory space at all times. The system user can determine the relative importance of efficiency or security, using knowledge about whether the system is a closed environment.

vHost Performance Improvements

A number of user space vHost performance improvements were made in the OvS 2.4 time frame, increasing the throughput to the VM by approximately 3x, including the following:

- vHost packet batch size changed to 32.
- Retries added in the enqueue to vHost.

Datapath Performance Improvements

A number of datapath performance improvements were made in the OvS 2.4 time frame, increasing physical-to-physical throughput, including the following:

- Rx vectorization was added in the OvS PMD I/O path.
- Miniflow fix.
- The EMC table size increased from 1K to 8K.

Open vSwitch 2.4 Features Enabled by Intel

- Data Plane Development Kit (DPDK) Support
- vHost User
- OpenDaylight*/OpenStack* Detection of DPDK Ports
- User Space Link Bonding
- IVSHMEM
- vHost Performance Improvements
- Datapath Performance Improvements

Conclusion

Working with the open source community, Intel has significantly contributed to the enhancement of OvS performance to meet the business needs of enterprise and CSPs implementing SDN and NFV in their infrastructures. By adding DPDK to OvS, switching performance increased nearly 12x in physical-to-physical testing and 7x when switched through a VM. This is a significant leap forward for advancing OvS deployments, especially for applications where small packet performance is paramount.

OvS with DPDK is integral to the Intel ONP reference architecture, showcasing the work Intel has done with OvS and highlighting the value of using open network platforms for SDN/NFV solutions.

Intel remains committed to investments in open source contributions for SDN and NFV and to collaborating with the community, such as with the OPNFV project, to help drive deployments of software-defined networks and NFV.

To find out more about the OPNFV project, visit the **OPNFV website**.

For more information about Open vSwitch, visit the **Open vSwitch website**.

For more information about the Intel Open Network Platform Reference Architecture, visit the **Intel® Open Network Platform website**.

[1] "OpenStack User Survey Insights: November 2014." http://superuser.openstack.org/articles/openstack-user-survey-insights-november-2014

Printed in USA 1015/LBEC/KC/PDF ♻ Please Recycle 333441-001