IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

*International Journal of Advanced Research in Computer and Communication Engineering*
*Vol. 5, Issue 2, February 2016*

# A Review: Integration of Open vSwitch on the OpenDataPlane Framework

**N. D. Kakade[1], S. G. Tamhankar[2]**

Department of Information Technology, Walchand College of Engineering, Sangli, MS, India[1]

Department of Electronics Engineering, Walchand College of Engineering, Sangli, MS, India[2]

**Abstract:** Expansion of data traffic in communication devices requisite advanced and pioneered software provisions to manage bandwidth, performance, and power essentials, as well as scalable systems management and availability solutions. At present there are various type of SoC's in the market and these SoC's provides advanced and advanced and pioneered hardware provisions to some usual networking obstacles, permitting packet processing up to 100Gb/s speeds. To make full use of these capabilities of hardware there is a requirement of software, such that it can use HW potential in well organized manner. Therefore discrete vendors supplies their own software development kits (SDK's) to design the data plane applications. Because of these SDK's it is hard for applications to be entirely portable across platforms. Still in recent past there has not been an open software development kit (SDK) available for SoC's designers and individual software vendors (ISVs) which can issue APIs for the data plane of different SoC architectures. But now there is, developing rapidly through an industry organization, OpenDataPlane™ (ODP) [1] is a standardized data plane API that can be used to aid Linux-based network applications over the array of silicon architectures and hardware/software configurations. This cross platform characteristic is used by Open vSwitch (OVS) [3] to accelerate it on different architectures. This paper explores ODP and its characteristics including aid to OVS, and also ODP's relation to the Intel Data Plane Development Kit (DPDK) [5].

**Keywords:** OpenDataPlane, Open virtual Switch, Software Development Kit, Xen Project Hypervisor, System on Chips.

## 1. INTRODUCTION

In areas as diverse as architecture, engineering, and manufacturing, industry standards have demonstrated to be a very good thing. On the authority of Institute of Electrical and Electronics Engineers (IEEE), standards have allowed faster innovation as diverse products and technologies can interoperate. Ecosystems of innovators large and small can more easily and rapidly create new productions based on industry standards that assist billions of consumers and number of organizations in markets around the world.

However, until recently Networking silicon vendors had no similar data plane standards designed by the consensus of a critical mass of those vendors, and therefore original equipment manufacturers (OEMs) and individual software vendors (ISVs) have had to deal with a broad range of disparate, incompatible interfaces to hardware (e.g., HyperExec, NetOS, Simple Executive, DPDK). This has validate inefficient and it considerably restrict developer innovation and customer options.

Silicon vendors want any application to be able to run on their platform without the application designers needing to master the complexity of the platform. With open standards, silicon vendors can give optimized implementations of open APIs that can be leveraged across all applications running on the platform. Thus, the platform can suitable for any socket without having to overcome customer application migration obstacles. Instead, an application can be easily recompiled to execute straight away.

With a standard set of data plane APIs, silicon developers can address a much wider market and deliver customers more varied SoC's.

ISVs want software interoperability so they do not have to write dissimilar versions of code for distinct platforms to do hardware acceleration, service chaining between distinct chips, and other tasks.

Application developers want to be able to shift applications easily between platforms while conveniently taking advantage of the hardware offload capabilities of each. Open data plane standards offer application developers the widest range of target platforms at various price/performance points to optimize the market reach of the application.

In the computer graphics industry, software developers once had to write custom interfaces and drivers for each hardware platform. There was no common standard for writing software for graphics hardware until the development of the Open Graphics Library (OpenGL) standard.

Originally developed by Silicon Graphics in 1991 and released in 1992, OpenGL is a multi-platform API. It interfaces with a graphics-processing unit (GPU) to enable hardware-accelerated rendering.

The time has come for ODP, a common, open data plane programming interface across diverse silicon architectures (Figure 1).

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

*International Journal of Advanced Research in Computer and Communication Engineering*
*Vol. 5, Issue 2, February 2016*

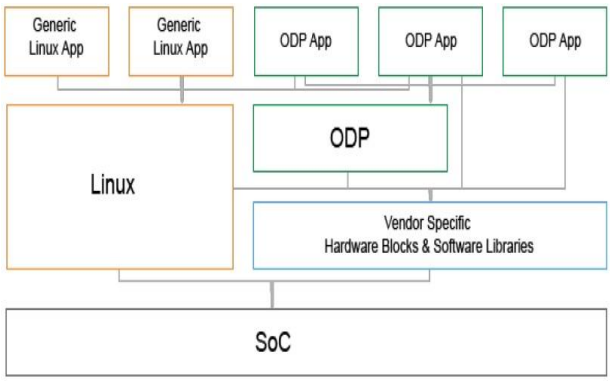


*Figure 1: OpenDataPlane Standards for SoC's*

OpenDataPlane (ODP) is an open source project that gives an application programming atmosphere for data plane applications that is simple in use, have immense performance and mobile across networking SoC's of different instruction sets and architectures. The atmosphere includes common APIs, configuration files, services, and utilities on top of an implementation optimized for the underlying hardware. The goal of ODP is to create a truly cross platform framework for data plane applications. OpenDataPlane (ODP) is a standardized data plane API that can be used to aid Linux-based network applications across the number of silicon architectures and hardware/software configurations.

The Xen Project hypervisor [4] is an open-source bare metal or type 1 hypervisor, which makes it possible to run number of instances of an operating system or in actual diverse operating systems simultaneously on a individual machine (or host). At present in the market Xen Project hypervisor is the only type-1 hypervisor which is available as open source. It is utilize as the basis for various commercial and open source applications, for example Infrastructure as a Service (IaaS), server virtualization, security applications, embedded and hardware appliances, desktop virtualization.

Open vSwitch is a software switch which operates at multiple layers and it is licensed under the open source apache 2 licenses. Its goal is to implement a production quality switch platform that aids standard management interfaces and opens the advancing functions to programmatic extension and control.

Hypervisors would like the power to bridge traffic between VMs and with the skin world. On Linux-based hypervisors, this wont to mean mistreatment the inbuilt L2 switch (the Linux bridge), that is quick and reliable. So, it's reasonable to raise why Open vSwitch is employed.

The answer is that Open vSwitch is targeted at multi-server virtualization deployments, a landscape that the previous stack is not compatible. These environments are typically characterized by extremely dynamic end-points, the upkeep of logical abstractions, and (sometimes) integration with or offloading to special purpose change hardware.

The following characteristics and design issues facilitate Open vSwitch deal with the higher than necessities.

• The mobility of state: All network state related to a network entity (say a virtual machine) ought to be simply recognizable and migratable between completely different hosts.
Open vSwitch has aid configuring as well as transferring both slow (configuration) and quick network state between instances. For instance, if a VM migrates between end-hosts, it is doable to not solely migrate associated configuration (SPAN rules, ACLs, QoS) however any live network state (including, for instance, existing state which can be tough to reconstruct). Further, Open vSwitch state is typed and backed by a true data-model leaving the development of structured automation systems.

• Acknowledging to network dynamics: Virtual atmospheres are typically outlined by high-rates of amendment. VMs returning and going, VMs proceeding backwards and forwards within time, changes to the logical network atmospheres, then forth.
Open vSwitch supports variety of options that enable a network system to retort and adapt because the atmosphere changes. This includes easy accounting and visibility support like NetFlow, IPFIX, and sFlow. However maybe additional helpful, Open vSwitch supports a network state information (OVSDB) that supports remote triggers. Therefore, a chunk of orchestration software system can watch numerous aspects of the network and respond if/when they modify. This is used heavily these days, for instance, to reply to and track VM migrations.
Open vSwitch additionally supports OpenFlow as a technique of exportation remote access to manage traffic. There are variety of uses for this.

• Preservation of logical tags: Distributed virtual switches (for example VMware vDS and Cisco's Nexus 1000V) typically maintain logical context inside the network through either appending or changing tags in network packets. This could be used to unambiguously determine a VM or to carry another context that is solely relevant within the logical domain.
Open vSwitch includes multiple strategies for specifying and maintaining tagging rules, all of that are accessible to an overseas method for orchestration. Further, in several cases these tagging rules are hold on in an optimized kind so that they have not got to be including a heavyweight network device. this permits, for instance, thousands of tagging or address remapping rules to be organized, modified, and transferred.
In a similar layer, Open vSwitch aids a GRE implementation that may handle thousands of concurrent GRE tunnels and aids remote configuration for tunnel creation, configuration, and tear-down. This, for instance, can be used to connect non-public VM networks in numerous information centers.

• Hardware integration: Open vSwitch's forwarding path is developed to be amenable to offloading packet processing to hardware chipsets, whether or not housed in a classic hardware switch chassis or in an end-host NIC. This permits for the Open vSwitch management path to be ready to each management a pure software implementation or a hardware switch.


DOI 10.17148/IJARCCE.2016.52118

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

*International Journal of Advanced Research in Computer and Communication Engineering*
*Vol. 5, Issue 2, February 2016*

There are several in progress efforts to port Open vSwitch to hardware chipsets. These embody multiple merchandiser silicon chipsets (Broadcom and Marvell), furthermore as variety of vendor-specific platforms.
The advantage of hardware integration is not solely performance inside virtualized atmospheres. If physical switches also expose the Open vSwitch management abstractions, each bare-metal and virtualized hosting environments will be managed using a similar mechanism for machine-driven network management.

In many ways, Open vSwitch targets a distinct purpose within the design space than previous hypervisor networking stacks, specializing in the requirement for machine-driven and dynamic network management in large-scale Linux-based virtualization environments.

By using a feature of ODP to support array of SoC's, Open vSwitch has been supplemented with an access layer to ODP this allows OVS to pick acceleration on supported ODP platforms without change.

Figure 2 below demonstrates possible ODP deployments on a multicore SoC with virtual environment. The deployment has Linux and bare metal in separate virtual machines, and three ODP applications running, the first two in separate (sets of) Linux processes in user space and a third one outside Linux in bare metal environment which lies in different VM. Linux user space supports direct hardware access from ODP applications (through a SoC specific SDK). ODP is designed to coexist with all standard Virtual Machine Monitors (VMMs) and virtualization hardware to enable data plane applications to run as virtualized containers in support of initiatives like Network Functions Virtualization (NFV).

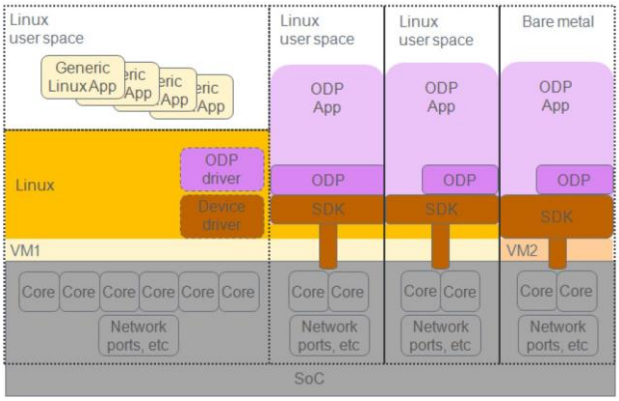


*Figure 2: ODP deployment example with vitalized environment.*

In the virtual environment deployment of ODP, the hypervisor used is a Xen. It is the only open source bare metal hypervisor available in the market. In addition, it has some key features such as small footprint and interface, operating system agnostic, driver isolation, Paravirtualization and many more.

## 2. LITERATURE REVIEW

Before the ODP, there is no such a common, open data plane programming interface across different silicon architecture. Hardware vendors issue individual SDKs to give convenient interface to hardware. The most famous SDK is the Data Plane Development Kit (DPDK) and other such as HyperExec, NetOS.

DPDK is an Intel® implementation of packet processing best practices in software, actually designed for Intel processors. DPDK includes the set of data plane libraries with network interface controller drivers for rapid packet processing. The DPDK gives a programming framework for Intel x86 processors and permits faster development of high speed data packet networking applications. It works from Intel atom processors to Intel Xeon processors.

The DPDK framework produces a set of libraries for specific hardware/software atmospheres through the production of an Environment Abstraction Layer (EAL). The EAL cancels the atmosphere specific and gives a benchmarked programming interface to libraries, obtained hardware accelerators and other hardware and operating system (Linux, FreeBSD) elements. Once the EAL is developed for a specific atmosphere, developers link to the library to develop their applications. For instance, EAL gives the frameworks to support Linux, FreeBSD, Intel IA 32- or 64-bit.

The EAL also gives additional services consisting time references, PCIe bus access, trace and debug functions and alarm operations.

The DPDK executes a much less overhead run-to-completion model for rapid data plane performance and access devices via polling to remove the performance overhead of interrupt processing.

The DPDK includes data plane libraries and enhanced NIC drivers for the following:

- A queue manager execute lockless queues
- A buffer manager pre-assign fixed size buffers
- A memory manager assign pools of objects in memory and uses a ring to store free objects; confirms that objects are spread proportionally on all DRAM channels
- Poll mode drivers (PMD) are developed to work in absence of asynchronous notifications, which reduces overhead
- A packet framework – set of libraries which act as a helpers to develop packet processing

Intel's Data Plane Development Kit (DPDK) giving a new programmable forwarding module and application program interface (API) to the Open Virtual Switch (Open vSwitch) that enhances virtual switching. Intel's DPDK is recreating forwarding module for Open vSwitch that enhances and speedups small-packet throughput, performance and packet processing. Intel is now planning to make its DPDK available to the open source community for consideration for inclusion in the Open vSwitch project, which VMware controls.

## 3. DISCUSSION

DPDK is an Intel® execution of packet processing best practices in software, actually developed for Intel processors. Packet processing is executed in software, interfacing to a NIC, with thin aid for hardware

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

*International Journal of Advanced Research in Computer and Communication Engineering*
*Vol. 5, Issue 2, February 2016*

acceleration below it. DPDK has been placed by its developers as a framework for packet processing, preferring to let developers determine how legacy APIs interface to this framework. In addition, there are a various technical beliefs made that tend to narrow the resilience of SoC vendors wishing to hold different hardware capabilities.

ODP takes a diverse view and does actually want to drive software interoperability by defining specific APIs for common SoC characteristics and acceleration potential. That is why, ODP has a wider scope that has taken the approach to authorize implementers to stretch and embrace DPDK, just like any other vendor-optimized runtime environment, when DPDK's software-centric execution is desired most typically with Intel x86 based systems for which it was designed.

ODP maintains an optimized, low-overhead integration on top of DPDK (using DPDK for packet I/O and as a software accelerator). Legacy applications and interfaces created using DPDK can be enhanced with the addition of ODP APIs to migrate *ad hoc*, legacy hardware interfaces to use ODP to broaden abstraction of hardware acceleration and enable instant access to a growing list of optimized ODP implementations. ODP provides portable versions of equivalent DPDK functionality (e.g., get packet, get buffer) to facilitate easy porting.

## 4. CONCLUSION

ODP opens up a world of innovation and flexibility for silicon OEMs and ISVs and new choice and value for customers. For silicon developers, ODP opens up a much wider addressable market as application developers will find it much easier to write code for varied types of SoCs. For software developers, it provides true software interoperability across diverse silicon architectures without the need to write different versions of code. ISVs can use ODP to take advantage of the unique features of each SoC design.

By providing access layer to the ODP allows open vSwitch to pick up acceleration on supported ODP platforms without change.

One of the most important take away is the fact that ODP is the only project focused on this problem of networking silicon abstraction with a critical mass of representative stakeholders from the networking SoC vendors, including networking OEMs and ISVs.

## REFERENCES

[1] http://www.opendataplane.org
[2] http://www.linaro.org
[3] http://www.openvswitch.org
[4] http://www.xenproject.org
[5] http://dpdk.org


DOI 10.17148/IJARCCE.2016.52118