Data Plane Acceleration

Project Name:
- Proposed name for the project: Data Plane Acceleration
- Proposed name for the repository: dpacc
- Project Categories: Requirements

Project description:
As a result of convergence of various traffic types and increasing data rates, the performance requirements (both in terms of bandwidth and real-time-ness) on data plane devices within network infrastructure have been growing at significantly higher rates than in the past. As the traditional 'bump-in-the-wire' network functions evolve to a virtualized paradigm with NFV, the focus will be even higher to deliver high performance within very competitive cost envelopes. At the same time, application developers have, in some cases, taken advantage of various hardware and software acceleration capabilities, many of which are platform supplier dependent. There is a clear impetus to move away from proprietary data plane interfaces, in favor of more standardized interfaces to leveraging the data plane capability of underlying platforms – whether using specialized hardware accelerators or general purpose CPUs.

The goal of this project is to specify a general framework for VNF data plane acceleration (or DPA for short), including a common suite of abstract APIs at various OPNFV interfaces, to enable VNF portability and resource management across various underlying integrated SOCs that may include hardware accelerators or standard high volume (or SHV) server platforms that may include attached hardware accelerators. It may be desirable, as a design choice in some cases, that such DPA API framework could easily fit underneath existing prevalent APIs (e.g. sockets) – mainly for legacy implementations even though they may not be most performance efficient. But this project should not seek to dictate what APIs an application must use, rather recognizing that API abstraction is likely a layered approach and developers can decide which layer to access directly, depending on the design choice for a given application usage.

This project proposes to define such DPA API framework by considering a set of use cases that are most common and important for data plane devices, namely:
- Usecase1: Packet processing
- Usecase2: Encryption
- Usecase3: Transcoding

By utilizing such cross-usecase, cross-platform and cross-accelerator general framework, it is expected that data plane VNFs can be easily migrated across available SHV server platforms and/or hardware accelerators per communication service provider (or CSP)’s demand, while the CSPs could also change the platform or apply new hardware/software accelerators with minimal impact to the VNFs.

As shown in the following figure, there are basically two alternatives for realizing the data plane APIs for a VNF. The functional abstraction layer framework and architecture should support both, and provide a unified interface to the upper VNF. It will be the person configuring the VNF, hypervisor/host OS and hardware (policies) that decides which model to use.

"Pass-Through" Model

"Fully Intermediated" Model
Note: As one can see the scope of the project includes hardware offloading accelerators (or HWAs) on the local hardware platform or general purpose SHV platforms.

In the “pass-through” model, the VNF is making use of a common suite of DPA APIs in discovering the hardware accelerators and/or generalized network interfaces (NWI) available and using the correct and specific “direct drivers” to directly access the allocated hardware resources. The features of this model include:

1. It enables the most efficient use of hardware resources by bypassing the hypervisor/host OS, yielding higher performance than the other model.
2. It cannot provide “absolute transparency” to the VNFs using hardware accelerators, as they have to upgrade to make changes to their VM image each time they are making use to a new type of hardware accelerator, to load the specific driver and make it known to the application.

Alternatively, there is the “fully intermediated” model where the VNF talks to a group of abstracted functional “synthetic drivers”. These “synthetic drivers” relays the call to a backend driver in the hypervisor that actually interacts with specific driver for the underlying HWA and/or NWI. The features of this model include:

1. Through this intermediate layer in the hypervisor, a registration mechanism is possible for a new HWA to make them mapped to the backend driver and then be used automatically with no changes to the upper VNFs.
2. Access control and/or resource scheduling mechanisms for HWA allocation to different VNFs can also be included in the hypervisor to enable flexible policies for operation considerations.

Scope:

- Problem Statement

As stated earlier, there is an existing problem for application developers who do use various hardware and software acceleration mechanisms that are supplier platform specific, or they would like to be able to leverage such acceleration but are reluctant to have the resultant migration issues when porting software to other platforms. Generally there are varied interfaces to underlying hardware and there is a need to establish a consistent, high performance data plane API that would facilitate development of production data plane VNFs that can make the best use of underlying hardware resources while maintaining portability across platforms.

To this end, the proposed project is intended to

Phase 1: (by 2015Q2)

1. Document typical VNF use-cases and high-level requirements for the generic functional abstraction for high performance data plane and acceleration functions, including hardware and software acceleration; and
2. Identify the potential extensions across various NFV interfaces and evaluate current state-of-art solutions from open-source upstream projects according to identified requirements and targeted framework.

Phase 2: (by 2015Q4)

1. Specify detailed framework/API design/choice and document test cases for selected use-cases;
2. Provide open source implementation for both the framework and test tools;
3. Coordinate integrated testing and release testing results; and
4. Interface specification.

- Interface specification

Using the small cell GW VNF as an example, where the VNF is composed of a signaling GW (SmGW) VM and a security GW (SeGW) VM. In this example, SmGW VM is using hardware acceleration technology for high performance packet processing, while SeGW VM is using IPSec offloading in addition. The following figure highlights the potential extensions to OPNFV interfaces that might be needed to enable hardware-independent data plane VNFs.
What is in or out of scope

1. Data plane acceleration use-cases other than packet-processing, encryption or transcoding are currently out of scope, which could be included in later phases.
2. Management plane APIs are currently out of scope, for the sake of quick application, and could be included in later phases.

The project can be extended in the following aspects in future

1. To include more use-cases;
2. To include management plane interfaces;
3. To coordinate integrated testing and release testing results.

Dependencies:

- There is no similar proposal that is underway or being proposed in OPNFV or upstream project.
- Related upstream projects:
  1. OpenDataPlane (ODP) provides a programming abstraction for networking System on Chip (SoC) devices, including abstractions for packet processing, timers, buffers, events, queues and other hardware and software constructs. ODP is an API abstraction layer which hides the differences between hardware and software implementations, both of which are supported, underneath, which can be unique for each SoC as the author sees fit, and provides unified APIs for some specific functions. ODP is ISA agnostic and has been ported to ARM, MIPS, PowerPC and x86 architectures.
  2. DPDK is a similar project, which provides a set of libraries and drivers for faster packet processing on x86 architecture and has been ported to Open Power, ARM and even MIPS.
  3. OpenCL is a framework for writing programs that execute across heterogeneous platforms consisting of CPUs, GPUs, DSPs, FPGAs and other processors. OpenCL enabled a rich range of algorithms and programming patterns to be easily accelerated.
  4. libvirt is a toolkit to interact with the virtualization capabilities of recent versions of Linux (and other OSes). Its goal is to provide a common and stable layer sufficient to securely manage domains on a single physical machine, where a domain is an instance of an operating system (or subsystem in the case of container virtualization) running on a virtualized machine provided by the hypervisor.
  5. Virtio is a de-facto hardware transparent interface to expose storage and network devices to virtual machines, which is being standardized in OASIS ‘virtio’ group. Virtio framework is one of the candidates to create vendor independent drivers for look-aside accelerators.
  6. Openstack is a cloud operating system that controls large pools of compute, storage, and networking resources throughout a datacenter. It is considered a candidate to implement Virtualized Infrastructure Manager (VIM) for OPNFV platform.

Committers and Contributors:

Names and affiliations of the committers:

- Lingli Deng (China Mobile, denglingli@chinamobile.com)
- Bob Monkman (ARM, Bob.Monkman@arm.com)
- Kin-Yip Liu (Cavium, Kin-Yip.Liu@caviumnetworks.com)
- Xinyu Hu (Huawei, huxinyu@huawei.com)
- Vincent Jardin (6WIND, vincent.jardin@6wind.com)
- Wenjing Chu (DELL, Wenjing_Chu@DELL.com)
- Saikrishna M Kotha (Xilinx, saikrishna.kotha@xilinx.com)
- Bin Hu (AT&T, bh526r@att.com)
- Subhashini Venkataraman (Freescale, subhaav@freescale.com)
Planned deliverables

- Phase 1: (by 2015Q2) Use-cases, requirements and gap analysis
- Phase 2: (by 2015Q4) General framework specification, running code and testing report

Proposed Release Schedule:

- The first release is scheduled by 2015Q2 (tentatively).
- Not planned to be included in the first release of OPNFV.