OPNFV

FastDataStacks

NFV solution stacks for fast and flexible networking

A Linux Foundation Collaborative Project

FastDataStacks

INTRODUCTION

NFV and virtualized high performance applications, such as video processing, require a “fast data stack” NFV solution stack that provides carrier-grade forwarding performance, scalability and open extensibility, along with functionality for realizing application policies and controlling a complex network topology. The OPNFV FastDataStacks project meets this need by building solution stacks based on the world-class forwarding infrastructure supplied by the FD.io Linux Foundation collaborative project.

Any NFV solution stack is only as good as its foundation, i.e. the components that it is built upon. Key foundational assets for a NFV infrastructure are:

- ***The virtual forwarder:*** The virtual forwarder needs to be a feature-rich, high performance, highly scalable virtual switch-router. It needs to leverage hardware accelerators when available and run in user space. In addition, it should be modular and easily extensible.
- ***Forwarder diversity:*** A solution stack should support a variety of forwarders, hardware forwarders (physical switches and routers) as well as software forwarders. This way virtual and physical forwarding domains can be seamlessly glued together.
- ***Policy driven connectivity:*** Business policies should determine network level connectivity, rather than the other way around. Historically the latter has often been the case, which quite often resulted in operational challenges. A classic example is where VLANs represent departments in an enterprise, which lead to non-scalable, enterprise broadcast domain with little to no control.

In order to meet these desired qualities of an NFV infrastructure, the OPNFV FastDataStacks project within OPNFV was started in spring 2016, shortly after the FD.io project. FastDataStacks sets out to compose a variety of scenarios—with all of them using FD.io as a foundation to create an NFV solution that is both fast and flexible.

OPNFV performs system integration as an open community effort, which means OPNFV creates and evolves components in lock-step with upstream communities, composes and integrates those components, deploys and tests the integrated system, and finally publishes the test results—all in an iterative, fully automated way. OPNFV could be seen as an example of DevOps for networking in the open.

The two initial releases of OPNFV, named Arno and Brahmaputra, focused on establishing the base DevOps infrastructure while assembling an initial set of NFV solutions. For the 3rd release, Colorado, the focus was on the "networking" part of NFV—to up the game of the NFV infrastructure solutions composed and integrate a network forwarder that offers the scale, performance and extensibility leading edge NFV and virtualized high performance applications, such as video processing, require. Putting a spotlight on the "network" part of NFV translates into crafting a solution that provides carrier-grade forwarding performance, scalability and open extensibility, along with functionality for realizing application policies and controlling a complex network topology.

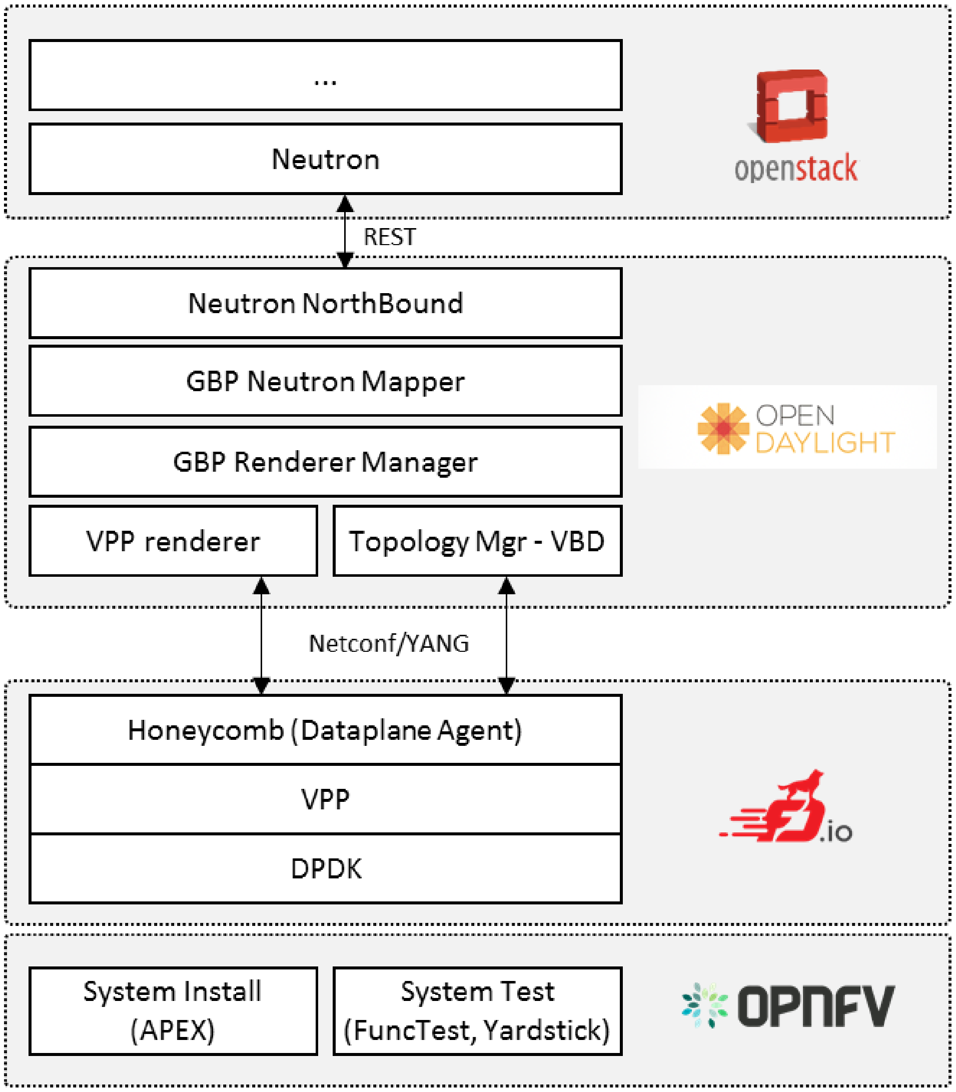
COMPOSING FASTDATASTACKS

Fast, yet flexible forwarding, easy extensibility, support for a large variety of networking configurations—combined with automated install and test capabilities. These are foundational principles for state of the art NFV solution stacks.

OPNFV refers to NFV solution stacks as “scenarios.” A scenario is a composition of software components and their associated configuration. All FastDataStacks scenarios are created with components from a variety of open source projects. While performing the integration, the FastDataStacks project had to integrate and significantly evolve the functionality of the different upstream project components used, as well as evolve the automated installation and testing tools in OPNFV. In this way, FastDataStacks lives up to OPNFV’s modus operandi: create, compose, deploy, test, iterate.

The key ingredient for all these scenarios is the forwarding infrastructure supplied by the FD.io open source project of the Linux Foundation, the Vector Packet Processor (VPP)—along with OpenStack as the VM manager. In addition, OpenDaylight as a network controller plays a key role in many of the scenarios built by FastDataStacks.

The picture showcases typical key components in FastDataStacks:



OpenStack – serves as the VM manager with key building blocks such as Nova, Neutron, Keystone, etc. The network setup is controlled through Neutron, which either uses the networking-odl plugin that interfaces with OpenDaylight as network controller, or the new networking-vpp ML2 driver. Optionally, OpenStack (as well as OpenDaylight mentioned below) can be deployed with high availability (HA). In case of a HA deployment, each of the 3 control nodes will be running the following services: Stateless OpenStack services, MariaDB / Galera, RabbitMQ, OpenDaylight, HA Proxy, Pacemaker, and VIPs. VM migration is configured by the TripleO/APEX installer (see below) and VMs can be evacuated as needed or as invoked by tools such as Heat as part of a monitored stack deployment.

OpenDaylight Controller – an extensible controller platform which offers the ability to separate business logic from networking constructs, supports a diverse set of network devices (virtual and physical) via the “group based policy (GBP)” subsystem, and can be clustered to achieve a highly available deployment. The role of the OpenDaylight network controller in this integration is twofold: It provides a network device configuration and topology abstraction via the OpenStack Neutron interface while offering the capability to realize more complex network policies by means of Group Based Policies. Furthermore, it also provides the capabilities to monitor as well as visualize the operation of the virtual network devices and their topologies. The FastDataStacks project contributes significantly to OpenDaylight projects, like the aforementioned Group Based Policy (GBP) or the Virtual Bridge Domain (VBD) manager.

Note that for simple deployments, FastDataStacks also builds scenarios which relinquish the use of a network controller and directly integrate the VM manager with the network forwarding infrastructure.

HoneyComb dataplane management agent – provides for an easy to use, model driven (YANG) Restconf-based or Netconf-based interface to the forwarding infrastructure. The Honeycomb management agent, which is a project within the FD.io Linux Foundation umbrella project, is built around a generic data processing layer which supplies transaction management and validation of requests. It receives requests through its northbound Restconf or Netconf interfaces and invokes the appropriate low-level binary APIs of the network forwarder VPP (see below). It handles configuration updates as well as offers the operational state of the network forwarder. Key features of Honeycomb include configuration, operational and context data processing, persistence and reconciliation, configuration rollback, CRUD operations on data, notifications as well as simple JSON plugin configuration. When looking at the code base of Honeycomb, OpenDaylight users will find a lot of components that look familiar. This is not a surprise. Honeycomb leverages a couple of key components from the OpenDaylight code base, though does not use any of OpenDaylight’s configuration subsystem or packaging.

Vector Packet Processor – serves as network forwarder and is the cornerstone project of FD.io. VPP is a rapid packet processing development platform for highly performing network applications, with a very high forwarding performance and support of multi-million entry forwarding information bases (FIBs). For example, 480 Gbps of forwarding on just 24 cores while having FIB sizes with more than a million entries are easily achievable. VPP’s forwarding pipeline is vector based and groups features into directed graphs of “feature nodes,” making the forwarding infrastructure both flexible as well as very easy to extend.

OPNFV install tools, APEX/TripleO – is used as the automated deployment tool for the FastDataStacks scenarios. APEX/TripleO leverages OpenStack to install OpenStack and associated components—hence the name TripleO – "OpenStack on OpenStack". TripleO's easy extensibility and flexibility allowed a fast integration of the different components and their configuration for the different scenarios that FastDataStacks builds.

OPNFV test suites, FuncTest and Yardstick – are used to validate the system composition.

All of the above mentioned components had to be evolved to fully meet the needs of the scenarios composed by FastDataStacks. In an "upstream first" mentality, the FastDataStacks project team, which consists of members from a diverse set of companies, submits the required patches to the corresponding upstream projects rather than works on forked code.

SCENARIOS BUILT BY FASTDATASTACKS

The scenarios built by FastDataStacks correspond to the different market needs—with or without a network controller, Layer 2- or Layer 3-focused, simple or highly available.

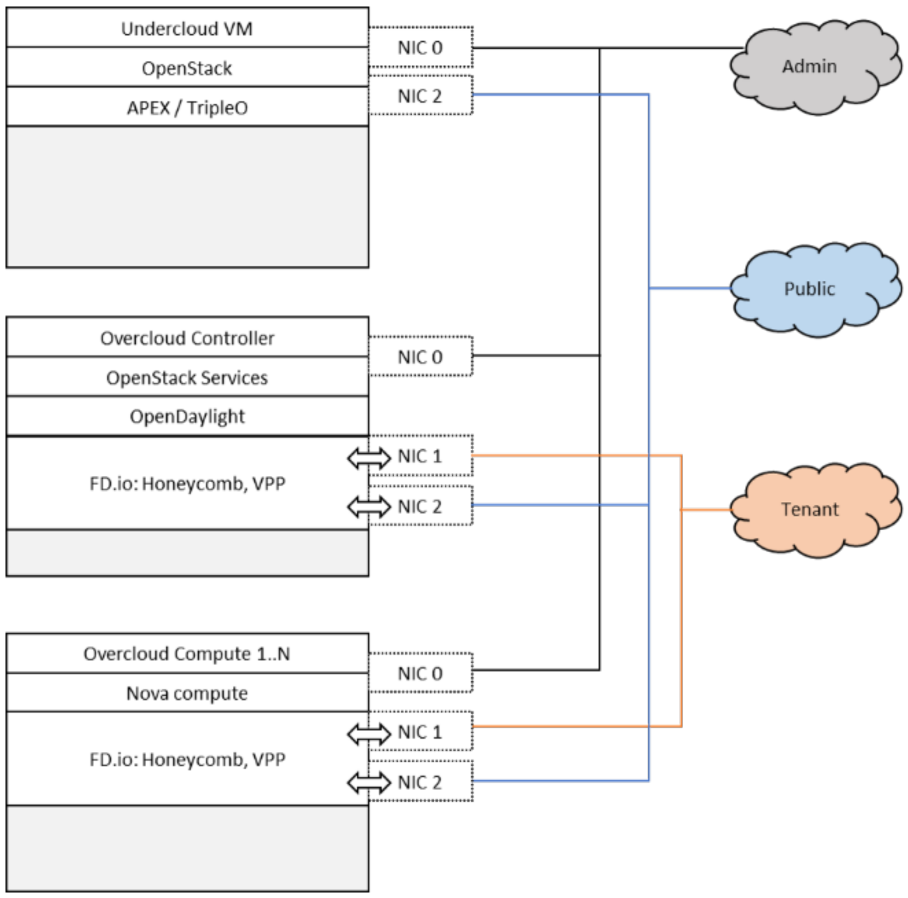
FastDataStacks builds a series of scenarios with a variety of features and functions, following the different needs of the market. These include for example:

- ***"OpenStack – VPP":*** This scenario offers a direct integration of OpenStack with VPP by the means of a new Neutron ML2 driver for VPP. It is well suited for simple network setups, and in that way equally applies to NFV as well as typical cloud compute deployments.
- ***"OpenStack – OpenDaylight (Layer2) – Honeycomb – VPP":*** This solution leverages VPP as a forwarder for traffic within a tenant network, but relies on the classic OpenStack solution ("qrouter") for external connectivity. OpenDaylight only controls layer 2 networking but leaves layer 3 networking to OpenStack. High availability and non-HA deployment options are being created.
- ***"OpenStack – OpenDaylight (Layer3) – Honeycomb – VPP":*** "OpenDaylight(L3)": This means that OpenDaylight controls all aspects of networking, including external connectivity. Tenant networking (either with bridging between VMs or routing between VMs) as well as external connectivity, including simple network address translation (SNAT), are managed through OpenDaylight. High availability and non-HA deployment options are being created.

FastDataStacks scenarios are released once complete and fully tested. The first FastDataStacks scenario, termed "OpenStack – OpenDaylight (Layer2) – Honeycomb – VPP," became available as part of the OPNFV Colorado 1.0 release. All FastDataStacks scenarios continue to be enhanced as new capabilities mature in the upstream projects. In this way, features like service function chaining (SFC), or IPv6, Gluon, etc. will naturally find their way into FastDataStacks.

Following the OPNFV deployment model which defaults to bare-metal deployment, FastDataStacks scenario deployments consist of 3 or more servers:

- 1 Jumphost hosting the TripleO/APEX installer and the so called “undercloud”. The “undercloud” is a purpose-fit virtual OpenStack deployment to drive the installation of the “overcloud”, which is the target OpenStack installation for the particular FastDataStacks scenario.
- 1 to 3 Controlhost(s), which runs the overcloud and OpenStack services. A single controlhost is sufficient for deployments which do not require high availability. High availability deployments which employ OpenStack HA as well as OpenDaylight clustering require 3 control nodes.
- 1 or more Computehosts.



While the individual feature and component composition varies between the scenarios built by FastDataStacks, all share the same install and test tool chain.

The diagram showcases an example deployment scenario of FastDataStacks (“OpenStack – OpenDaylight(Layer 3) – Honeycomb – VPP”).

SCENARIO: “OPENSTACK – FD.IO/VPP”

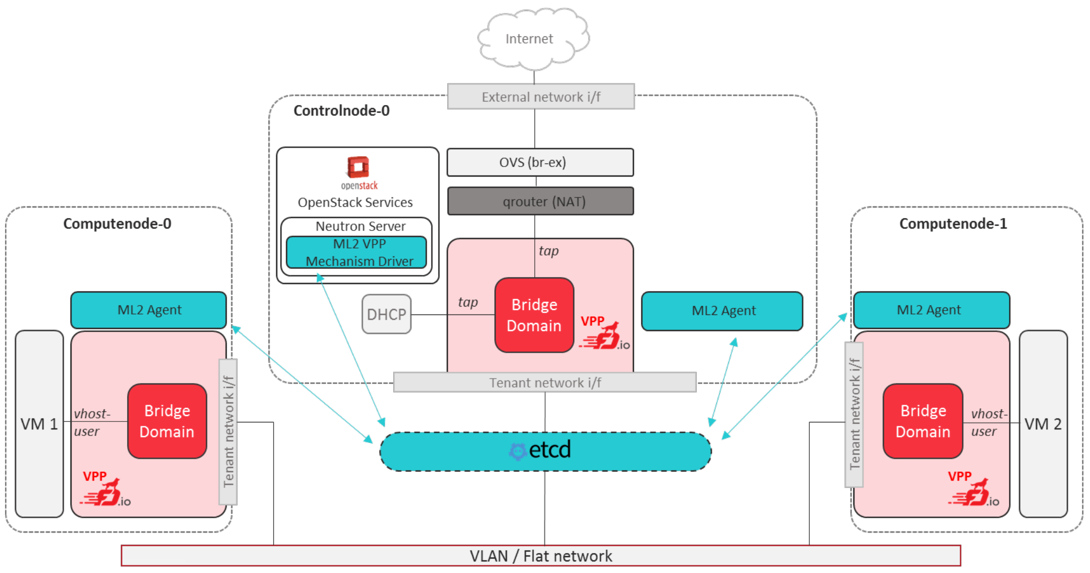
A solution stack for simple, fast and flexible tenant networking for NFV and cloud deployments.

The "OpenStack - FD.io/VPP" scenario provides the capability to realize a set of use cases relevant to the deployment of NFV nodes instantiated by means of an OpenStack system on FD.io/VPP-enabled compute nodes. Direct integration of the forwarder with OpenStack through the means of an OpenStack Neutron ML2-mechanism driver for VPP is often chosen for cloud or simple NFV deployments which do not require sophisticated network control.

The new Neutron mechanism driver brings the advantages of VPP to OpenStack deployments. It’s been written with the following objectives in mind:

- Efficient management communications.
  - Asynchronous, REST-based communication.
- Availability and testability.
  - All state information is maintained in a highly scalable KV store cluster (*etcd*).
  - All modules are unit and system tested.
- Robustness in the face of failures (of one of several Neutron servers, of agents, of the etcd nodes in the cluster gluing them together).
- Simplicity.

The picture outlines the high-level architecture of the overall solution based on the VPP-ML2 mechanism driver:



Tenant networking leverages FD.io/VPP while Open vSwitch (OVS) is used for all other connectivity. In particular, the connectivity to public networking / the Internet (i.e. br-ex) is performed via OVS as in any standard OpenStack deployment. A VPP management agent ("ML2 agent" in the diagram) that corresponds to the Neutron ML2 mechanism driver is used to setup and manage layer 2 networking for the scenario. The Neutron ML2 plugin is configured to use the ML2-VPP networking mechanism driver. Tenant networking can either leverage VLANs or plain interfaces. Layer 3 connectivity for a tenant network is provided centrally via qrouter on the control node. As in a standard OpenStack deployment, the Layer 3 agent configures the qrouter and associated rulesets for security (security groups) and NAT (floating IPs). Public IP network connectivity for a tenant network is provided by interconnecting the VPP-based bridge domain representing the tenant network to qrouter using a tap interface.

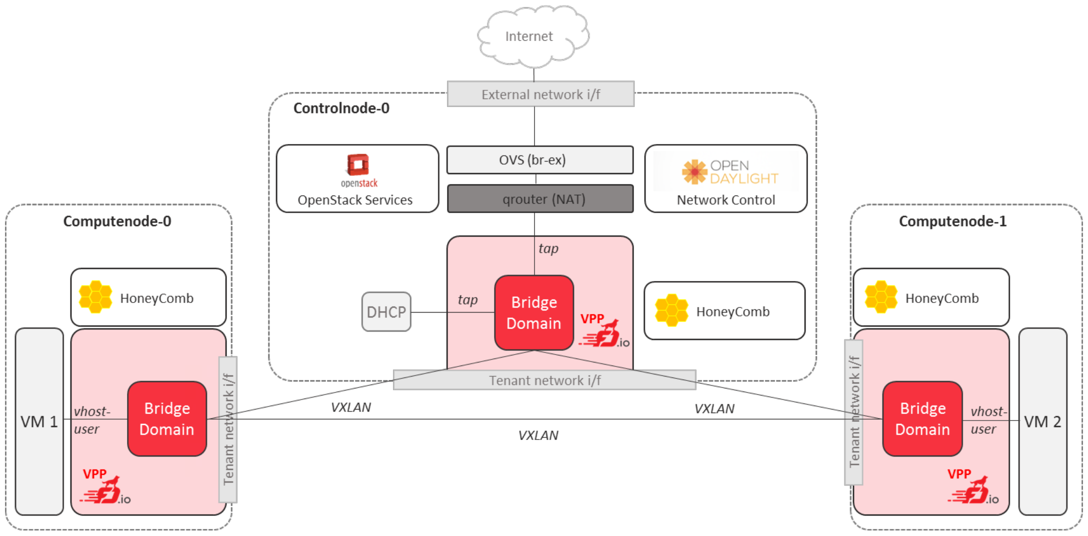
SCENARIO: "OPENSTACK – OPENDAYLIGHT (LAYER2) – HONEYCOMB – VPP"

A NFV focused solution stack for fast and flexible tenant networking and sophisticated network setups.

The "OpenStack - OpenDaylight – Honeycomb – FD.io/VPP" scenario provides the capability to realize a set of use cases relevant to the deployment of NFV nodes instantiated by means of an OpenStack system on FD.io/VPP-enabled compute nodes. The role of the OpenDaylight network controller in this integration is twofold. It provides a network device configuration and topology abstraction via the OpenStack Neutron interface, while providing the capability to realize more complex network policies by means of Group Based Policies. Furthermore, it also provides the capabilities to monitor as well as visualize the operation of the virtual network devices and their topologies. In supporting the general use-case of instantiating an NFV instance, two specific types of network transport use cases are realized:

- NFV instances with VPP data-plane forwarding using a VLAN provider network.
- NFV instances with VPP data-plane forwarding using a VXLAN overlay transport network.

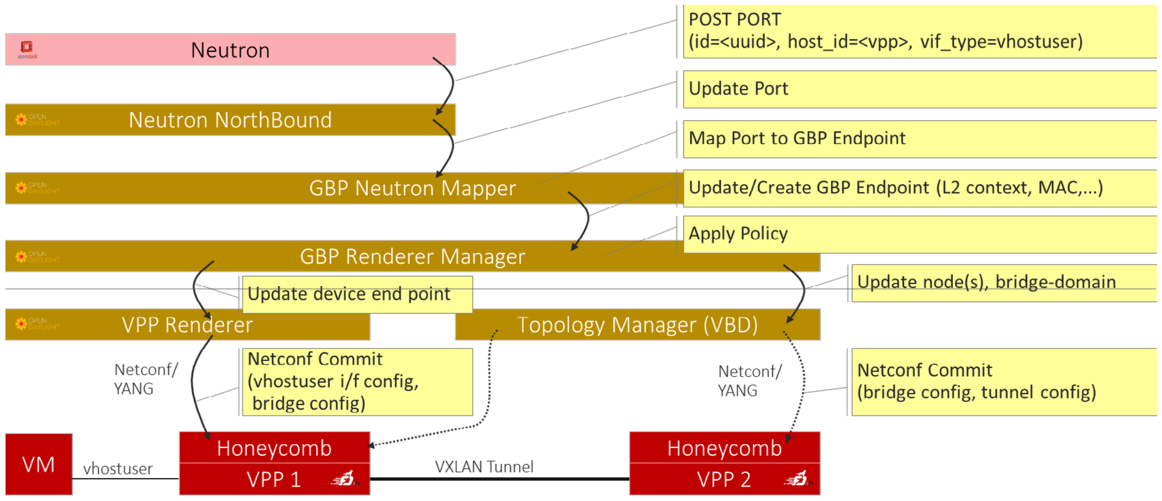
Similar to the "OpenStack – VPP" scenario, tenant networking leverages FD.io/VPP whereas Open vSwitch and the Linux Kernel (qrouter) is used for external connectivity. The OpenDaylight network controller sets up and manages Layer 2 networking for the scenario. Tenant networking can either employ VXLAN (in which case a full mesh of VXLAN tunnels is created) or VLANs for forwarding packets between individual control and compute nodes. Layer 3 connectivity for a tenant network is provided centrally via qrouter on the control node. As in a standard OpenStack deployment, the Layer 3 agent configures the qrouter. Public IP network connectivity for a tenant network is provided by interconnecting the VPP-based bridge domain representing the tenant network to qrouter using a tap interface.



The “OpenStack - OpenDaylight – Honeycomb – FD.io/VPP” can be configured to offer high availability for OpenStack services and OpenDaylight. The picture above outlines the connectivity architecture of the scenario.

The diagram below offers a simplified overview on the call-flow through the different software components of the system when a port is created by OpenStack Neutron. While highlighting the main procedure, the picture also shows the mapping operations between the different abstractions for what constitutes a “Neutron port.” Group Based Policy (GBP) provides for a generic, technology and domain independent way to describe endpoints. This way policies and rules can be applied in a consistent and forwarding technology independent way—offloading the user (here: OpenStack) from understanding the specifics of the different forwarding technologies used.

It is the GBP Neutron Mapper component which translates the request for an “OpenStack Neutron port” into the associated “GBP Endpoint”, i.e. one translates from a use case or domain specific description (OpenStack’s namespace) to a generic description (GBP’s namespace). In a second step, that generic description is mapped onto a specific forwarder and technology (here the VPP namespace). In this case, the GBP endpoint is mapped to a port on VPP. This mapping operation is carried out by the “VPP Renderer” component. Should the endpoint be located on an OVS switch, a corresponding “OVS Renderer” component would be used.

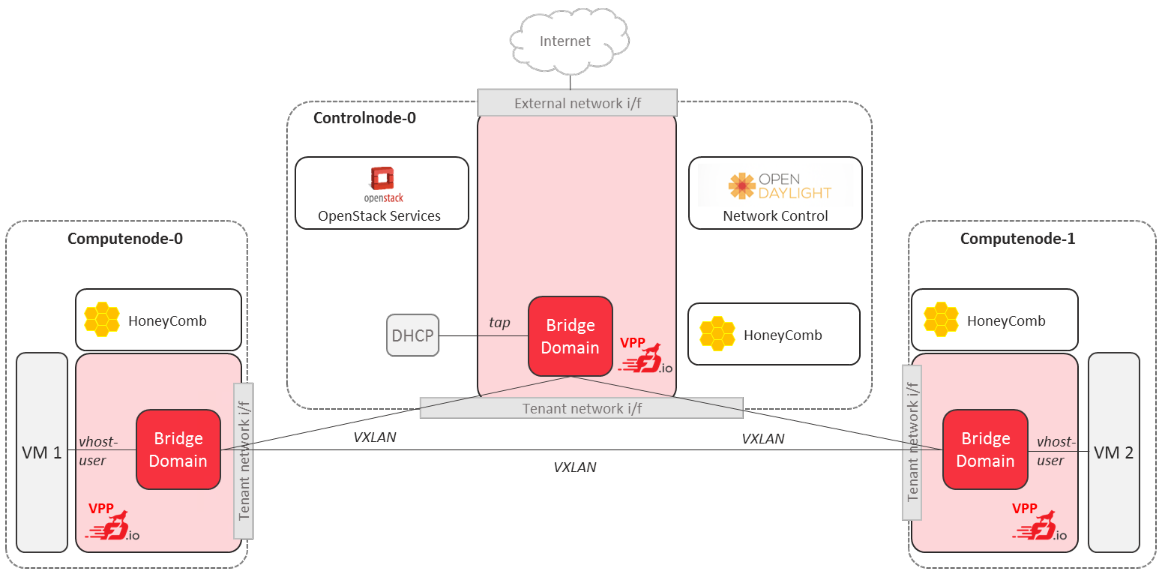

POST PORT
(id=<uuid>, host_id=<vpp>, vif_type=vhostuser)
Neutron
Update Port
Neutron NorthBound
Map Port to GBP Endpoint
GBP Neutron Mapper
Update/Create GBP Endpoint (L2 context, MAC,...)
Apply Policy
GBP Renderer Manager
Update node(s), bridge-domain
Update device end point
VPP Renderer
Topology Manager (VBD)
Netconf/
YANG
Netconf Commit
(vhostuser i/f config,
bridge config)
Netconf/
YANG
Netconf Commit
(bridge config, tunnel config)
VM
vhostuser
Honeycomb
VPP 1
VXLAN Tunnel
Honeycomb
VPP 2

SCENARIO: “OPENSTACK – OPENDAYLIGHT (LAYER2 AND 3) – HONEYCOMB – VPP”

A NFV focused solution stack featuring comprehensive, fast and flexible networking.

This scenario enhances the external connectivity for tenant networking found in the “OpenStack – OpenDaylight (Layer2) – Honeycomb – VPP” scenario discussed earlier and leverages FD.io/VPP for all means of communication. Consequently, VPP is used for Layer 2 and Layer 3 networking and users receive the scale and performance benefits from VPP—such as support for very large MAC-address or IP-routing tables with millions of entries—without an impact to forwarding performance. Initially the Controlnode will serve as the central Layer3 gateway—where VPP serves as a bridge and a router. Future evolutions of the scenario will evolve every VPP node to serve Layer 2 and Layer 3, implementing a fully distributed virtual router solution.

The picture shows an example non-HA setup of the “OpenStack – OpenDaylight (Layer2 and 3) – Honeycomb – VPP” scenario. In this scenario Layer 3 connectivity is provided centrally on the Controlnode.



CONCLUSION

The FastDataStacks project introduces a series of scenarios which include the VPP software supplied by the FD.io project to enable high-performance networking for NFV.

To date, NFV stacks have been missing an easy-to-extend, feature-rich, high-performance/scale virtual switch-router—integrated into a versatile stack architecture that can work with a variety of software and hardware forwarders. The OPNFV FastDataStacks project fills this need and introduces scenarios that leverage the vector packet processor (VPP) as supplied by the FD.io project as the foundation. Further flexibility is achieved by optionally leveraging OpenDaylight Group-Based Policy (GBP) to meet the needs of flexible network deployments while efficiently reflecting business policy. The “OpenStack – OpenDaylight(Layer2) – FD.io/VPP” scenario was the first scenario introduced with the Colorado 1.0 release. Scenarios offering a direct integration of OpenStack and FD.io/VPP or Layer 3 networking using VPP have followed. FastDataStacks will continue to evolve its set of scenarios in lockstep with the many different feature projects in OPNFV (e.g. fault management, supply service function chaining, or Layer 3 VPNs, etc.).

REFERENCES

- FastDataStacks OPNFV project wiki: https://wiki.opnfv.org/display/fds
- FastDataStacks mailing list: fds-dev@lists.opnfv.org
- OPNFV Colorado release: http://www.opnfv.org/colorado
- Fast Data (FD.io): https://fd.io/
- FD.io Vector Packet Processor (VPP): https://wiki.fd.io/view/VPP
- FD.io Honeycomb dataplane management agent: https://wiki.fd.io/view/Honeycomb
- OpenDaylight Controller: https://www.opendaylight.org/
- OpenDaylight Group-Based-Policy project: https://wiki.opendaylight.org/view/Group_Based_Policy_(GBP)
- OpenStack: https://www.openstack.org/
- Neutron ML2 driver for VPP: https://git.openstack.org/cgit/openstack/networking-vpp