OpenStack Based VNF Forwarding Graph

OpenStack Based VNF Forwarding Graph

- Proposed name for the project: OpenStack Based VNF Forwarding Graph
- Proposed name for the repository: vnf_forwarding_graph
- Project Categories: Requirement

Project description:

This project provides a framework for automatic and rapid deployment of end-to-end VNF services through VNFFG, which is one of the NFV components documented in the ETSI NFV Management and Orchestration (MANO) specification [NFVMAN001]. ETSI GS NFV specification also dedicates a specific section (section 6) to VNFFG based network services. VNFs and VNFFGs are also known by the terms Service Function (SF) and Service Function Chain (SFC) respectively.

This project proposal will bring in ongoing OpenStack work on VNFFG and ONF OpenFlow work on service chaining. It also includes integrating OpenStack SFC related components with different vendors’ VNFs instantiated on virtual machines. This project will create a link between OpenStack, ONF OpenFlow, and OPNFV. The diagram below shows an end-to-end service provided by a VNFFG consisting of a number of VNFs.

- VNFFG Architecture Slides:
- Requirement: https://etherpad.opnfv.org/p/vnffg_requirement
- Project specification:
- Blueprints: https://etherpad.opnfv.org/p/vnffg_bps
- Meeting Info

Scope:

The goal of the project is to demonstrate an OpenStack based and OpenFlow compliant solution which will dynamically set up VNFFG so that different tenants’ flows can be steered through different sequence of VNFs.

The diagram below shows the VNF Chaining Architecture which is in line with Figure 4 in the ETSI NFV Architecture [NFVSWA001] and ONF Service Chain Architecture [ONF-SFC]. The NFV VNFFG architecture consists of the OpenStack-based VNFFG Manager, multiple OpenFlow based SDN Controllers, and other components in the NFV infrastructure such as the Traffic Classifiers hosted on the Server, the Service Function Forwarders hosted on vSwitch and the Service Functions running on VMs.
The interfaces between these virtual devices include the management plane interfaces, the control plane interfaces, and the data plane interfaces.

### Components

<table>
<thead>
<tr>
<th>Plane</th>
<th>Components</th>
</tr>
</thead>
<tbody>
<tr>
<td>Management</td>
<td>VNFFG Manager, VNF Instance Manager</td>
</tr>
<tr>
<td>Control</td>
<td>SDN Controller</td>
</tr>
<tr>
<td>Data</td>
<td>Classifier, SF Forwarder(Proxy), VNF Nodes</td>
</tr>
</tbody>
</table>

#### VNFFG Manager

The OpenStack based VNFFG Manager provides an interface for the VNFFG Service Admin to specify a high-level service policy intent for a tenant’s flow from the source to the destination (such as the sequence of service function flavors and the flow classification criteria). The Admin’s requirement should be network topology agnostic. The VNFFG Manager’s role is to translate the Admin’s high-level abstract policy-based VNF service requirements into flow classification constructs as well as the sequence of detail L4-L7 VNF instance constructs.

#### VNF Manager

The VNF (Service Function) Manager manages the life cycle of the VNFs that are hosted on the Service VMs or on external devices. The VNF Manager registers its VNF instances with VNFFG Manager via VNF Registration API. VNFFG Manager imports the VNF instances into its VNF Catalog which is used to create the VNFFGs.

#### SDN Controller

The SDN Controller manages a data-plane service domain that consists of Classifiers, Service Function Forwarders, VNFs and Proxy Devices. The SDN controller is responsible for installing flow rules in the classifiers, SFFs and Proxy Devices to direct the flows to the VNFs. The SDN Controller receives the location information for the VNFs to be included in VNFFGs from the VNFFG Manager.

#### Service Function Forwarder (OpenFlow Switch/Router)
Each SFF is responsible for forwarding the traffic to their designated local VNFs and for forwarding the traffic to the next hop SFF after the local VNF processing. A SFF may also perform a chain termination function where it removes a Service Chain Header (NSH) before delivering the packet to its final destination using normal routing. The SFFs uses an overlay transport network such as VXLAN to forward the traffic in the service domain. The Service Function Forwarder can be a virtual/physical switch or router.

**VNF**

The VNF (Service Function) performs the actual service function treatment on the traffic flows. It can be a virtual device that runs on a VM or a physical device.

**Traffic Classifier**

An ingress Traffic Classifier is used to classify packet flows to different VNFFGs at the entry point of the VNFG. The Classifier performs packet inspection and maps flows to VNFFGs by adding a Service Chain Header (NSH) encapsulation [SFC-SCH] to the packet.

**Proxy Device**

The Proxy Device is used to interface with a non SFC Header-aware VNF.

**Interfaces**

**Management Interfaces**

The management interfaces include:

1. Management interface M1 is between the VNFFG Application/Admin and the VNFFG Manager. This allows the Application/Admin to express a high-level intent for the creation of VNFs and VNFFGs, but does not specify their actual realization. This includes the OpenStack Policy APIs and Advanced Services APIs for specification of user's requirement/intention for service chaining.
2. Management interface M2 is between the VNF Manager and the VNFFG Manager. The VNF Manager uses this API to register (on-board) VNF Instances and VNF Templates with the VNFFG Manager.

**Control Plane Interfaces**

The control plane interfaces include:

1. Control plane interface C1 is between the VNFFG Plugin and different vendors' SDN Controller drivers. This interface allows communication of VNF instance connectivity and flavor details as well as a VNFFG's service function ordering information. The interface information can be passed through transparently to the SDN Controller to become the NBI of the SDN Controller.
2. Control plane interface C2 is between the SDN Controller and the OpenFlow vSwitches. This interface allows communication of OpenFlow commands between the Controller and the vSwitches so that vSwitches can be programmed to correctly steer different flows to different VNFFG paths.

**Data-plane Interfaces**

A data-plane VNFFG starts at the ingress Classifier and traverses Service Function Forwarders, NSH-aware VNFs, Proxy Devices and legacy non NSH-aware VNFs through to a terminating SFF at the end of the VNFFG are shown in the diagram below.
Data-plane interface D1 is the interface between the Classifier, Service Function Forwarder, and Service Proxy Device. An overlay network, such as VXLAN, creates a service transport domain connecting the Classifier, Service Function Forwarder, and the Proxy Device.

Data-plane interface D2 is the interface between the Service Function Forwarder and NSH-aware VNF.

Data-plane interface D3 is the interface between the Proxy Device and non NSH-aware VNF.

**Documentation:**

OPNFV Documentation on all interfaces will be provided.

**Dependencies:**

This project is dependent on the upstream OpenStack projects:

- networking-sfc [http://docs.openstack.org/developer/networking-sfc/](http://docs.openstack.org/developer/networking-sfc/)

The SFC architecture is dependent on ONF SFC architecture specification work in the L4-L7 Work Group [ONF-SFC].

The SFC encapsulation in the data-plane is dependent on ongoing work in the SFC workgroup in IETF [SFC-ARCH] and [SFC-SCH].

**References**

- [NFVMAN001] ETSI GS NFV MAN 001 Management and Orchestration
- [NFVSWA001] ETSI GS NFV SWA 001 NFV Virtual Network Functions Architecture
- [ONF-SFC] ONF Service Chain Architecture
- [SFC-ARCH] IETF RFC 7665 Service Function Chaining (SFC) Architecture
- [SFC-SCH] IETF Network Service Header draft-ietf-sfc-nsh

**Key Project Facts**

* Repository: vnf_forwarding_graph
* Project Category: requirement
* Lifecycle State:
* Primary Contact: cathy.h.zhang@huawei.com
* Project Lead: cathy.h.zhang@huawei.com
* Jira Project Name: vfnforwardinggraph
* Jira Project Prefix: vnf

**Names and affiliations of the committers:**

- Cathy Zhang, cathy.h.zhang@huawei.com
- Louis Fourie, louis.fourie@huawei.com
Feature List

The following features will be developed as part of the project.

- VNF Instance and VNF Template registration (on-boarding) and management
- Intent based specification of a tenant’s flow and its associated service function requirement/intention
- Information (such as 3rd party’s VNF instance connectivity and flavor details and VNFFG’s service function ordering information) passing mechanism between the VNFFG Manager and Network Controllers’ drivers.
- OpenStack based and OpenFlow compliant VNFFG setup.
- NSH encapsulation mechanism for chain-ID and metadata in the data plane.
- Service by-pass feature which can dynamically bypass a VNF in the case that the VNF service is not needed anymore.

Planned Deliverables

This project is built on top of the OpenStack networking-sfc code base (Mitaka release) and has a dependency on successful integration of OpenStack code into OPNFV platform. It is also dependent on current development work for the Tacker project targeted at the OpenStack Mitaka release. So the first phase delivery of this OPNFV project will be a vendor-neutral SFC requirement and Interface specification that is based on OpenStack+SDN Controller integration and is ONF architecture compliant.

The second phase will bring in the code of the OpenStack VNFFG Orchestrator/Manager including the Tacker SFC Interface, SFC Path Construction functionality on the SDN Controller, flow classification and Service Function Forwarder functionality on vSwitch. OpenStack VNFFG Manager will be developed in OpenStack targeted at Newton release time frame and then ported over to OPNFV. SFC path computation will be added to the SDN Controller. Flow classification and SFF functionality will be added to Open vSwitch code base. The interface between SDN Controller and vSwitch will be based on OpenFlow and its extensions.

Contributed Components

The OpenStack VNFFG Manager, SDN Controller, VNFFG Classifier and the vSwitch will be developed as part of the project. In addition, Service Chain related work in OpenStack Horizon will also be included in the contribution.

Contributed Interfaces

The management plane, control plane, and the data plane interfaces described in the “Scope” section will be contributed as part of this project.

Proposed Release Schedule:

May 2016 or aligned with OPNFV release.