sfc

Release Latest

Jan 23, 2019

Contents

CHAPTER 1

Release Documetation

1.1 SFC installation and configuration instruction

1.1.1 Abstract

This document provides information on how to install the OpenDaylight SFC features in OPNFV with the use of os_odl-sfc-(no)ha scenario.

1.1.2 SFC feature desciription

For details of the scenarios and their provided capabilities refer to the scenario description documents:

- <os-odl-sfc-ha>
- <os-odl-sfc-noha>

The SFC feature enables creation of Service Fuction Chains - an ordered list of chained network funcions (e.g. firewalls, NAT, QoS)

The SFC feature in OPNFV is implemented by 3 major components:

- OpenDaylight SDN controller
- Tacker: Generic VNF Manager (VNFM) and a NFV Orchestrator (NFVO)
- OpenvSwitch: The Service Function Forwarder(s)

1.1.3 Hardware requirements

The SFC scenarios can be deployed on a bare-metal OPNFV cluster or on a virtual environment on a single host.

Bare metal deployment on (OPNFV) Pharos lab

Hardware requirements for bare-metal deployments of the OPNFV infrastructure are given by the Pharos project. The Pharos project provides an OPNFV hardware specification for configuring your hardware: http://artifacts.opnfv.org/pharos/docs/pharos-spec.html

Virtual deployment

SFC scenarios can be deployed using APEX installer and xci utility. Check the requirements from those in order to be able to deploy the OPNFV-SFC:

Apex: https://wiki.opnfv.org/display/apex/Apex XCI: https://wiki.opnfv.org/display/INF/XCI+Developer+Sandbox

1.2 SFC Release Notes

1.2.1 Abstract

This document compiles the release notes for the Gambia release of OPNFV SFC

1.2.2 Important notes

These notes provide release information for the use of SFC with the Apex installer, xci tool and Compass4NFV for the Gambia release of OPNFV.

1.2.3 Summary

The goal of the SFC release is to integrate the OpenDaylight SFC project into an OPNFV environment, with either the Apex installer, xci tool or Compass4NFV.

More information about OpenDaylight and SFC can be found here.

- OpenDaylight version "Fluorine SR1"
- Service function chaining
- Documentation built by Jenkins
  - Overall OPNFV documentation
  - *Design document*
  - *User Guide*
  - *Installation Instructions*
  - Release Notes (this document)

1.2.4 Release Data

Project	sfc
Repo/tag	opnfv-7.2.0
Release designation	Gambia 7.2
Release date	January 25th, 2019
Purpose of the delivery	Move to OpenStack Rocky, ODL FLuorine and OVS 2.9.2 (NSH native support) Move to odl_v2 driver in n-sfc

Version change

Module version changes

This release of OPNFV sfc is based on following upstream versions:

- OpenStack Rocky release
- OpenDaylight Fluorine SR1 release
- Open vSwitch 2.9.2

Document changes

This is the first tracked version of OPNFV SFC Gambia. It comes with the following documentation:

- *Design document*
- User Guide
- Installation Instructions
- Release notes (This document)

Reason for version

Feature additions

- *Use odl_v2 driver for n-sfc*
- *Unit test creation*
- *Code refactored*
- *Tests can be run without tacker and with n-sfc directly*

Bug corrections

Deliverables

Software deliverables

No specific deliverables are created, as SFC is included with Apex and Compass4NFV.

Documentation deliverables

- *Design document*
- User Guide
- Installation Instructions
- Release notes (This document)

1.2.5 Known Limitations, Issues and Workarounds

System Limitations

The Gambia 2.0 release has a few limitations:

1 - The testcase sfc_two_chains_SSH_and_HTTP is disabled in this release due to a missing feature in ODL. We are unable to currently update a chain config

Known issues

1 - When tacker is deployed without Mistral, there is an ERROR in the logs and the VIM is always in 'PENDING' state because tacker cannot monitor its health. However, everything works and SFs can be created.

2 - When tacker is deployed without barbican, it cannot be in HA mode because barbican is the only way to fetch the fernet keys.

Workarounds

1.2.6 Test results

The Gambia release of SFC has undergone QA test runs with Functest tests on the Apex and Compass installers and xci utility

1.2.7 References

For more information on the OPNFV Gambia release, please see:

OPNFV

1. OPNFV Home Page
2. OPNFV documentation- and software downloads
3. OPNFV Gambia release

OpenStack

4. OpenStack Rocky Release artifacts
5. OpenStack documentation

OpenDaylight

6. OpenDaylight artifacts

1.3 SFC User Guide

1.3.1 SFC description

The OPNFV SFC feature will create service chains, classifiers, and create VMs for Service Functions, allowing for client traffic intended to be sent to a server to first traverse the provisioned service chain.

The Service Chain creation consists of configuring the OpenDaylight SFC feature. This configuration will in-turn configure Service Function Forwarders to route traffic to Service Functions. A Service Function Forwarder in the context of OPNFV SFC is the "br-int" OVS bridge on an Open Stack compute node.

The classifier(s) consist of configuring the OpenDaylight Netvirt feature. Netvirt is a Neutron backend which handles the networking for VMs. Netvirt can also create simple classification rules (5-tuples) to send specific traffic to a pre-configured Service Chain. A common example of a classification rule would be to send all HTTP traffic (tcp port 80) to a pre-configured Service Chain.

Service Function VM creation is performed via a VNF Manager. Currently, OPNFV SFC is integrated with OpenStack Tacker, which in addition to being a VNF Manager, also orchestrates the SFC configuration. In OPNFV SFC Tacker creates service chains, classification rules, creates VMs in OpenStack for Service Functions, and then communicates the relevant configuration to OpenDaylight SFC.

1.3.2 SFC capabilities and usage

The OPNFV SFC feature can be deployed with either the "os-odl-sfc-ha" or the "os-odl-sfc-noha" scenario. SFC usage for both of these scenarios is the same.

As previously mentioned, Tacker is used as a VNF Manager and SFC Orchestrator. All the configuration necessary to create working service chains and classifiers can be performed using the Tacker command line. Refer to the Tacker walkthrough (step 3 and onwards) for more information.

SFC API usage guidelines and example

Refer to the Tacker walkthrough for Tacker usage guidelines and examples.

CHAPTER 2

Development Documentation

2.1 Service Function Chaining (SFC)

2.1.1 Requirements

This section defines requirements for the initial OPNFV SFC implementation, including those requirements driving upstream project enhancements.

Minimal Viable Requirement

Deploy a complete SFC solution by integrating OpenDaylight SFC with OpenStack in an OPNFV environment.

Detailed Requirements

These are the Fraser specific requirements:

1 The supported Service Chaining encapsulation will be NSH VXLAN-GPE.

2 The version of OVS used must support NSH.

3 The SF VM life cycle will be managed by the Tacker VNF Manager.

4 The supported classifier is OpenDaylight NetVirt.

5 ODL will be the OpenStack Neutron backend and will handle all networking on the compute nodes.

6 Tacker will use the networking-sfc API to configure ODL

7 ODL will use flow based tunnels to create the VXLAN-GPE tunnels

Long Term Requirements

These requirements are out of the scope of the Fraser release.

1 Dynamic movement of SFs across multiple Compute nodes.

2 Load Balancing across multiple SFs

3 Support of a different MANO component apart from Tacker

2.2 Service Function Chaining (SFC)

2.2.1 Introduction

The OPNFV Service Function Chaining (SFC) project aims to provide the ability to define an ordered list of a network services (e.g. firewalls, NAT, QoS). These services are then "stitched" together in the network to create a service chain. This project provides the infrastructure to install the upstream ODL SFC implementation project in an NFV environment.

2.2.2 Definitions

Definitions of most terms used here are provided in the IETF SFC Architecture RFC. Additional terms specific to the OPNFV SFC project are defined below.

2.2.3 Abbreviations

Table 1: Abbreviations

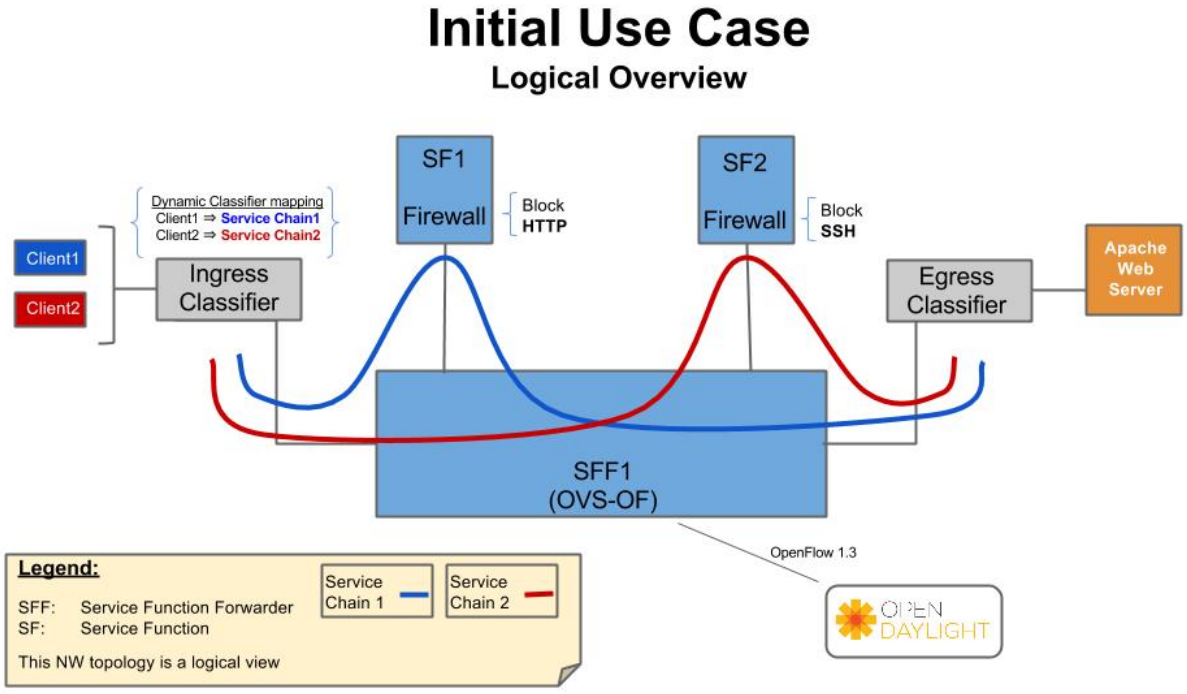
Abbreviation	Term
NS	Network Service
NFVO	Network Function Virtualization Orchestrator
NF	Network Function
NSH	Network Services Header (Service chaining encapsulation)
ODL	OpenDaylight SDN Controller
RSP	Rendered Service Path
SDN	Software Defined Networking
SF	Service Function
SFC	Service Function Chain(ing)
SFF	Service Function Forwarder
SFP	Service Function Path
VNF	Virtual Network Function
VNFM	Virtual Network Function Manager
VNF-FG	Virtual Network Function Forwarding Graph
VIM	Virtual Infrastructure Manager

2.2.4 Use Cases

This section outlines the Danube use cases driving the initial OPNFV SFC implementation.

Use Case 1 - Two chains

This use case is targeted on creating simple Service Chains using Firewall Service Functions. As can be seen in the following diagram, 2 service chains are created, each through a different Service Function Firewall. Service Chain 1 will block HTTP, while Service Chain 2 will block SSH.



Use Case 2 - One chain traverses two service functions

This use case creates two service functions, and a chain that makes the traffic flow through both of them. More information is available in the OPNFV-SFC wiki:

https://wiki.opnfv.org/display/sfc/Functest+SFC-ODL+-+Test+2

2.2.5 Architecture

This section describes the architectural approach to incorporating the upstream OpenDaylight (ODL) SFC project into the OPNFV Danube platform.

Service Functions

A Service Function (SF) is a Function that provides services to flows traversing a Service Chain. Examples of typical SFs include: Firewall, NAT, QoS, and DPI. In the context of OPNFV, the SF will be a Virtual Network Function. The SFs receive data packets from a Service Function Forwarder.

Service Function Forwarders

The Service Function Forwarder (SFF) is the core element used in Service Chaining. It is an OpenFlow switch that, in the context of OPNFV, is hosted in an OVS bridge. In OPNFV there will be one SFF per Compute Node that will be hosted in the "br-int" OpenStack OVS bridge.

The responsibility of the SFF is to steer incoming packets to the corresponding Service Function, or to the SFF in the next compute node. The flows in the SFF are programmed by the OpenDaylight SFC SDN Controller.

Service Chains

Service Chains are defined in the OpenDaylight SFC Controller using the following constructs:

SFC A Service Function Chain (SFC) is an ordered list of abstract SF types.

SFP A Service Function Path (SFP) references an SFC, and optionally provides concrete information about the SFC, like concrete SF instances. If SF instances are not supplied, then the RSP will choose them.

RSP A Rendered Service Path (RSP) is the actual Service Chain. An RSP references an SFP, and effectively merges the information from the SFP and SFC to create the Service Chain. If concrete SF details were not provided in the SFP, then SF selection algorithms are used to choose one. When the RSP is created, the OpenFlows will be programmed and written to the SFF(s).

Service Chaining Encapsulation

Service Chaining Encapsulation encapsulates traffic sent through the Service Chaining domain to facilitate easier steering of packets through Service Chains. If no Service Chaining Encapsulation is used, then packets much be classified at every hop of the chain, which would be slow and would not scale well.

In ODL SFC, Network Service Headers (NSH) is used for Service Chaining encapsulation. NSH is an IETF specification that uses 2 main header fields to facilitate packet steering, namely:

NSP (NSH Path) The NSP is the Service Path ID.

NSI (NSH Index) The NSI is the Hop in the Service Chain. The NSI starts at 255 and is decremented by every SF. If the NSI reaches 0, then the packet is dropped, which avoids loops.

NSH also has metadata fields, but that's beyond the scope of this architecture.

In ODL SFC, NSH packets are encapsulated in VXLAN-GPE.

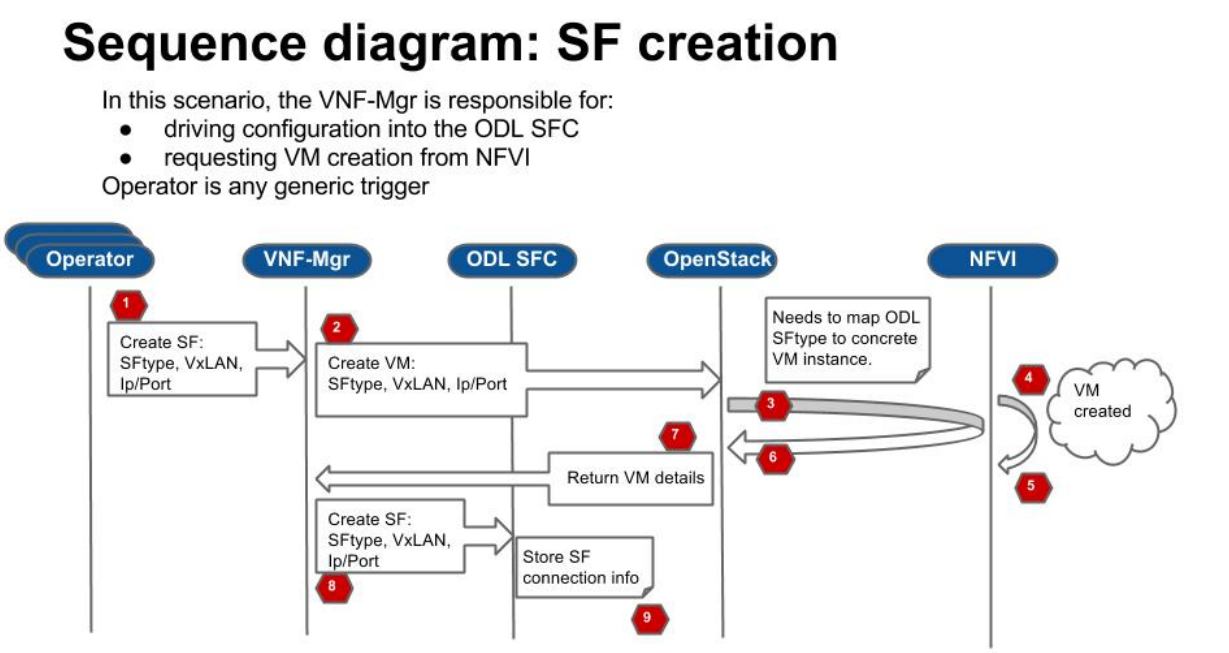
Classifiers

A classifier is the entry point into Service Chaining. The role of the classifier is to map incoming traffic to Service Chains. In ODL SFC, this mapping is performed by matching the packets and encapsulating the packets in a VXLAN-GPE NSH tunnel.

The packet matching is specific to the classifier implementation, but can be as simple as an ACL, or can be more complex by using PCRF information or DPI.

VNF Manager

In OPNFV SFC, a VNF Manager is needed to spin-up VMs for Service Functions. It has been decided to use the OpenStack Tacker VNF Mgr to spin-up and manage the life cycle of the SFs. Tacker will receive the ODL SFC configuration, manage the SF VMs, and forward the configuration to ODL SFC. The following sequence diagram details the interactions with the VNF Mgr:



OPNFV SFC Network Topology

The following image details the Network Topology used in OPNFV Danube SFC: