Leveraging OPNFV test tools beyond the NFV domain

Emma Foley, Georg Kunz

@EmmaL Foley, @the_georg_kunz
Purpose of this talk

1. Create awareness for OPNFV test tools
   - Targeting telcos not active in OPNFV and users outside of NFV domain
   - Beneficial for most cloud operators and developers
   - Leverage the extensive tooling OPNFV has built over 4 years

2. Have a discussion about the evolution of the OPNFV test tools
   - How to evolve the test tools to address emerging use cases?
   - Learn from people outside of NFV domain about their needs
OPNFV facilitates the development and evolution of NFV components across various open source ecosystems. Through system level integration, deployment and testing, OPNFV creates a reference NFV platform to accelerate the transformation of enterprise and service provider networks. Participation is open to anyone, whether you are an employee of a member company or just passionate about network transformation.
What does OPNFV do?

- Development / Integration
- Code
- CI
- Images
- Platform Deployment
- Systems
- Testing & Reporting

Federated OPNFV labs

Requirements -> Development / Integration -> Code -> CI -> Images -> Platform Deployment -> Systems -> Testing & Reporting
OPNFV Test Ecosystem

OPNFV Feature Tests
- Functest
  - NFVI, VIM APIs + Functions
- RefStack
- Tempest
- Cloudify
- VNF
- Kubernetes

Functional Testing

Non-functional Testing
- Yardstick
  - NFVI & VNF Performance
- Bottlenecks
  - Load tests staging manager
- vsperf, NFVBenchn
  - NFVI dataplane Performance
- StorPerf
  - NFVI Storage Performance

Test Result Database
- Dovetail
  - OPNFV Compliance Verification

Performance
- NFVI & VNF Performance
- Load tests staging manager
- NFVI dataplane Performance
- NFVI Storage Performance

Bottlenecks
- NFVI Storage Performance

The Linux Foundation
OPNFV Test Ecosystem

**Functional Testing**
- Functest
- RefStack
- Tempest
- VNF
- Kubernetes

**OPNFV Feature Tests**
- NFVI, VIM APIs + Functions

**Non-functional Testing**
- Yardstick: NFVI & VNF Performance
- Bottlenecks: Load tests staging manager
- vsperf, NFVBench: NFVI dataplane Performance
- StorPerf: NFVI Storage Performance

**Test Result Database**

**Dovetail**
- OPNFV Compliance Verification

**Compliance Verification**
OPNFV Test Ecosystem

Non-functional Testing

Yardstick
- NFVI & VNF Performance

Bottlenecks
- Load tests staging manager

vperf, NFVBench
- NFVI dataplane Performance

StorPerf
- NFVI Storage Performance

Dovetail
- OPNFV Compliance Verification

Test Result Database

Functional Testing

OPNFV Feature Tests

Functest
- NFVI, VIM APIs + Functions

RefStack
- openstack

Tempest
- openstack

Cloudify
- VNF Clearwater

Kubernetes

OPNFV Feature Tests

NFVI & VNF Performance

Load tests staging manager

NFVI dataplane Performance

NFVI Storage Performance

OPNFV Compliance Verification
OPNFV Test Ecosystem

**Functional Testing**
- OPNFV Feature Tests
  - NFVI, VIM APIs + Functions
- ReferStack
- Tempest
- Cloudify
- VNF
- Kubernetes

**Non-functional Testing**
- Yardstick
  - NFVI & VNF Performance
- Bottlenecks
  - Load tests staging manager
- NFV Bench
  - NFVI dataplane Performance
- StorPerf
  - NFVI Storage Performance

**Test Result Database**

**Dovetail**
- OPNFV Compliance Verification
OPNFV Test Ecosystem

OPNFV Feature Tests
- NFVI, VIM APIs + Functions
- RefStack
- Tempest
- Cloudify
- VNF Clearwater
- Kubernetes

Functional Testing
- Functest
- NFVI & VNF Performance
- Load tests staging manager
- NFVI dataplane Performance
- NFVI Storage Performance
- NFVI & VNF Performance
- Load tests staging manager
- NFVI dataplane Performance
- NFVI Storage Performance

Non-functional Testing
- Yardstick
- Bottlenecks
- vsperrf, NFVBech
- StorPerf

Test Result Database
- Dovetail
- OPNFV Compliance Verification

The Linux Foundation
OPNFV Test Ecosystem

**Functional Testing**
- OPNFV Feature Tests
- NFVI, VIM APIs + Functions
- RefStack (openstack)
- Tempest (openstack)
- Cloudify
- VNF Clearwater
- Kubernetes

**Non-functional Testing**
- Yardstick
  - NFVI & VNF Performance
- Bottlenecks
  - Load tests staging manager
- vsperf, NFVBench
  - NFVI dataplane Performance
- StorPerf
  - NFVI Storage Performance

**Test Result Database**

**Compliance Verification**
- Dovetail
- OPNFV Compliance Verification
OPNFV Test Ecosystem

**Functional Testing**
- NFVI, VIM APIs + Functions
- RefStack (openstack)
- Tempest (openstack)
- Cloudify
- VNF
- Kubernetes

**Non-functional Testing**
- Yardstick: NFVI & VNF Performance
- Bottlenecks: Load tests staging manager
- vsperf, NFVBank: NFVI dataplane Performance
- StorPerf: NFVI Storage Performance

**Test Result Database**
- Dovetail
- OPNFV Compliance Verification

**Feature Tests**
OPNFV Test Ecosystem

OPNFV Feature Tests

NFVI, VIM APIs + Functions

RefStack

Tempest

Cloudify VNF

Kubernetes

Functional Testing

Yardstick

Bottlenecks

vsperf, NFVBench

StorPerf

Non-functional Testing

NFVI & VNF Performance

Load tests staging manager

NFVI dataplane Performance

NFVI Storage Performance

Test Result Database

Dovetail

OPNFV Compliance Verification

Compliance Verification
OPNFV Compliance Program

• OPNFV Verified Program (OVP) verifies that a commercial cloud platform exposes the same
  – key APIs,
  – behaviors, and
  – characteristics
  as a reference platform defined through a specific selection of test cases

• Main objective: Reduce vendor selection and application onboarding cost
  – Establish industry-accepted technical baseline
  – Simplify RFIs and RFPs

• Main components of OVP
  1. OPNFV test frameworks providing the actual OPNFV and upstream test cases
  2. Dovetail: Wrapper for OPNFV test tools and reporting tool
OPNFV Compliance Program

**Functional Testing**
- Functest
- NFVI, VIM APIs + Functions

**Non-functional Testing**
- Dovetail
- Test Result Database

**Compliance**
- OPNFV Feature Tests
- RefStack
- Tempest
- Cloudify
- Kubernetes

**Yardstick**
- NFVI & VNF Performance

**Bottlenecks**
- Load tests staging manager

**Non-functional Testing**
- vsperf, NFVBench
- StorPerf
- NFVI dataplane Performance
- NFVI Storage Performance

**Utilized by OVP 2018.09**

Candidates for later releases
Addressing emerging use cases
Addressing emerging use cases

• OPNFV traditionally focused on NFVi data center scenarios
  – Medium to large scale deployments in centralized data centers
  – VNFs = legacy Network Functions in VMs

• Emerging use cases impose new requirements on test tools
  – Edge computing
  – Cloud native computing

⇒ How to address those requirements?
Edge Computing

**Use cases**
- Low latency applications
- High-bandwidth applications
- ...

**Requirements**
- Small hardware footprint
- Zero touch deployment, provisioning and configuration
- (Some level of) autonomy in case of disconnects from higher-level sites
- ...

---

national sites

regional sites

edge sites

Edge Computing
Edge Computing

• Impact of edge computing on test tools and methods
  – Test topology
    • Automatic deployment of multiple sites
    • Inter-site connectivity
  – Consideration of networking effects
    • Control and data plane latency
    • Limited bandwidth, jitter, packet drops
    • Inter-site connectivity
  – Hardware resources
    • Limited resources in the edge: 1-4 servers
Virtual Edge in a Box

- **OPNFV XCI**
  - Mini flavor installs OpenStack from master in VMs
  - Can itself be in a VM
  - 2 full OpenStack environments in 1 server
Modeling of Edge Networking Environment

- Site 1 VM
  - Keystone

- Site 2 VM
  - Keystone

- Network Impairment VM
  - nic1
  - Netem delay N ms
  - nic2
Cloud Native Computing

- Monolithic App

- Break down into smaller chunks

- Microservice architecture puts functionality into separate services:
  - Iterative development
  - Division of labor
  - Reduce single point of failure
  - Language/deployment flexibility
  - Build different apps using subsets of services
Micro-service Instrumentation

- **ConfigMaps**
  - Manage/inject app configuration
  - Kubernetes resource
  - Keep containers agnostic

- **GRPC**
  - Open-source RPC framework
  - Client/server
  - Bindings for most languages

- **Shared Data Stores**
  - Exchange network data, state management
• Consider cloud native for OPNFV test projects
  – Package as micro-services
  – Many are already containerized
    • Functest divided into 8+
  – Add GRPC or REST server interfaces
  – Make actions more atomic within each
  – Orchestrate system level tests using different combinations of services/actions
  – Deploy all OPNFV test services in a single manifest potentially
  – Use tool-chains such as Spinnaker for CI/CD
  – Installer projects are also considering cloud native for some services
Summary

• Join us!
  – OPNFV test working group
    • https://wiki.opnfv.org/display/testing/TestPerf

  – OPNFV
    • https://wiki.opnfv.org/, https://www.opnfv.org/

  – OPNFV Verified
    • https://www.opnfv.org/

• Provide feedback and input!
opnfv-users@lists.opnfv.org

#functest
#yardstick #nsb
#bottlenecks
#nfvbench #vsperf
#dovetail
Backup
Test tools in more detail
Functest in a nutshell

• Verify any kind of OpenStack and Kubernetes deployments (OPNFV model) or production environments
• Conform with upstream rules (OpenStack gate jobs and Kubernetes conformance tests)
• Ensure that the platforms meet Network Functions Virtualization requirements
Functest suites

- All functional tests as defined by the upstream communities (e.g. Tempest, neutron-tempest-api, Barbican, Patrole...)
- Upstream API and dataplane benchmarking tools (Rally, Vmtp and Shaker)
- Virtual Network Function deployments and testing (vIMS, vRouter and vEPC)
Yardstick’s Goal is to **verify infrastructure compliance** from the perspective of a Virtual Network Function (VNF).

Yardstick’s scope is the development of a **testing framework**, test cases and test stimuli to enable NFVI verification. Yardstick also includes NSB (Network services benchmarking).
Bottlenecks

3. Upstream Develop
   - SFC
   - IPv6
   - SDNVPN
   - Installers
   - OVS4NFV

1. Classified bottlenecks

2. Feedback bottlenecks

4. Performance Improvement

DB

OVS4NFV

Yardstick

Test Cases
   - Network
   - Storage
   - Compute
   - Middleware
   - APP

Test Results

Bottlenecks Testing Results

Monitoring

OPNFV Reference Platform
   - OpenStack
   - Kubernetes

THE LINUX FOUNDATION
VSPerf

- Automated Framework for dataplane performance benchmarking,
  - Switching Technologies with Physical and Virtual Interfaces
- Configuration and control of topology, vswitch, VNF, traffic-generator and other software components are performed by VSPERF.
  - VSPERF provides the user the ability to choose the vswitch, Traffic-generator, VNF, etc.
- VSPERF is used as a tool for optimizing switching technologies, qualifying packet processing components and for pre-deployment evaluation of the NFV platform datapath.
- Virtual Switches:
  - OVS, VPP
- Traffic Generators
  - T-Rex, Spirent, Ixia, Xena, Moongen
- Deployment Scenarios
  - Phy2Phy, PVP, PVVP, Custom.
- VSPERF tests are defined and driven by Level Test Design (LTD) Specification.
  - VSPERF supports designing and implementing custom tests through its 'integration-tests' feature.
- VSPERF supports multiple modes:
  - Ex: Trafficgen-off mode: VSPERF will do setup of DUT, but no control the traffic-generator.
VSPerf

Systems:
Deployment scripts for Ubuntu, SLES, Centos, Fedora, RHEL, Opensuse

Configuration: vswitch, traffic, vnf, pktfwd, custom, collector, loadgen, common

Sources:
- ovs
- vpp
- l2fwd
- dpdk
- trex
- qemu

Loader

Component Factory

Testcase

VSPERF

Publish results to OPNFV Dashboard

Report Results in different formats

Define Custom Steps

Other Utils: tasks, veth, module, etc.

TOOLS

VNF-Controller

vSwitch-Controller

vSwitch

Traffic-Controller

Traffic-Gen

load_gen Controller

Collector Controller

Collector

loadGen

Traffic-Gen

Ixia

Spirent

Stress-ng

Stressor VMs

Moongen

T-Rex

Stress-ng

Stressor VMs

Xena

pidstat

Collectd

VNFs

VnfQemu

p2p

tpv

cln

pv2

pvvp

op2p

ptunp

ovs-vanilla

ovs-dpdk-vhost

vpp-dpdk-vhost

ixia

spirent

stress-ng

moongen

t-rex

xena

pidstat

collectd
NFVbench

- Tool that provides an automated way to measure the network performance for the most common data plane packet flows on any OpenStack system.
- Designed to be easy to install and easy to use by non-experts — there is no need to be an expert in traffic generators and data plane performance testing.
- The tool is built around the open source T-Rex traffic generator and is useful for testing a full NFVI subsystem that includes ToR switches.
- The key areas of strength for NFVbench are in its automation of the traffic generator, ability to test a full subsystem, and to perform this testing on a production cloud.