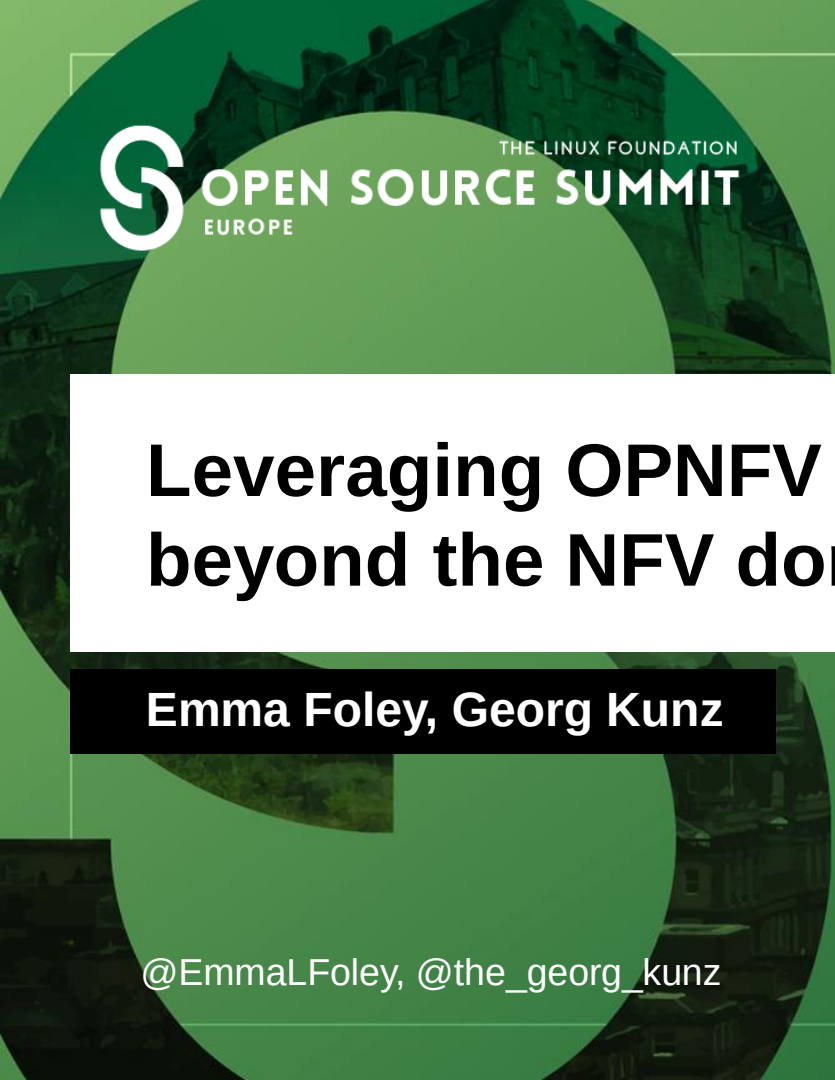


Leveraging OPNFV test tools beyond the NFV domain

Emma Foley, Georg Kunz

@EmmaLFoley, @the_georg_kunz

THE LINUX FOUNDATION

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

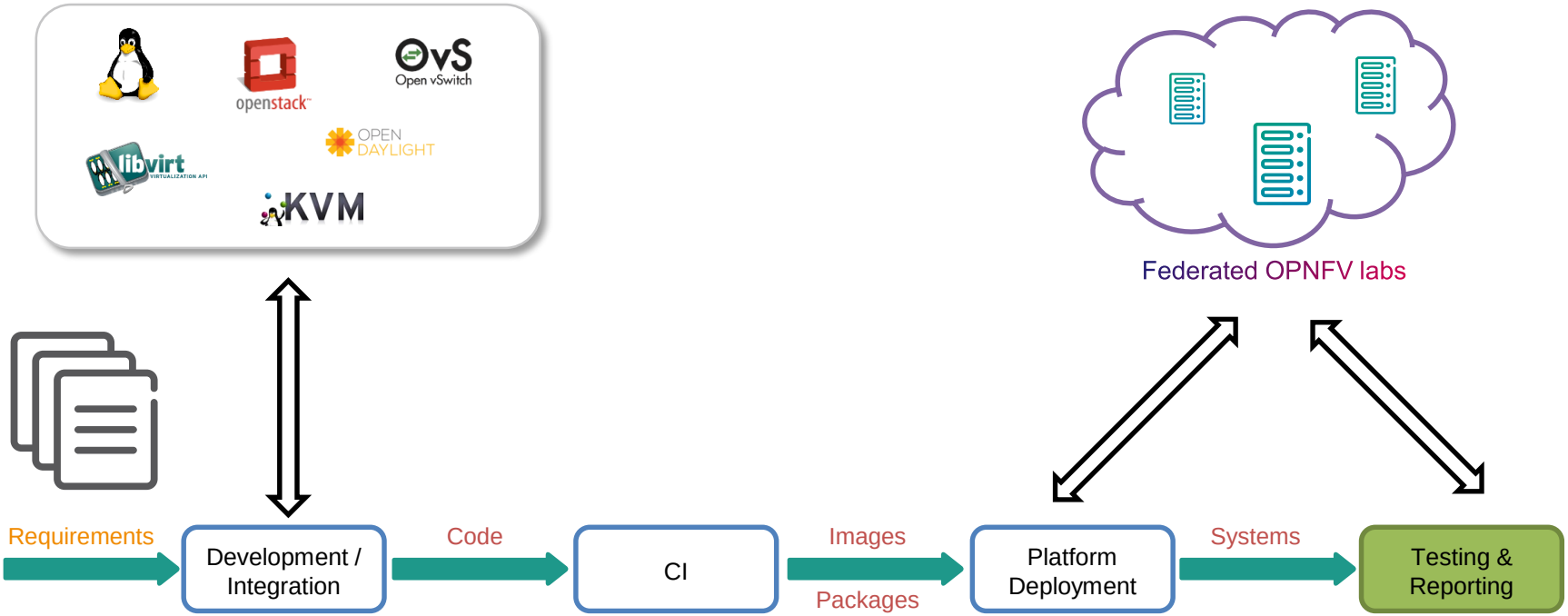
OPNFV

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.

OPNFV defines use cases, integrates & tests what other projects (OpenStack, Kubernetes, ODL, OVS, fd.io) create!

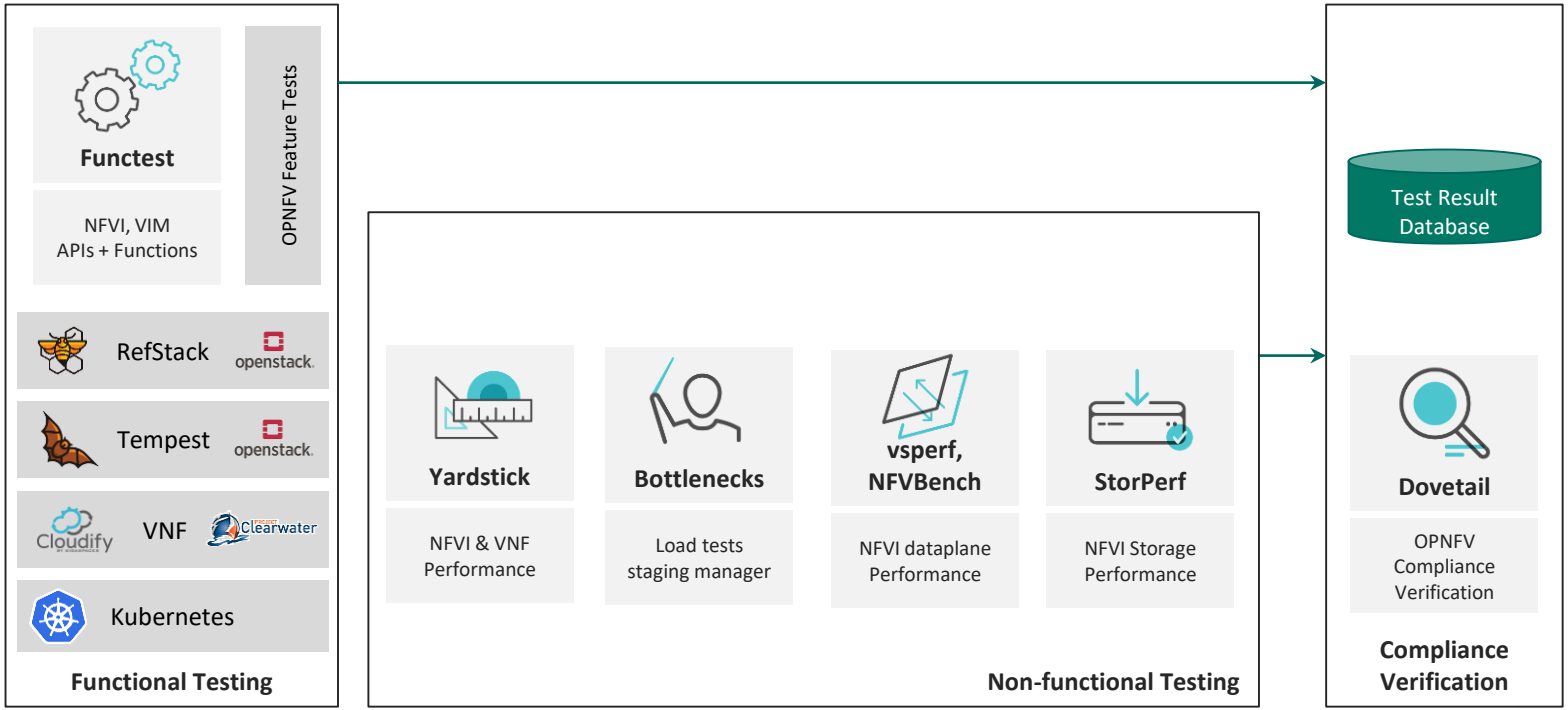
THE LINUX FOUNDATION

What does OPNFV do?



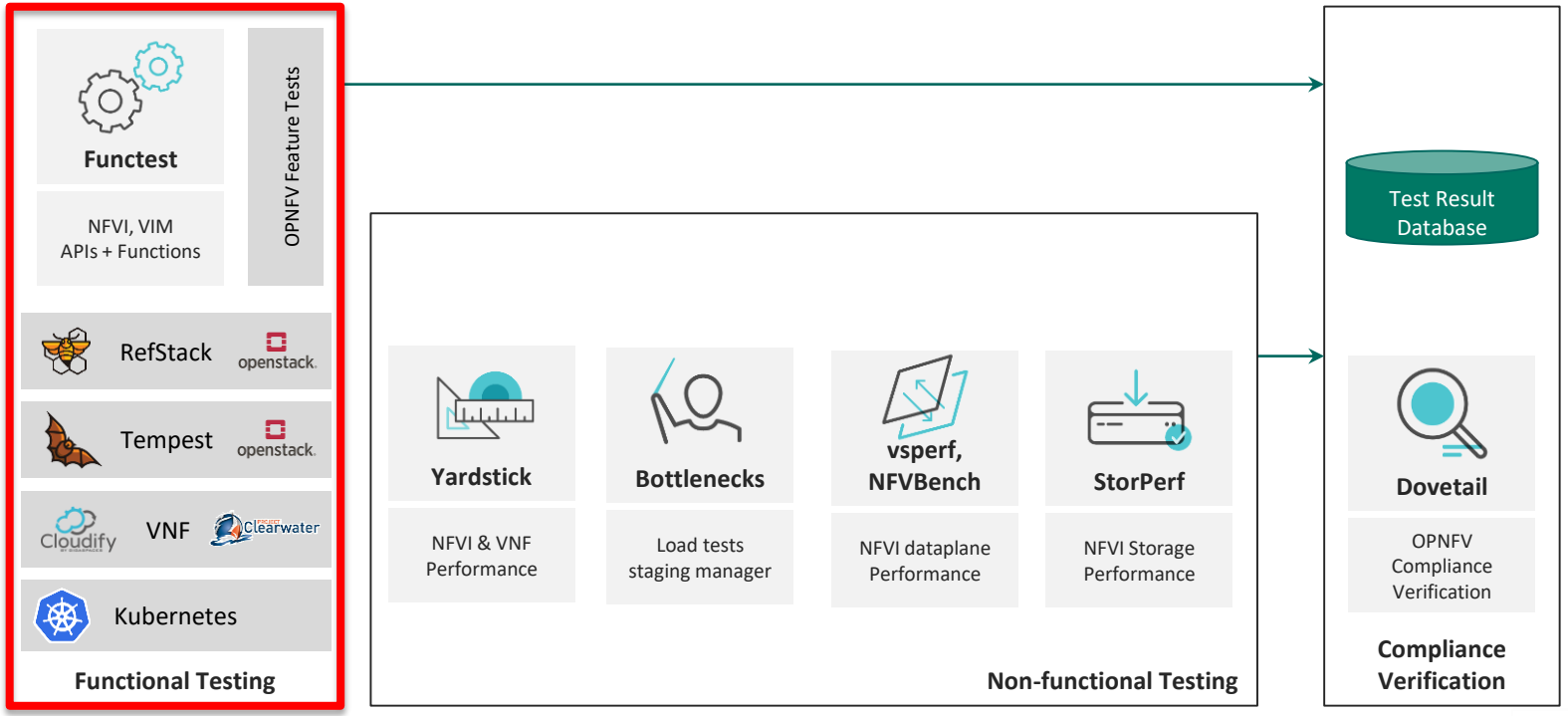
THE LINUX FOUNDATION

OPNFV Test Ecosystem



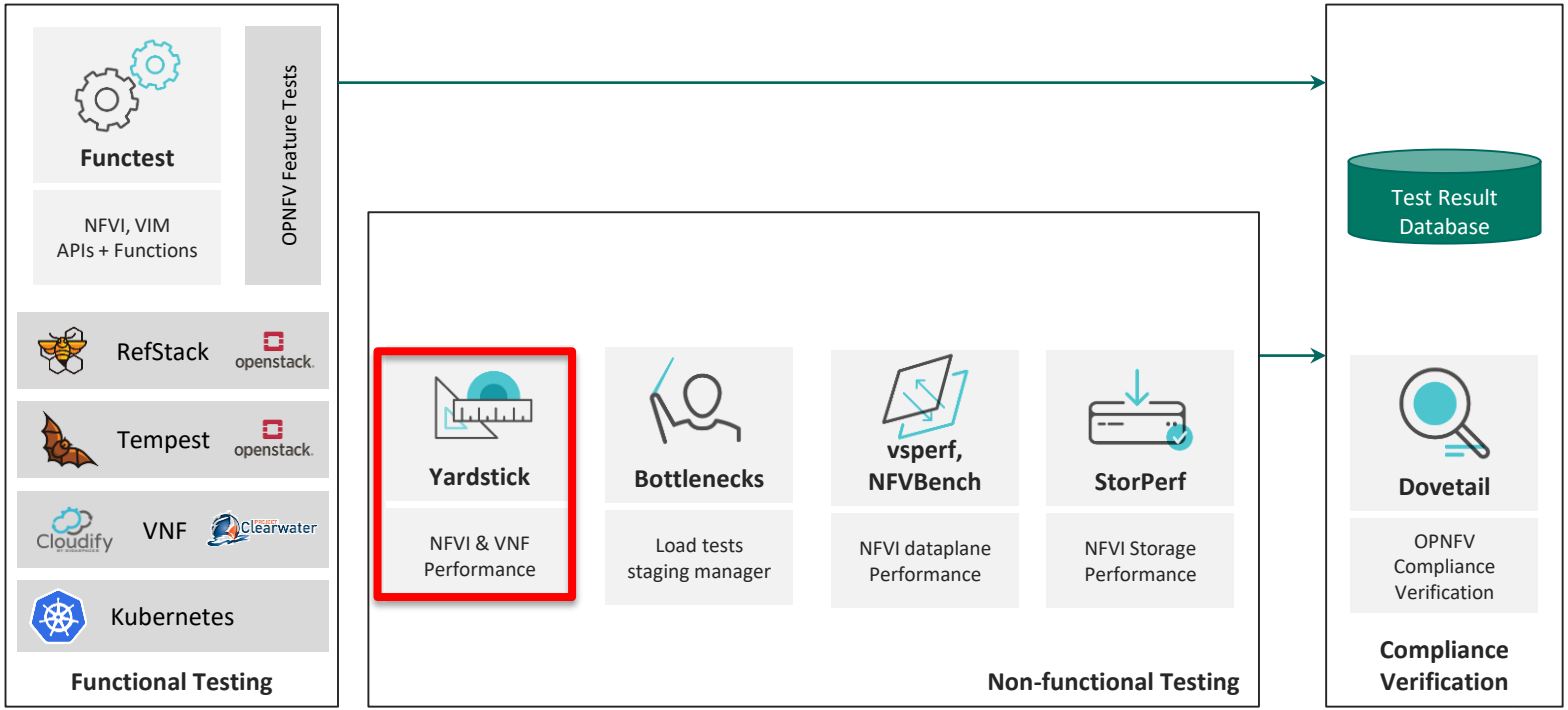
THE LINUX FOUNDATION

OPNFV Test Ecosystem

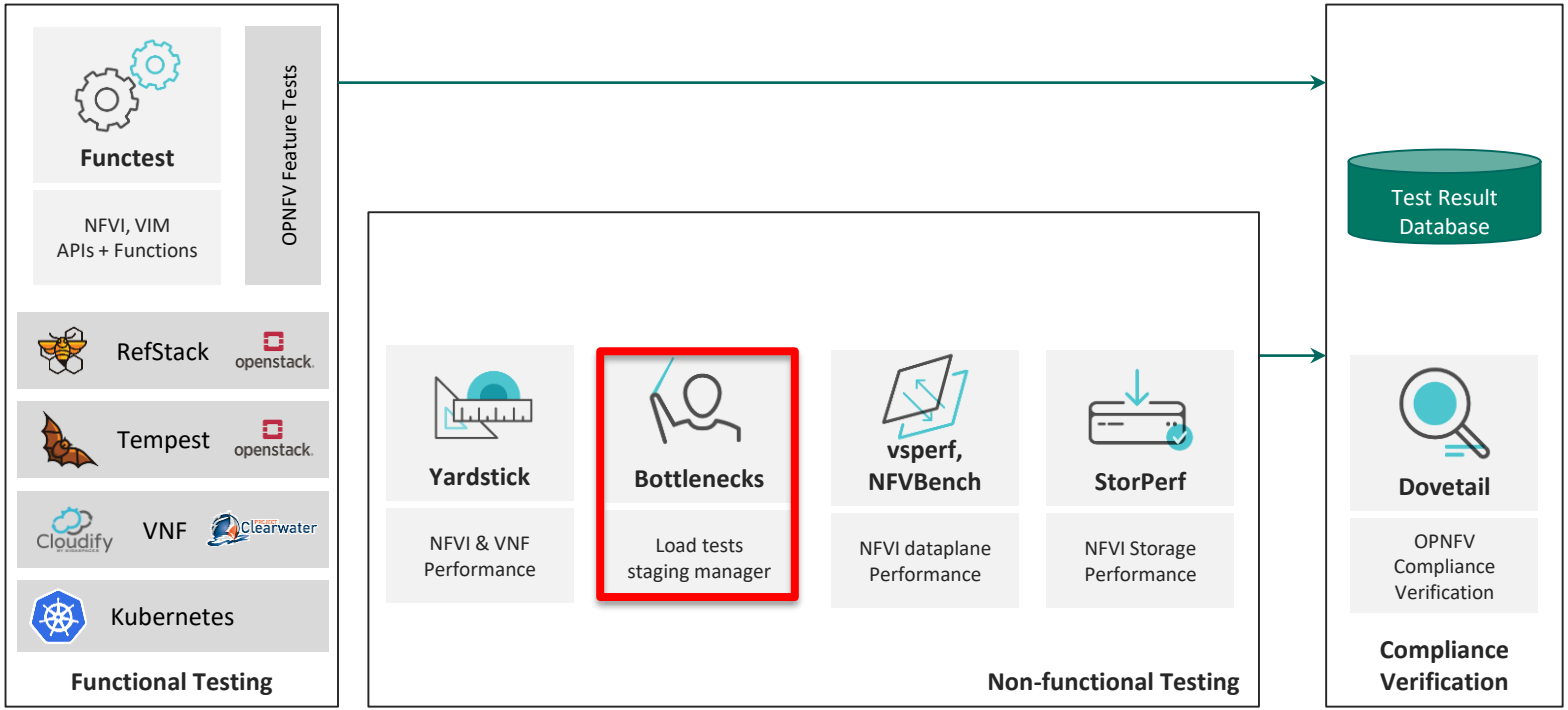


THE LINUX FOUNDATION

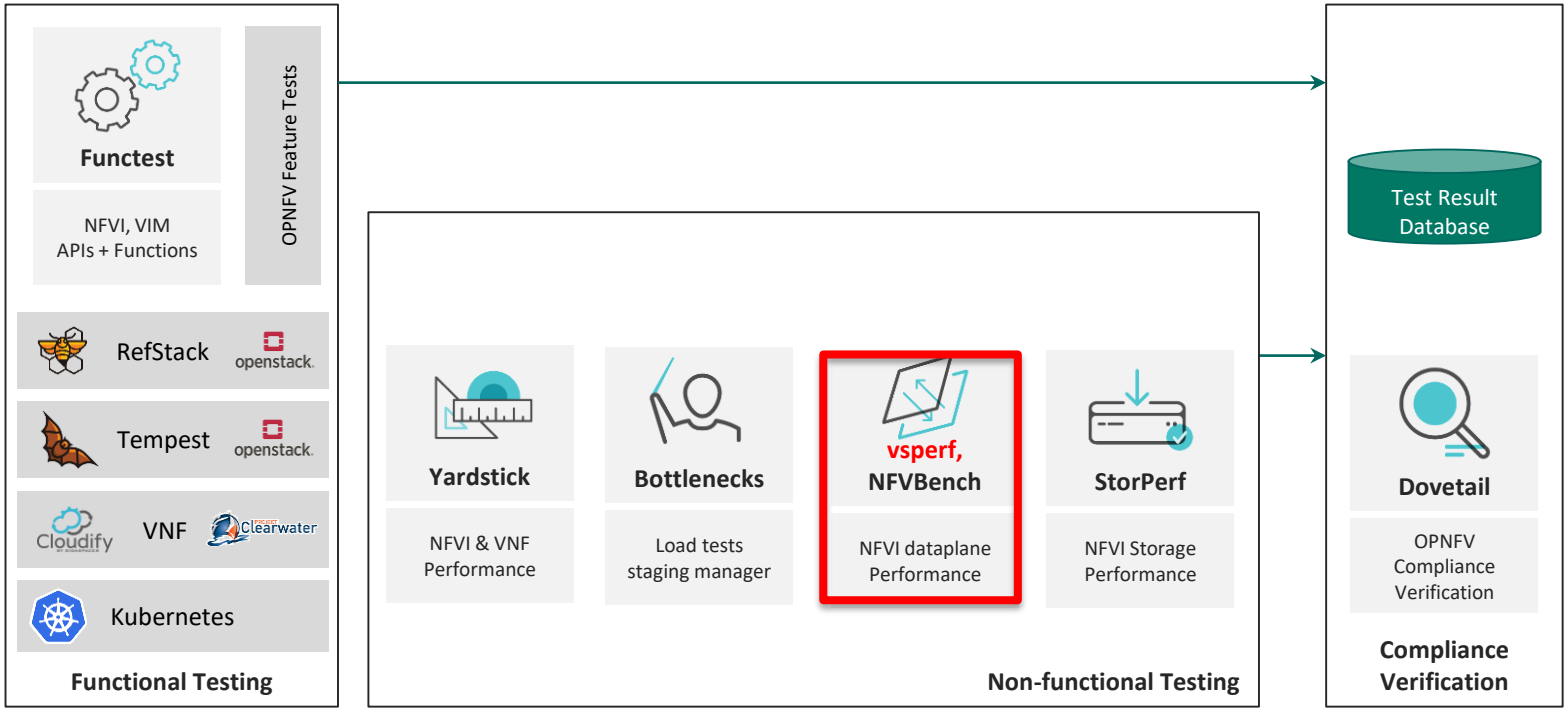
OPNFV Test Ecosystem



OPNFV Test Ecosystem

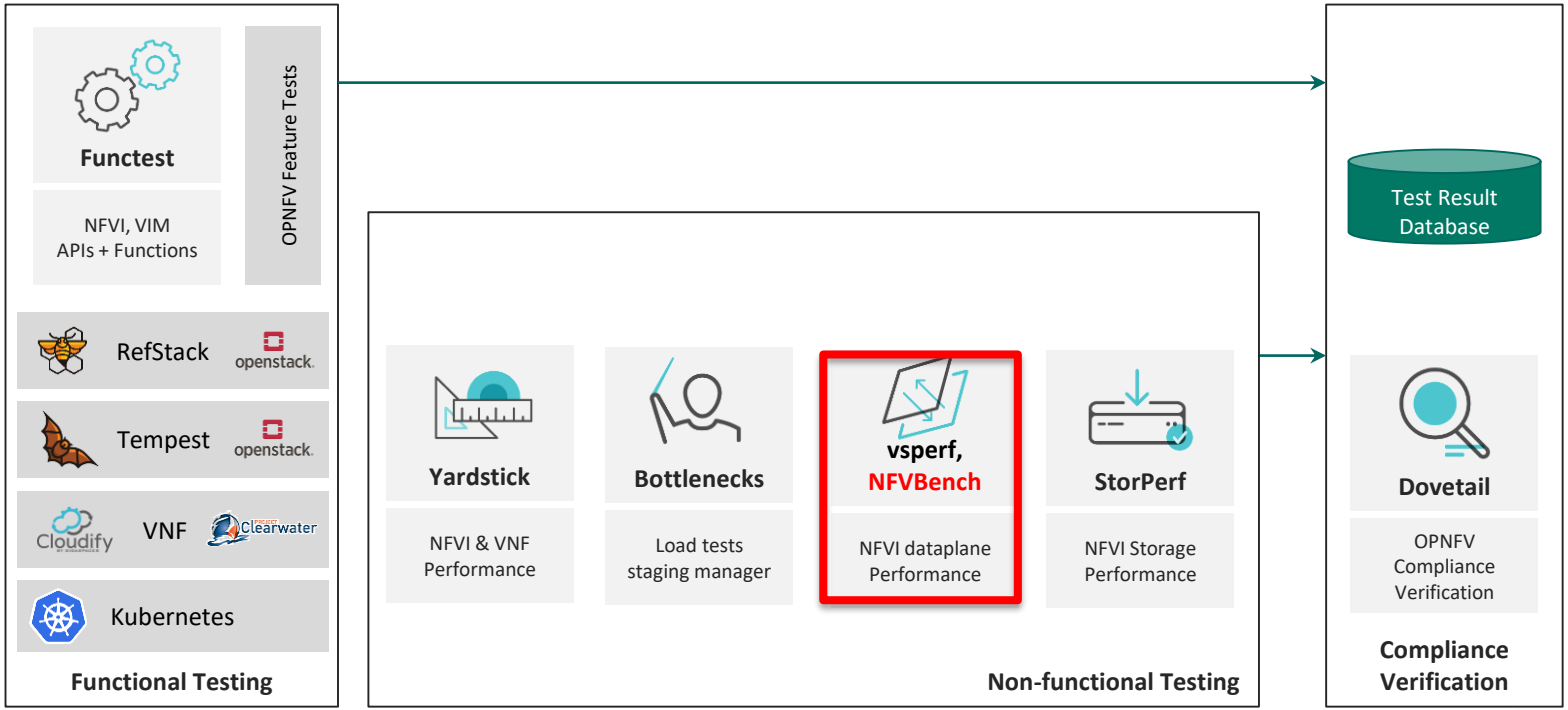


OPNFV Test Ecosystem



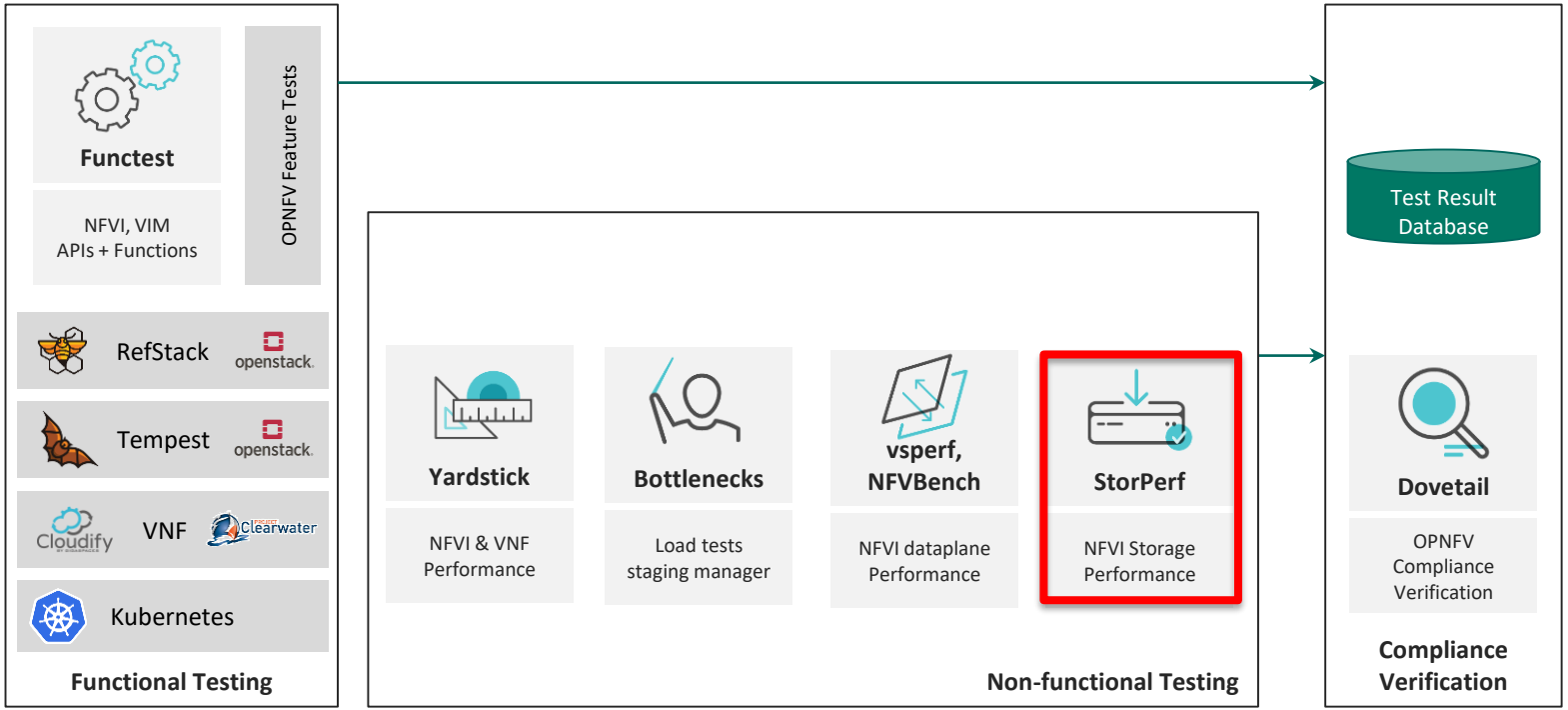
THE LINUX FOUNDATION

OPNFV Test Ecosystem



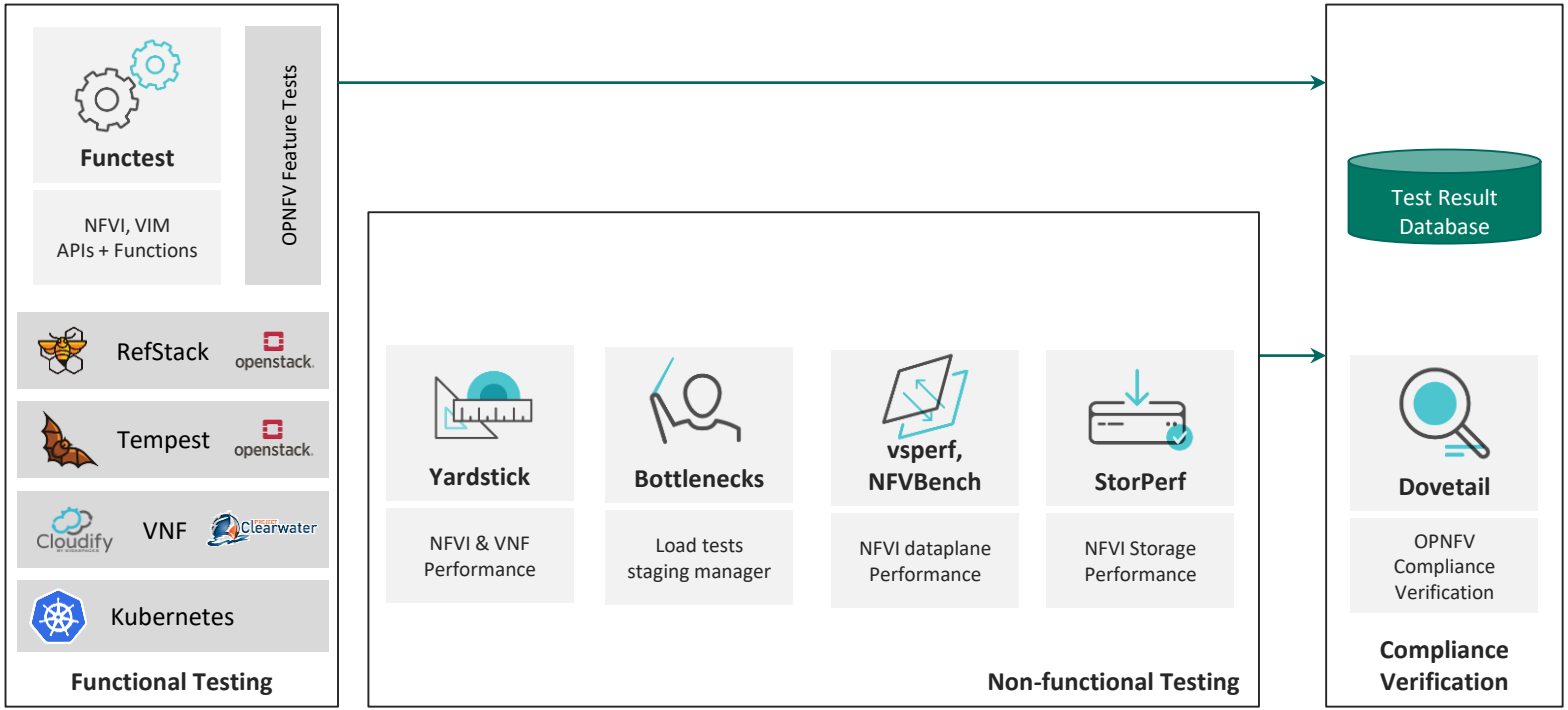
THE LINUX FOUNDATION

OPNFV Test Ecosystem



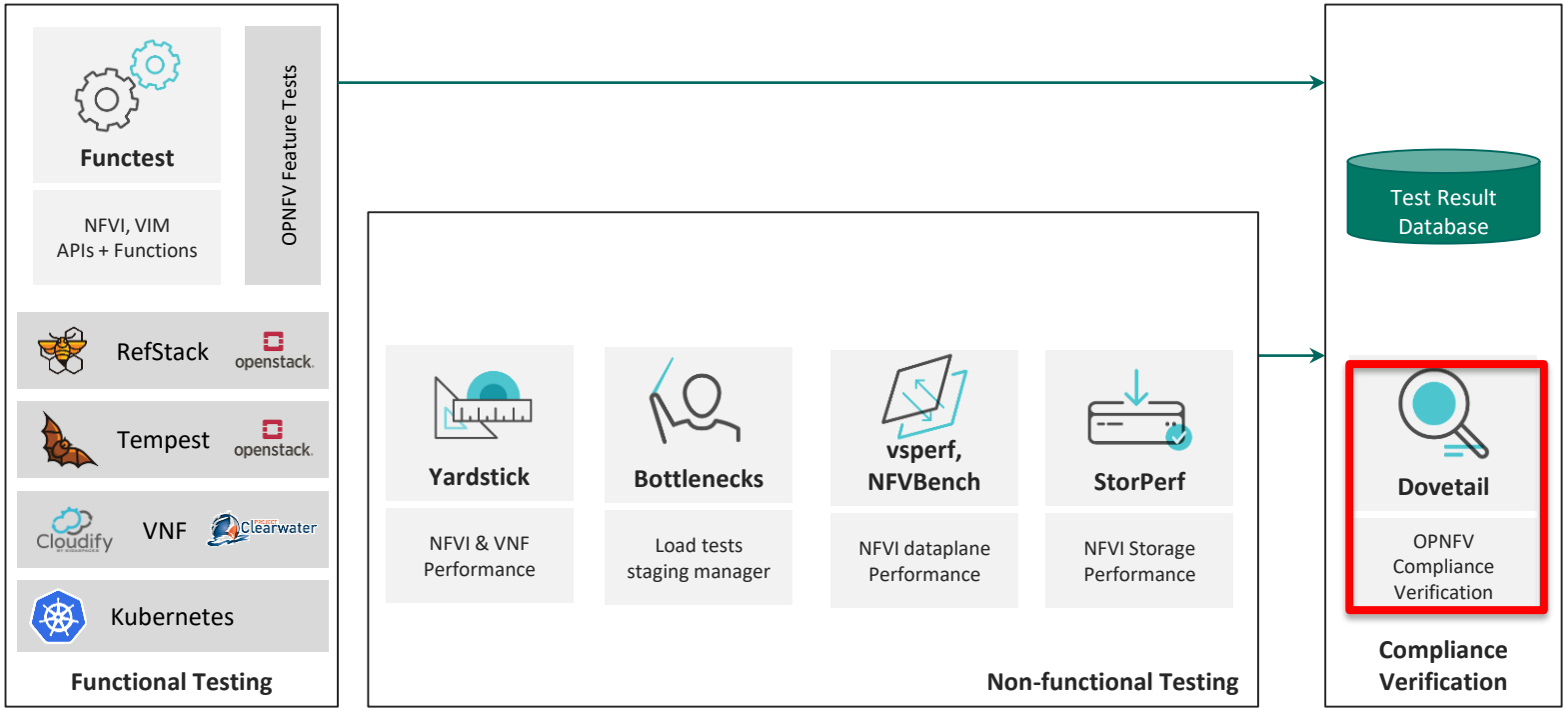
THE LINUX FOUNDATION

OPNFV Test Ecosystem



THE LINUX FOUNDATION

OPNFV Test Ecosystem



THE LINUX FOUNDATION

OPNFV Compliance Program

- OPNFV Verified Program (OVP) verifies that a commercial cloud platform exposes the same
  - key APIs,
  - behaviors, and
  - characteristics

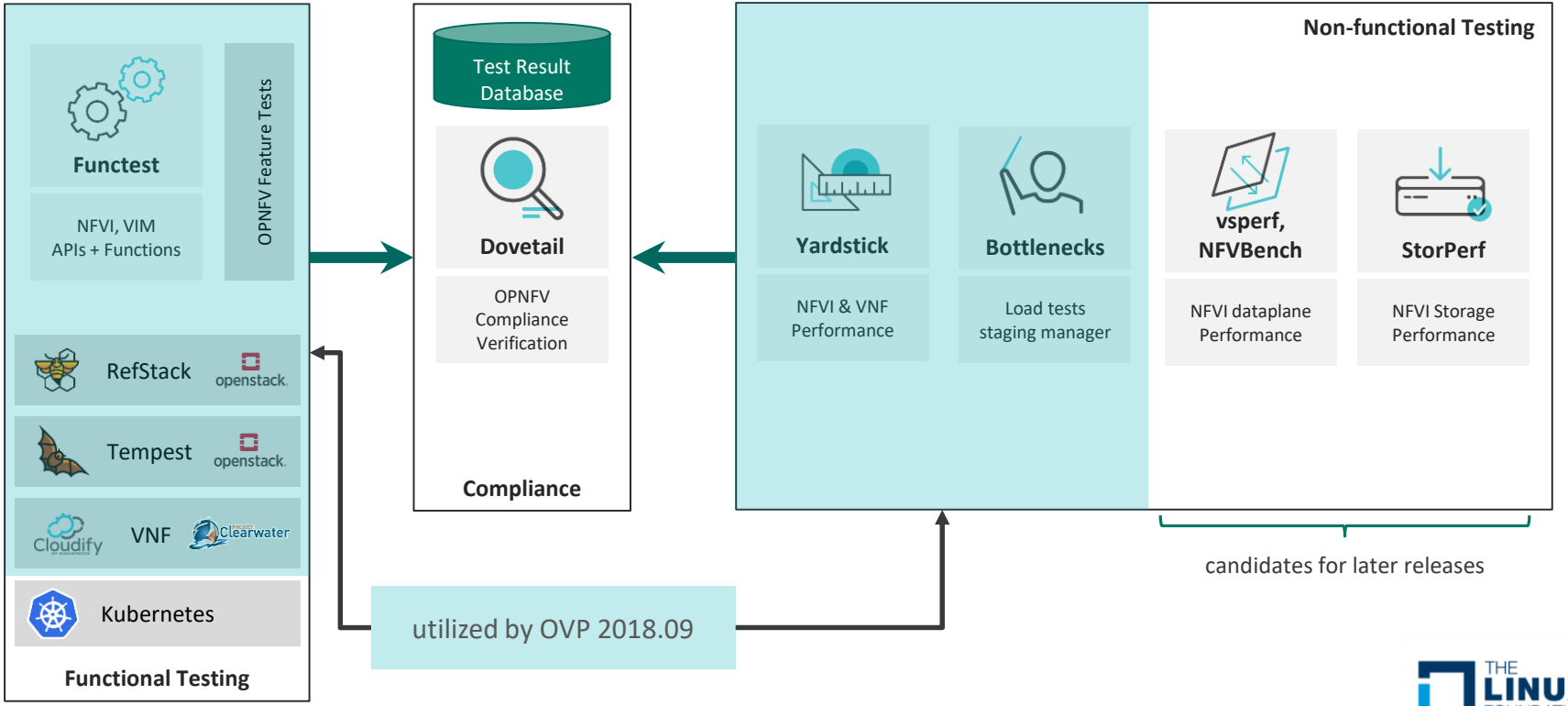
  as a reference platform **defined through a specific selection of test cases**

- 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

THE LINUX FOUNDATION

OPNFV Compliance Program



THE LINUX FOUNDATION

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

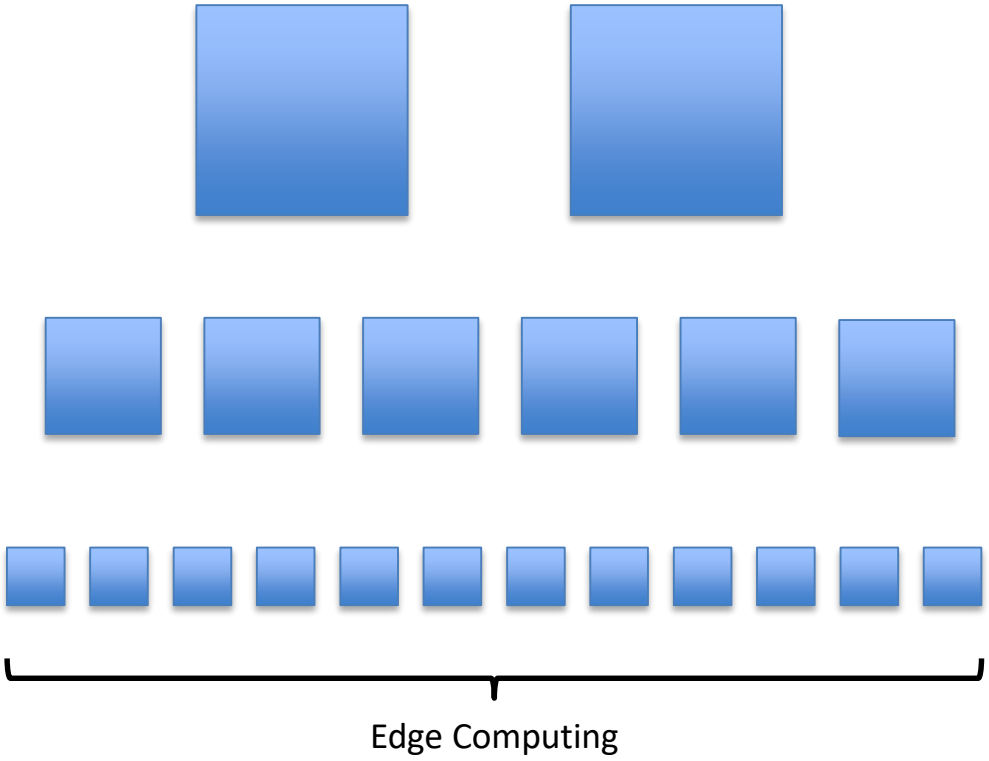
- Emerging use cases impose new requirements on test tools
  - Edge computing
  - Cloud native computing

⇨ 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
- …



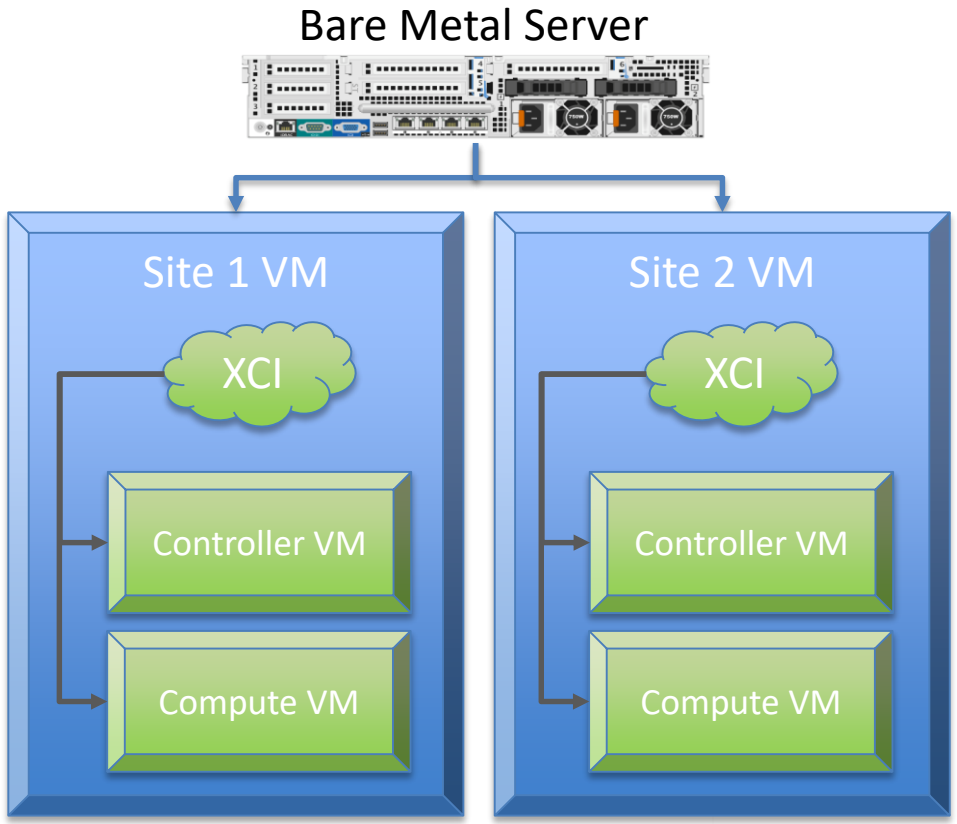
THE LINUX FOUNDATION

Edge Computing

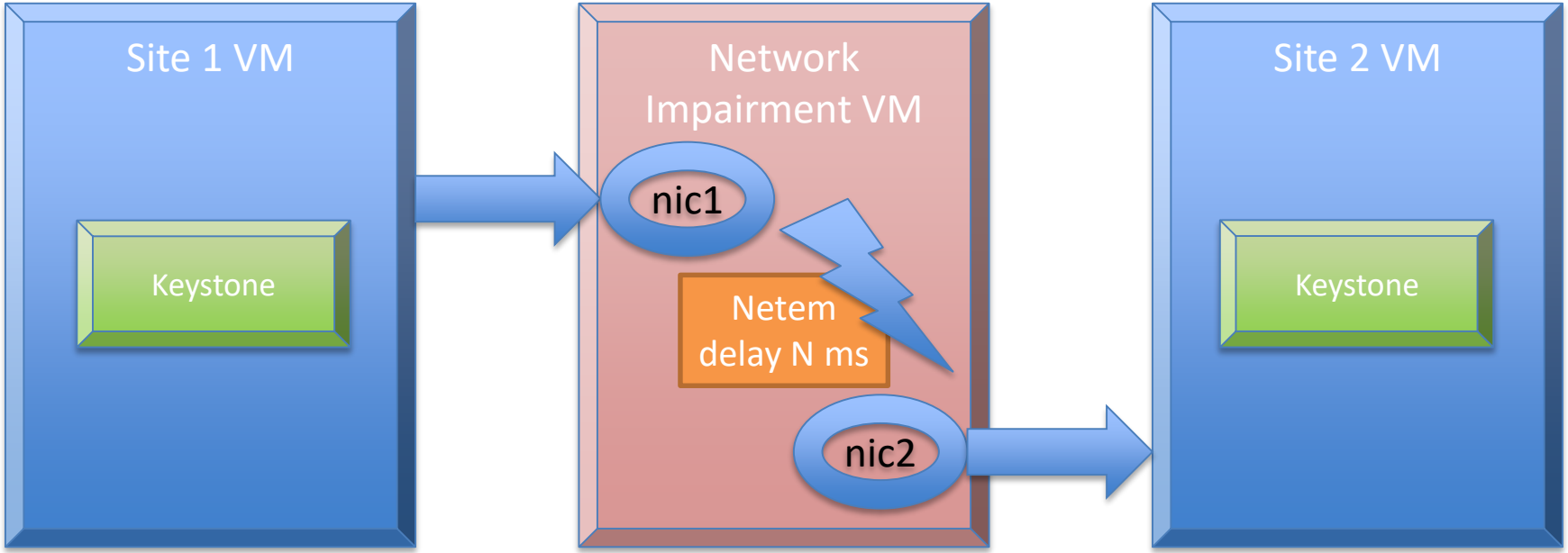
- Impact of edge computing on test tools and methods
  - Test topology
    - Automatic deployment of multiple sites
    - Inter-site connectivity
  - Consideration of networking effects
    - Control and data plane latency
    - Limited bandwidth, jitter, packet drops
    - Inter-site connectivity
 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

Virtual Edge in a Box

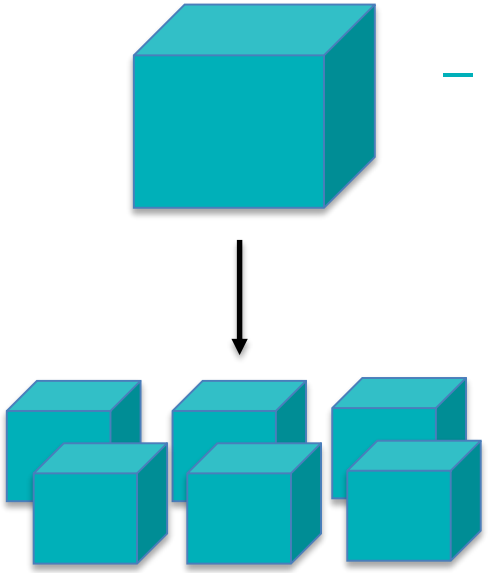
- OPNFV XCI
  - Mini flavor installs OpenStack from master in VMs
  - Can itself be in a VM
  - 2 full OpenStack environments in 1 server



Modeling of Edge Networking Environment



Cloud Native Computing



– Monolithic App

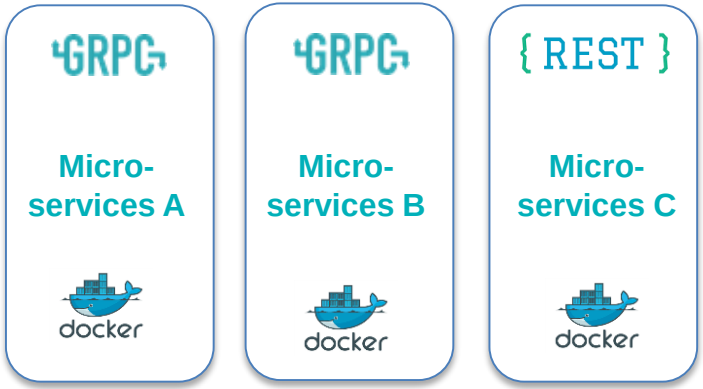
– Break down into smaller chunks

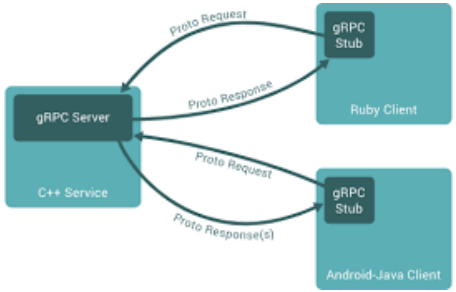
- Microservice architecture puts functionality into separate services:
  - Iterative development
  - Division of labor
  - Reduce single point of failure
  - Language/deployment flexibility
  - Build different apps using subsets of services

Micro-service Instrumentation

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

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



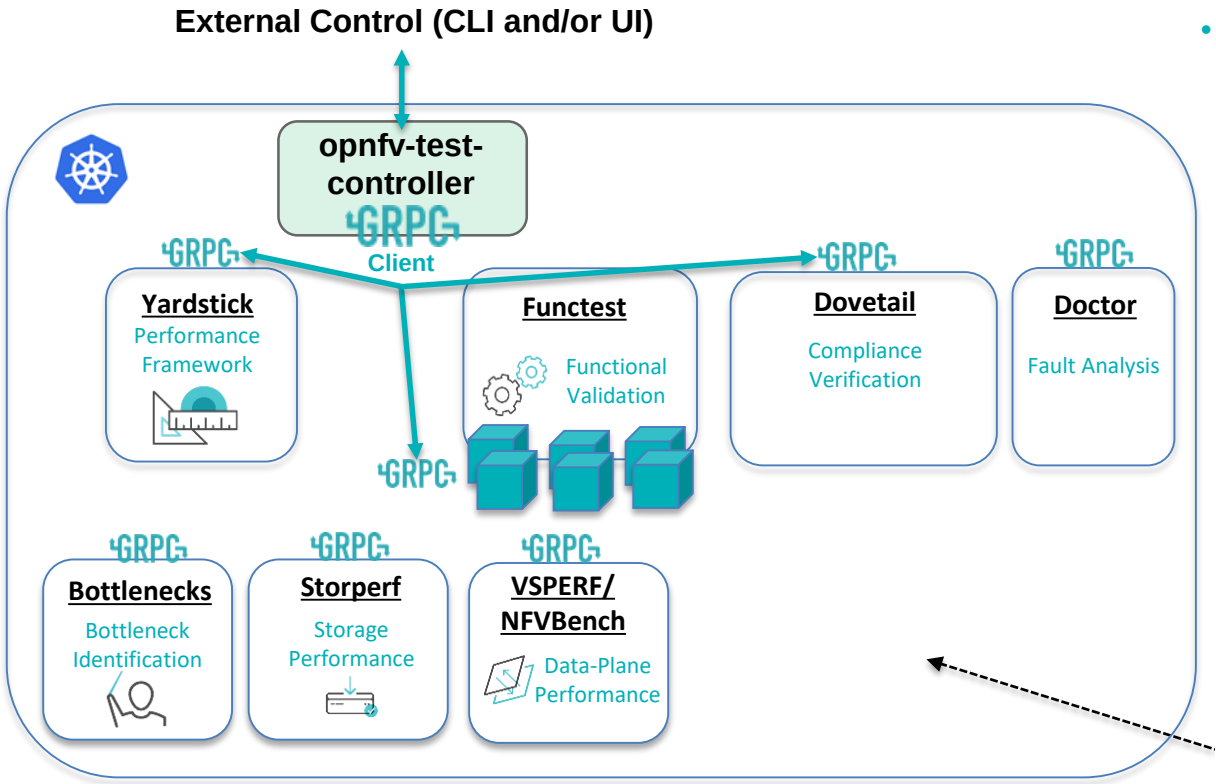


- Shared Data Stores
  - Exchange network data, state management

redis

THE LINUX FOUNDATION

Cloud Native & OPNFV Test Projects



- **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

Spinnaker

THE LINUX FOUNDATION

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!

THE LINUX FOUNDATION

THE LINUX FOUNDATION

OPEN SOURCE SUMMIT

EUROPE

THE LINUX FOUNDATION

Questions

opnfv-users@lists.opnfv.org

#functest

#yardstick #nsb

#bottlenecks

#nfvbench #vsperf

#dovetail

Backup

Test tools in more detail

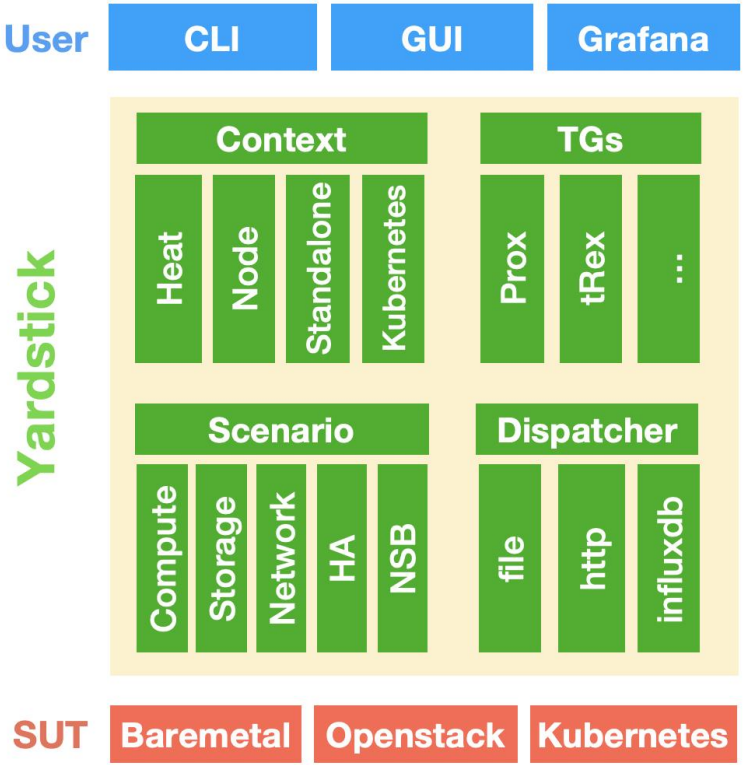
Functest in a nutshell

- 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)

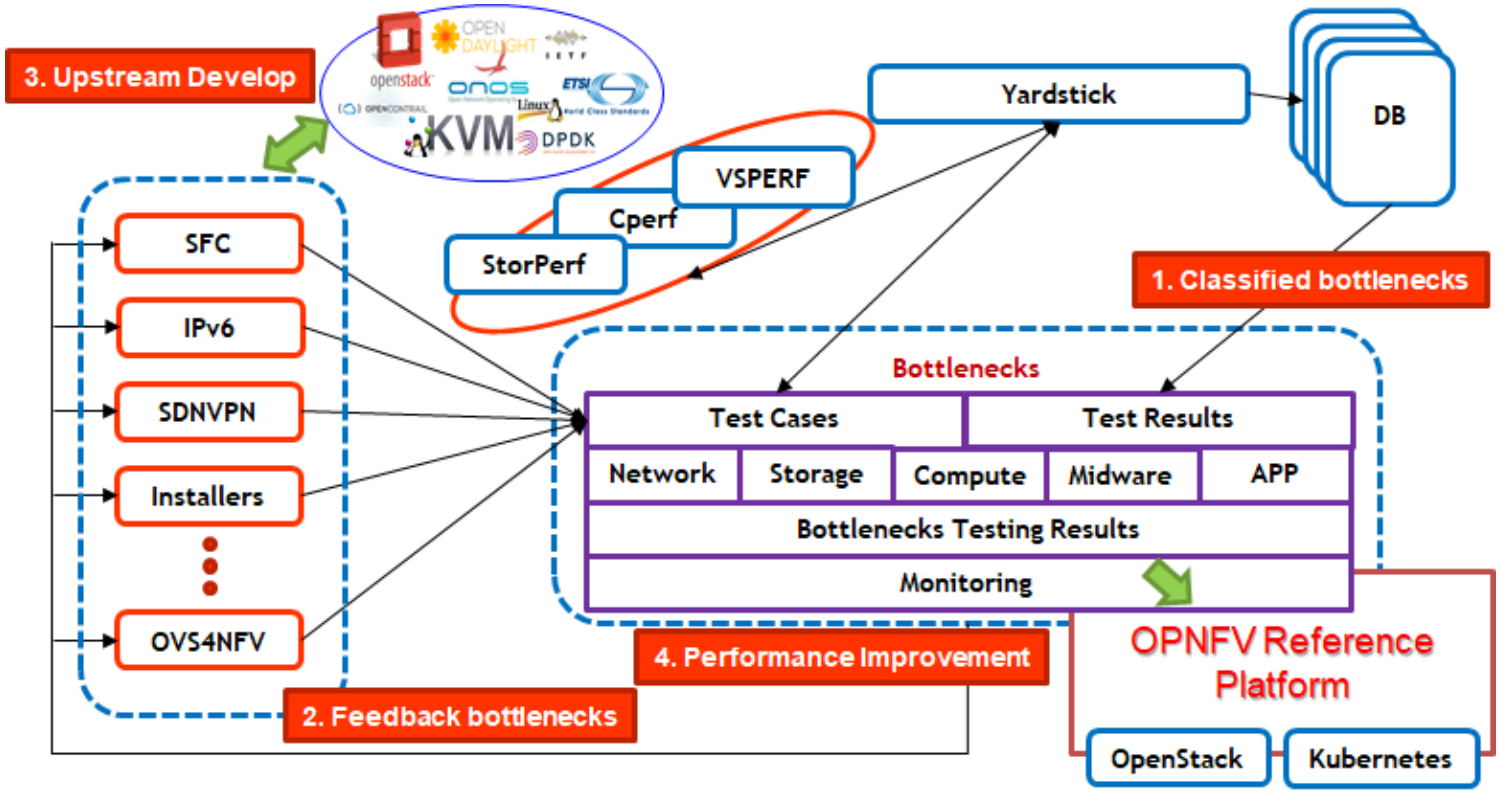
Yardstick



- 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).

THE LINUX FOUNDATION

Bottlenecks

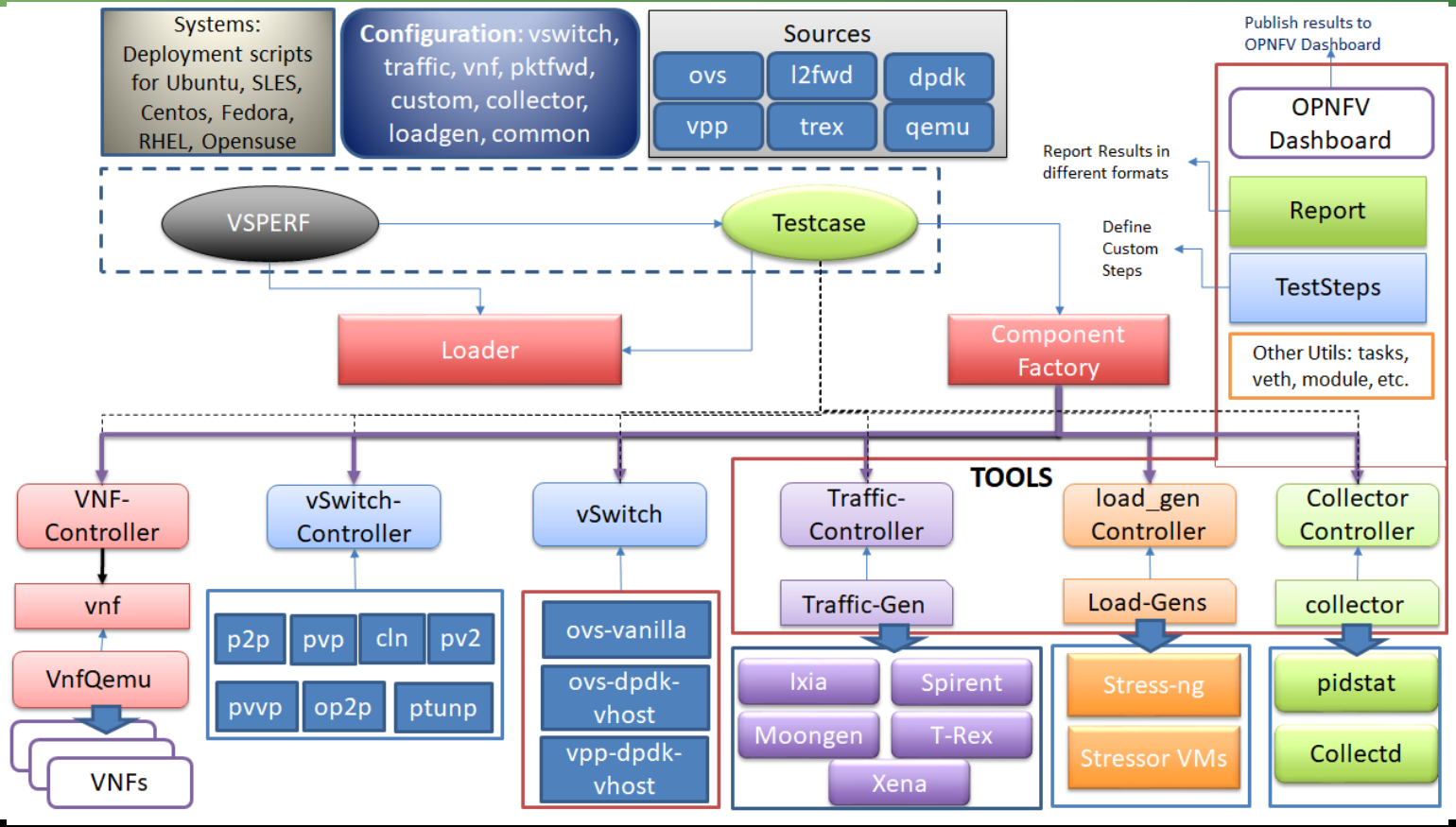


THE LINUX FOUNDATION

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



THE LINUX FOUNDATION

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.