Bottlenecks:
  Automated Design/Configuration
Evaluation
Background

• **Challenges**
  › It is difficult to find the system bottlenecks in terms of software and hardware.
  › Many of systems go through a detailed staging process that is mostly manual, complex and time-consuming.
  › During the staging process the system to be produced is subjected to workloads to determine whether it will meet the production workloads.
  › Finally, data gleaned from the staging process can be re-used to guide future designs and for management of system during operations.
  › Before submitting the infrastructure to production environment, it is needed to test and verify the infrastructure.

• **Current approaches**
  › The real workload to the deployed system is analyzed on-line and corresponding measurements are taken.
motivation

- Try to find system bottlenecks by testing and verifying OPNFV infrastructure in a staging environment before committing it to a production environment

- to have an automatic method for executing a benchmark on the deployment plan to validate the deployment during staging, instead of debugging a deployment during production use
Approach

- Create a powerful staging framework
- Automatically generate the full set of experimental specification and code
- Measure the performance of standard benchmarks over a wide range of hardware and software configurations
• Workload generator and VNFs (WV) : workload generator generates workloads which go through VNFs
• Monitor and Analysis (MA) : monitor VNFs status and infrastructure status to output analyzed results
• Deployment and Configuration (DC) : deploy and configure infrastructure and WV
• Automated Staging (AS) : implement automated staging
stages

Automated staging

Resource assignment

Redesign and Reconfiguration

Code generation and staging deployment

Automated Iterative Staging

Analyzer

policies

results

Analyzer

OpenStack

Workload generator VNF(a) VNF(b) VNF(c)

hypervisor ODL DAPP

infrastructure

WV

Automated staging*

Code generation and staging deployment

Redesign and Reconfiguration

Resource assignment

Automated staging*

Analyzer

policies

results

Workload generator VNF(a) VNF(b) VNF(c)

hypervisor ODL DAPP

infrastructure
Scope

- Provide framework, methods, codes and test cases to test and verify all kinds of bottlenecks in infrastructure
Use case

- Software bottlenecks
- Hardware bottlenecks
- ............
Dependencies

- Installers in “BGS” provide the framework foundation to be tested in Infrastructure layer.
- Octopus provides the continuous integration test.
- Bottlenecks will consider the outputs of “Yardstick”, “Funtest”, “VSPERF DPACC” and “Q-Tip”.
- Configuration methods of the upstream software, such as .conf, json files
Related projects

- **VSPERF**: Virtual Switch Performance
- **Q-Tip**: performance characterization of NFVI bottom-up in white-box
- **DPACC**: Hardware-assisted Data Plane Acceleration
- **Functest**:
  - Rally Bench, Tempest, vPing, vIMS, SDN Controller suite, CI automation
- **Yardstick**: **NFVI verification from VNF perspective**
  - offers both functional and performance test cases addressing the whole system (where Rally for instance is more used for OpenStack and Robot used for ODL)
  - Good fault isolation to be able to identify/detect faults early (usecase: Fault injection)
- **BGS**: as installers
Planned deliveries

- Framework
- Test cases
- Diagrams showing the test results
- Reference documents
- BP and Codes for upstream such as ODL, KVM, Openstack
- .............
thanks