Tutorial: Take a Test Drive with the Cloud Native Network Function (CNF) Testbed

Sponsored by Cloud Native Computing Foundation

Presented by:
Taylor Carpenter, Vulk Coop
Lucina Stricko, Vulk Coop
Denver Williams, Vulk Coop
Michael Pedersen, Tieto/Intel
CNF Testbed Tutorial Goals

• To gain a shared understanding of:
  – How to set up a CNF Testbed workstation,
  – How to provision Packet machines,
  – How to create Kubernetes clusters,
  – How to deploy use cases,
  – How to stay connected with the CNF Testbed initiative
Agenda - 90 minutes

- Intro to CNF Testbed
- Overview of components + stages
- Pre-reqs + setup workstations
- Stage 1: Hardware provisioning
- Stage 2: Cluster provisioning
- Q/A + break - 15 minutes
- Use cases + examples
- Q/A + open help - 30 minutes
Please ask the CNF Testbed Team for workstation instructions
Get these slides at https://sched.co/ScCA
Live session instructions
Sponsored by:

CLOUD NATIVE COMPUTING FOUNDATION
Cloud Native Computing Foundation

• Nonprofit, part of the Linux Foundation; founded Dec 2015

Graduated

- Kubernetes: Orchestration
- Prometheus: Monitoring
- Envoy: Network Proxy
- CoreDNS: Service Discovery
- Fluentd: Logging

Incubating

- Opentracing: Distributed Tracing API
- GRPC: Remote Procedure Call
- Helm: Package Management
- Linkerd: Service Mesh
- CNI: Networking API
- Jaeger: Distributed Tracing
- ROOK: Storage
- Harbor: Registry
- Etcd: Key/Value Store
- Kubebuilder: Open Policy Agent
- Cri-o: Container Runtime

Platinum members:

- Alibaba Cloud
- AWS
- Apple
- CISCO
- Dell Technologies
- Fujitsu
- Google Cloud
- IBM Cloud
- Intel
- JD.COM
- Microsoft Azure
- Oracle
- Pivotal
- Red Hat
- Samsung
- SAP
- VMware
Cloud Native Network Function (CNF)
Testbed Intro
CNF Testbed

- Open source initiative from CNCF
- Collaborating with CNCF Telecom User Group
- Testing and reviewing emerging cloud native technologies in the Telecom domain
- Funneling the new technology to early adopters
- Providing fully reproducible use cases and examples
- Running on top of on-demand hardware from the bare metal hosting company, Packet
CNF Testbed Contributors

Dan Kohn @dankohn
Ed Warnicke @edwarnicke
Taylor Carpenter @taylor
Denver Williams @denverwilliams
W.Watson @wavell
Lucina Stricko @lixuna

Michael S. Pedersen @michaelspedersen
Robert Starmer @robertstarmer
Fred Sharp @linkous8
Peter Mikus @rpmikus
Maciek Konstantynowicz @maciekatbgpnu
Nikolay Nikolaev @nickolaev
CNF Testbed Contributors

- Cisco
- Cloud Native Computing Foundation
- doc.ai
- F5
- Kumulus Technologies
- Intel
- Network Service Mesh
- Packet
- Red Hat
- VMware
- VULK COOP
We Welcome Your Participation

• Replicate our results from github.com/cncf/cnf-testbed with an API key from packet.com/cnf

• Package your internal network functions in containers (ideally following cloud native principles) and run on your instance of the testbed
  – We don’t need to see the code but would love to see the results

• Create pull requests to have the CNF Testbed run on your bare metal servers or other cloud bare metal servers like AWS i3.metal
Contribute Use Cases and Enhancements

- Contribute new use cases to the CNF Testbed ([issues](#) or [spec board](#))

- Create pull requests to improve Kubernetes or OpenStack deployments
Get Connected with the CNF Testbed

- Join the #cnf-testbed channel on CNCF slack
  - slack.cncf.io
- Subscribe to the CNCF Telecom User Group mailing list:
  - telecom-user-group@lists.cncf.io
- Attend CNCF Telecom User Group meetings:
  - https://github.com/cncf/telecom-user-group
  - 1st Mondays at 5pm CET / 8am Pacific Time (US & Canada)
  - 3rd Mondays at 1pm CET / 7pm China Standard Time
- Stay for the F2F Telecom User Group Meeting today:
  - 10:45am - 12:15pm in Darwin https://sched.co/Saoc
Review & Roadmap
Review of CNF Testbed v1 - It begins

- Initiative started at ONS NA 2018 in Los Angeles
  - Apples-to-apples comparison of CNFs and VNFs
  - What can we re-use from ONAP and other projects?
  - What gaps are missing on the path to cloud native?
  - What is a POC to assist with discussions?
ONAP Demo to Ansible-based v1 CNF Testbed

- Started with onap-demo
- Pivot to building blocks: Docker + Vagrant first
- Next: OpenStack and K8s workload platforms
- VPP based vSwitch for both platforms
- Ansible for additional hardware, host and network provisioning
- Custom use cases with Ansible, scripts and HEAT templates
## CNF Testbed Review of May to August 2019

<table>
<thead>
<tr>
<th>Month</th>
<th>Activities</th>
</tr>
</thead>
</table>
| May 2019   | - Containerized VPP vSwitch  
             - RDMA/non-proprietary Mellanox vSwitch  
             - Support unprivileged CNFs  
             - Maintenance: version updates and bug fixes  
             - Presentations at KubeCon/CnC, Fd.io, LFN CN Days |
| June 2019  | - Single-node IPsec use case  
             - Planned NSM use case and requirements  
             - Maintenance: version updates and bug fixes  
             - Intro + Deep Dive BoF at KubeCon China |
| July 2019  | - Deutsche Telekom reproduced CNF Testbed  
             - Successfully tested the Intel container kit on Packet  
             - WIP Support for NSM use cases |
| Aug 2019   | - NSM Packet filter use case  
             - WIP NSM Physical NIC Gateway use case  
             - Containerized VPP vSwitch on Mellanox  
             - Maintenance: Fix OpenStack deploy, updates, bug fixes |
General goals - technology innovation review tool

- Support changing and trying different technology options
- Keep things as simple as reasonable
- Use upstream community tooling
- Target cloud native principles...
Target cloud native principles

• Where possible use cloud native principles for all levels (hardware to use case)
  – Immutable hardware
  – Version control all configuration including underlay networking
  – Workload bootstrapping repeatable by automation/pipeline
• Highlight where gaps are missing and out-of-band procedures are used
• Bring focus to technology which is attempting to provide solutions to meet cloud native principles
Quick Look of CNF Testbed v2

• Key features of CNF Testbed v2
  – using more in-band components
    • refactor using Helm for K8s use cases
    • replace cross-cloud provisioner with Terraform + Kubespray
    • more K8s-native replacements for out-of-band host setup
  – adding support for NSM, DANM, Intel device plugins
  – adding new use cases:
    • Hybrid K8s + OpenStack service chains
    • SR-IOV
  – example workload configs (eg. Nokia CPU Pooler + NSM)
## CNF Testbed Roadmap (Oct 2019 to Jan 2020)

| Oct 2019 | NSM IPSec single-node use case  
|          | NSM 2-node IPsec use case  
|          | NSM IPFwd Service Chain benchmark test  
|          | Separate hardware and workload provisioning stages + Kubespray for K8s | [TBD] |
| Nov 2019 | NSM Hybrid K8s+Openstack use case  
|          | DANM SR-IOV use case | NSMCon, KubeCon NA (Nov 18-21) |
| Dec 2019 | NSM SR-IOV Use Case  
|          | Intel Multus + CPU Manager use case  
|          | TBD: Kolla/Openstack-helm (TBD) | [TBD] |
| Jan 2020 | GSM/5G GW use case with NSM | [TBD] |
Overview of Components & Stages
Components of the CNF Testbed

- Hardware provisioning
- Workload provisioning (eg. K8s or OpenStack)
- Use Cases and Examples
- Network Functions (eg. Packet Filter, NIC Gateway)
- Testing tools (eg. NFVbench)
CNF Testbed Software components

**OpenStack compute**
- VPP IP Router
- vhost-user
- QEMU/KVM
- Kernel 4.4.0-134

**OpenStack controller**
- OS “rocky” services
- Neutron, API
- etcd
- Kernel 4.4.0-134

**Kubernetes master**
- K8s v1.12.2
- VPP vSwitch
- Kernel 4.4.0-134

**Kubernetes worker**
- VPP IP Router
- VPP vSwitch
- Kernel 4.4.0-134

**Traffic generator**
- NFVbench
- TRex
- Docker
- DPDK

**Packet.net router**
- Packet API
Workstation Set-up
Check-in

• Who has gained access to the workstation?
Live session instructions
Steps to Deploy the CNF Testbed
Steps to deploy the CNF Testbed

• Current v1 vs future v2
• Tutorial workstations
• How to set up a CNF Testbed environment:

• Deployed on servers hosted by: 

  packet
Pre-reqs to setting up the workstation

• Access to a project on Packet
  – Note the PROJECT_NAME and PROJECT_ID, both found through the Packet web portal, as these will be used throughout the deployment for provisioning servers and configuring the network.

• An available keypair for SSH access
  – Add your public key to the project on Packet through the web portal, which ensures that you will have passwordless SSH access to all servers used for deploying the CNF Testbed.
Set-up walkthrough

• Install the initial dependencies
  – Install base tools such as git, curl
  – Install Docker
  – Install Kubectl
  – Clone CNF Testbed
Set-up walkthrough

• Create a keypair on the workstation
  – Add this key to the project on Packet
• DNS setup for K8s cluster
• Create cluster configuration
Deploy Ansible environment

• Certain parts of the CNF Testbed are done directly using Ansible playbooks. The easiest way to run these is to set up an interactive container on the workstation server using "cnfdeploytools", which has been built previously.

• This container environment is not used for deploying the K8s clusters. When the environment is needed it will be mentioned (deploying packet generator and CNFs).
WIP - Hardware & Cluster Provisioning
Changing how to provision hardware and clusters

• Moving off “cross-cloud” custom provisioner, one stage for:
  – Hardware
  – Kubernetes clusters

• Moving to a two-stage process, using:

Terraform and Kubespray
Stage 1: Hardware Provisioning
Overview of hardware provisioning stage for v1

- Terraform is used to provision the Packet machines
- Ansible is used for additional host configuration and network underlay provisioning
- The K8s provisioning is tightly coupled to hardware provisioning
- OpenStack Chef is started via a Terraform Ansible plugin

Note: Tutorial clusters have been pre-provisioned
Overview of v2 hardware provisioning

- Loosely coupled independent stage
- Continuing to use Terraform and Ansible
- Support re-using existing machines with a reset
- Machines can be used for any purpose

Note: Tutorial clusters have been pre-provisioned
Stage 2: Workload or Cluster Provisioning
Overview of workload provisioning v1

- Cloud-init bootstrapped K8s cluster using cross-cloud
- Ansible for additional host provisioning including underlay networking
- Ansible for host vSwitch
- OpenStack Chef
Overview of workload provisioning v2

- Provisioner supporting Kubespray
- Integrate current Ansible host and Packet network provisioning
- Add support for K8s-native options where possible
- Move to Kolla or OpenStack-helm
Q/A & Break: 15 minutes
Any questions?
Use Cases & Examples
Overview of use cases

- Structure
- Purpose
- Different / multiple examples
- Different implementations
  - Out of band
  - Multus (https://github.com/intel/multus-cni)
  - DANM (https://github.com/nokia/danm)
  - Network Service Mesh (https://networkservicemesh.io)
Use case #1 - NSM single node packet filter

- Github [examples/use_case/packet-filtering-on-k8s-nsm-on-packet](https://github.com/cloud-native-computing/examples/use_case/packet-filtering-on-k8s-nsm-on-packet)
- Using NSM to connect two types clients through a packet filter network function
Deploying use case #1

- Pre-reqs
- Deployment of example
- Run tests for use case / example
Use case #2 - Physical NIC GW with shared access

- Client #2 (vEth/Tap)
- Client #1 (Memif)
- Packet filter CNF
- Physical NIC Gateway CNF

Packet VLAN with internal network (eg. 10.0.0.5/24)

Test end point
Use case #2 - Physical NIC GW with shared access

- Github use_case/external-packet-filtering-on-k8s-nsm-on-packet
- Physical NIC GW network function
- Multiple service chains with private networks
- DPDK + VPP-based access to Packet’s Intel x710 NIC
  - n2.xlarge machine type
  - host provisioning required
  - privileged GW container
Deploying use case #2

• Pre-reqs
  – NSM, GW, External end-point
• Deployment of example
  – Multiple PF Service chains
• Run tests for use case / example
Q/A & Open Help: 30mins
Questions? Concerns?
Thank you

• Today’s tutorial presented by:

• Today’s tutorial sponsored by:

CLOUD NATIVE COMPUTING FOUNDATION
Thank you for your participation!