KubeCon | CloudNativeCon

North America 2018

Intro: Cloud Native Network Functions (CNF) BoF

Dan Kohn, Executive Director, CNCF

CNFs vs. VNFs

Dan Kohn
Executive Director, CNCF

CLOUD NATIVE COMPUTING FOUNDATION

Cloud Native Computing Foundation

- Non-profit, part of the Linux Foundation; founded Dec 2015

Graduated

- kubernetes — Orchestration
- Prometheus — Monitoring
- envoy — Service Proxy

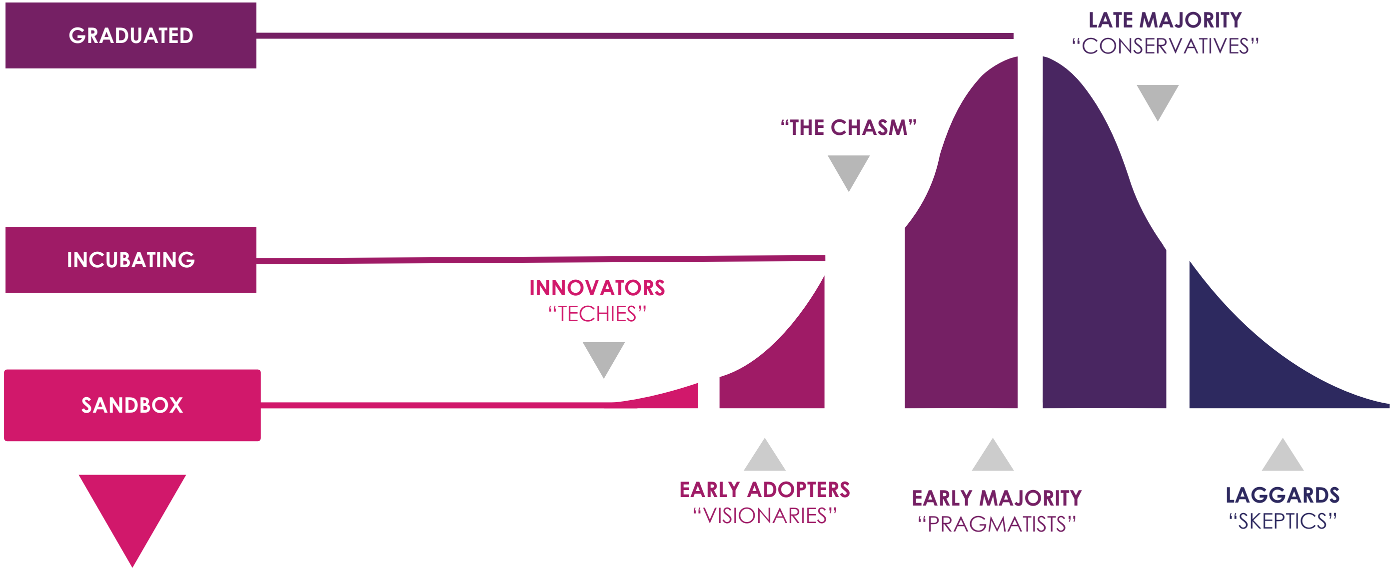
Incubating

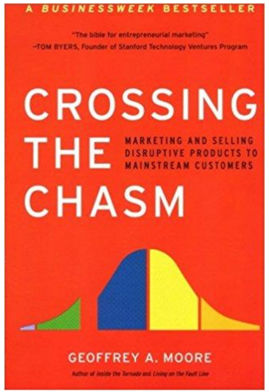
- OPENTRACING — Distributed Tracing API
- fluentd — Logging
- GRPC — Remote Procedure Call
- containerd — Container Runtime
- rkt — Container Runtime
- CNI — Networking API
- JAEGER — Distributed Tracing
- TUF — Software Update Spec
- Notary — Security
- Vitess — Storage
- CoreDNS — Service Discovery
- NATS — Messaging
- LINKERD — Service Mesh
- HELM — Package Management
- ROOK — Storage
- HARBOR — Registry
- etcd — Key/Value Store

- Platinum members:

Alibaba Cloud, aws, cisco, DELL Technologies, docker, FUJITSU, Google Cloud, HUAWEI, IBM Cloud, intel, JD.COM, MESOSPHERE, Microsoft Azure, ORACLE, Pivotal, redhat, SAMSUNG SDS, SAP, vmware

CNCF Project Maturities





SANDBOX

spiffe	SPIRE	Open Policy Agent	cloudevents	TELEPRESENCE	OPENMETRICS	TiKV	cortex	Buildpacks.io	Falco	Dragonfly	Virtual Kubelet
Identity Spec	Identity	Policy	Serverless	Tooling	Metrics Spec	Key/Value Store	Monitoring	Packaging Spec	Container Security	Image Distribution	Nodeless

TODAY THE LINUX FOUNDATION IS MUCH MORE THAN LINUX

Security	Networking	Cloud	Automotive	Blockchain	Web
We are helping global privacy and security through a program to encrypt the entire internet.	We are creating ecosystems around networking to improve agility in the evolving software-defined datacenter.	We are creating a portability layer for the cloud, driving de facto standards and developing the orchestration layer for all clouds.	We are creating the platform for infotainment in the auto industry that can be expanded into instrument clusters and telematics systems.	We are creating a permanent, secure distributed ledger that makes it easier to create cost-efficient, decentralized business networks.	We are providing the application development framework for next generation web, mobile, serverless, and IoT applications.
Let's Encrypt	LF NETWORKING	CLOUD NATIVE COMPUTING FOUNDATION	AUTOMOTIVE GRADE LINUX	HYPERLEDGER	node JS

We are regularly adding projects; for the most up-to-date listing of all projects visit tlfprojects.org

KubeCon + CloudNativeCon

- Europe 2019 (sponsorships and CFP open)
  - Barcelona: May 20-23, 2019
- China 2019 (sponsorships open)
  - Shanghai: June 24-26, 2019
- North America 2019 (sponsorships open)
  - San Diego: November 18-21, 2019

KubeCon CloudNativeCon
Europe 2019

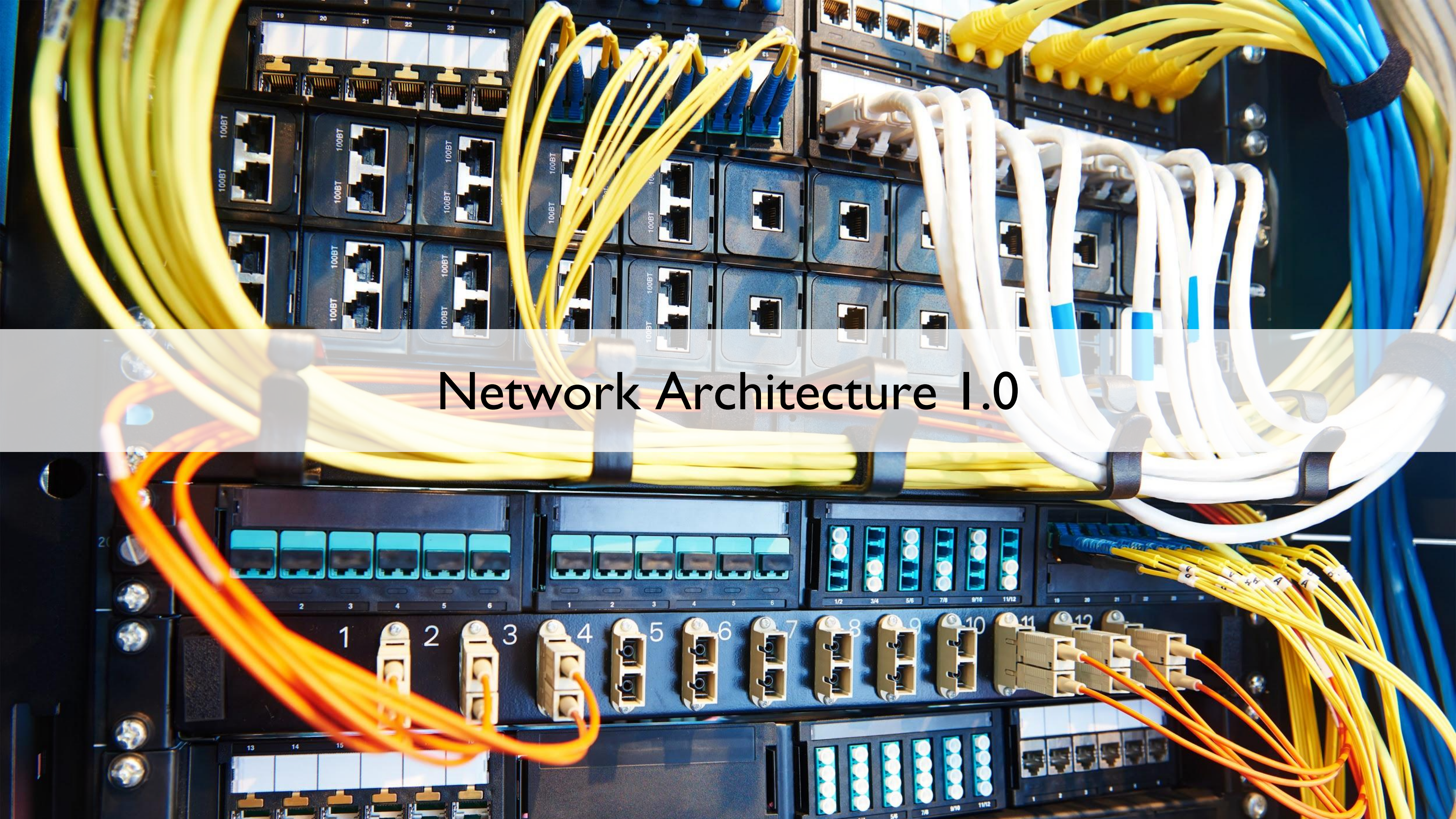
KubeCon CloudNativeCon
China 2019

KubeCon CloudNativeCon
North America 2019

KubeCon + CloudNativeCon Attendance

Network Architecture Evolution

- 1.0: Separate physical boxes for each component (e.g., routers, switches, firewalls)



Network Architecture 1.0

Network Architecture Evolution

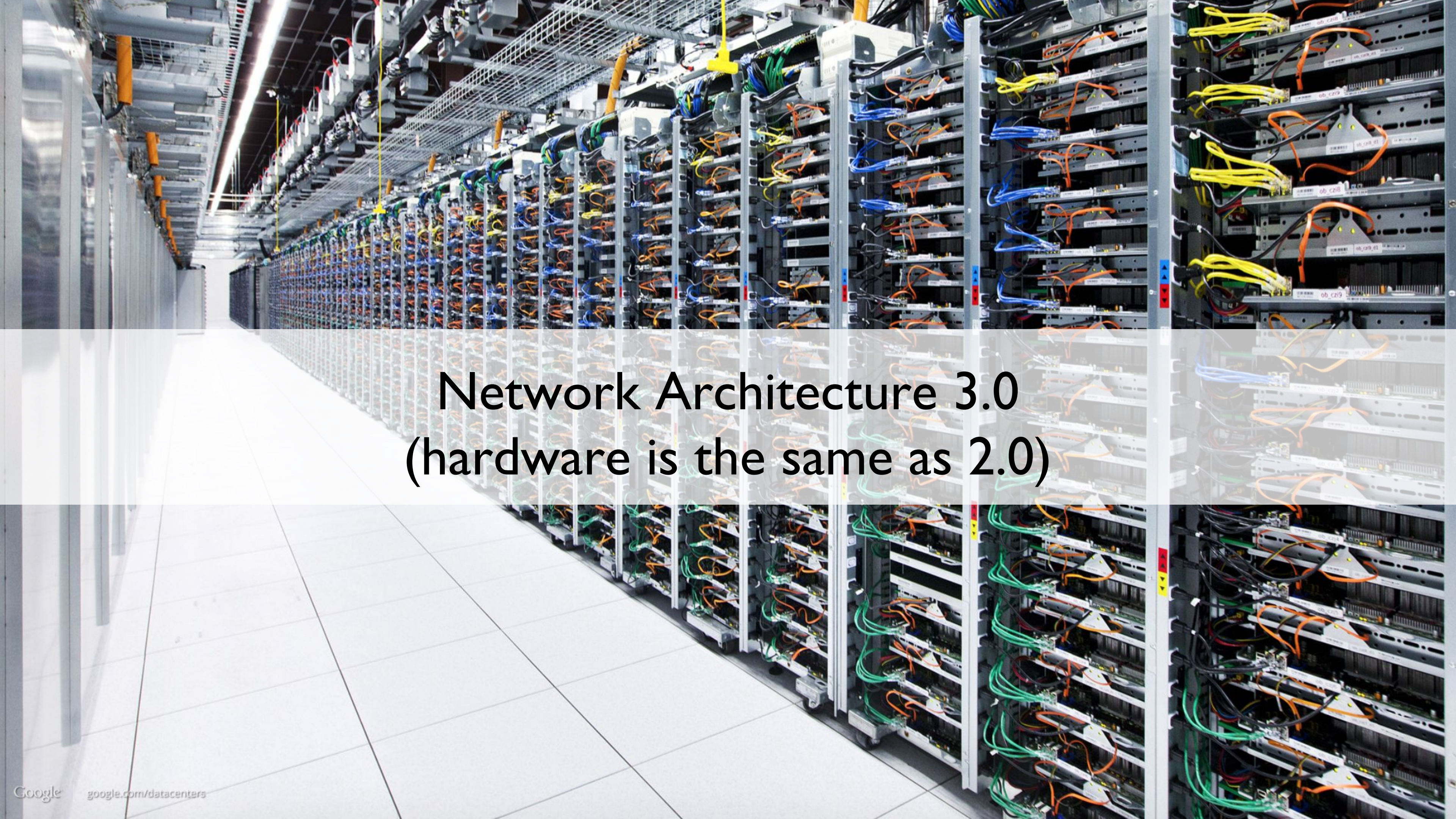
- 1.0: Separate physical boxes for each component (e.g., routers, switches, firewalls)
- 2.0: Physical boxes converted to virtual machines called Virtual Network Functions (VNFs) running on VMware or OpenStack

Network Architecture 2.0

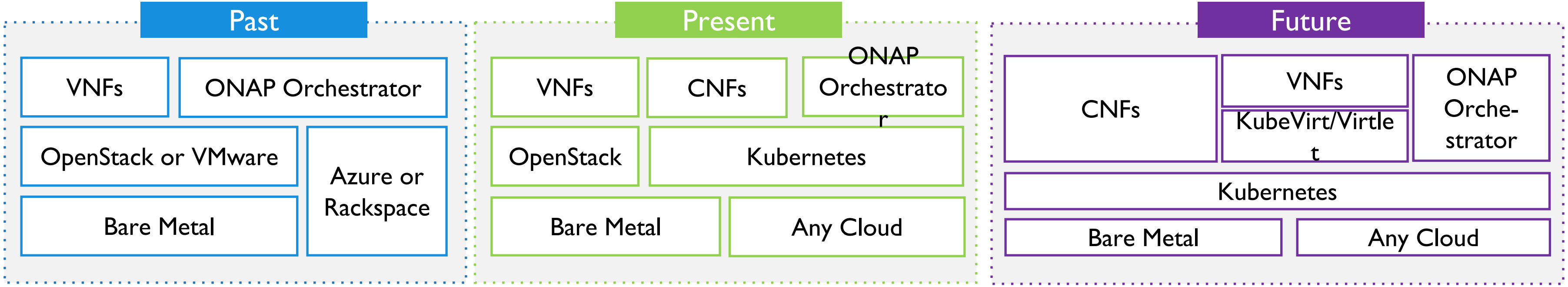
Network Architecture Evolution

- 1.0: Separate physical boxes for each component (e.g., routers, switches, firewalls)
- 2.0: Physical boxes converted to virtual machines called Virtual Network Functions (VNFs) running on VMware or OpenStack
- 3.0: Cloud-native Network Functions (CNFs) running on Kubernetes on public, private, or hybrid clouds

Network Architecture 3.0
(hardware is the same as 2.0)



Evolving from VNFs to CNFs



- ONAP Amsterdam (Past) runs on OpenStack, VMware, Azure or Rackspace
- ONAP Casablanca (Present) runs on Kubernetes and so works on any public, private or hybrid cloud
- Virtual Network Functions (VNFs) are virtual machines that run on OpenStack or VMware, or can be run on K8s via KubeVirt or Virtlet

Major Benefits

1. Cost savings (with public, private, and hybrid clouds)
2. Development velocity
3. Resiliency (to failures of individual CNFs, machines, and even data centers)

The challenge of transitioning VNFs to CNFs

- Moving from network functionality from *physical* hardware to encapsulating the software in a *virtual* machine (P2V) is generally easier than *containerizing* the software (P2C or V2C)
- Many network function virtualization VMs rely on kernel hacks or otherwise do not restrict themselves to just the stable Linux kernel userspace ABI
  - They also often need to use DPDK or SR-IOV to achieve sufficient performance
- Containers provide nearly direct access to the hardware with little or no virtualization overhead
  - But they expect containerized applications to use the stable userspace Linux kernel ABI, not to bypass it

Areas for More Discussion

- The strength of no longer being locked into specific OSs
  - Any version of Linux >3.10 is acceptable
- Multi-interface pods vs. Network Service Mesh
- Complete parity for IPv6 functionality and dual-stack support in K8s
- Security, and specifically recommendations from Google and Jess that come into play when hosting untrusted, user-provided code
  - Possible use of isolation layers such as Firecracker, gVisor, or Kata
- Scheduling container workloads with network-related hardware constraints (similar to what's been done for GPUs)
  - Network-specific functionality like traffic-shaping

Testbed Plans Underway

- VNFs vs. CNFs
  - Working on a demo of boot-time and throughput of VNFs on OpenStack vs. CNFs on Kubernetes, where the networking code and underlying hardware is identical
  - Will deliver opens source installers and Helm charts
- Cloud-native Customer Premises Equipment (CCPE) Project
  - Modify the ONAP vCPE use case and VNF deployment to show VNF vs. CNF deployments of chained network functions

Roll-Out Plans

- Open Source Summit NA, Vancouver, August 28: Joint workshop by CNCF executive director Dan Kohn and LF Networking head Arpit Joshipura on Cloud-native Network Functions
- Open Network Summit Europe, Amsterdam, September 25: Marketing launch
- KubeCon + CloudNativeCon NA, Seattle, December 11: Planned demo
- Mobile World Congress, Barcelona, February 25: Major roll-out
- Ongoing close collaboration with LF Networking and specific carriers providing feedback (AT&T, Bell Canada, Vodafone, etc.)

KubeCon | CloudNativeCon

North America 2018

Intro: Cloud-native Network Function (CNFs) Project

Taylor Carpenter, Vulk Coop

Agenda

- Who, What, Why
- Neutral environment requirements
- Reproducible test comparisons
- CNF vs VNF comparison results
- Verifying the tests
- What’s next?
- Q/A

CNCF CNFs Contributors

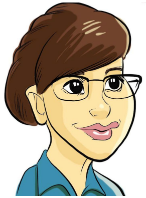
Dan Kohn
@dankohn

Ed Warnicke
@edwarnicke

Taylor Carpenter
@taylor

Denver Williams
@denverwilliams

W.Watson
@wavell



Lucina Stricko
@lixuna

Michael S. Pedersen
@michaels pedersen

Robert Starmer
@robertstarmer

Peter Mikus
@rpmikus

Maciek Konstantynowicz
@maciekatbgpnu

Ed Kern
@Snergster

Alec Hothan
@ahothan

Why Does the CNF Project Exist?

KubeCon | CloudNativeCon
North America 2018

CNCF is ushering the evolution of Virtualized Network Functions (VNFs) to Cloud-native Network Functions (CNFs) running on Kubernetes in public, private, or hybrid clouds.

The transition to CNFs will provide 3 major benefits to service providers:

1. Cost savings (capex/opex)
2. Improved resiliency
3. Higher development velocity

What Is the CNF Project?

The CNF project facilitates open collaboration on the development and use of Cloud-native Network Functions. The project creates reusable, open source, reference code and test comparisons of CNFs for real world use cases.

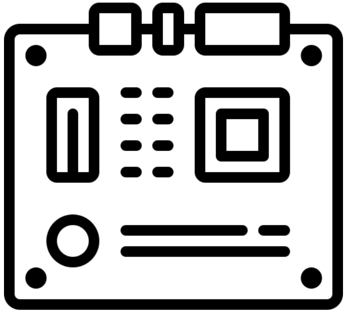
https://github.com/cncf/cnfs

What Are the Project Goals?

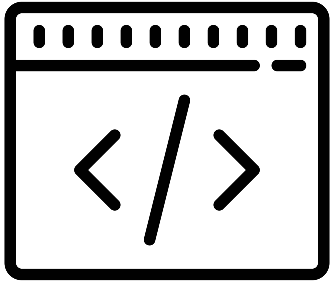
The ideal outcome of the CNF project is that a third party developer can recreate the entire test environment (and expected results) using the same reference code, a cloud provider API key and a couple of CLI commands.

Note: the CNF project is still in the prototype stage. Additional reference code and benchmarking tests will be added incrementally.

Creating a Neutral Test Environment



Hardware

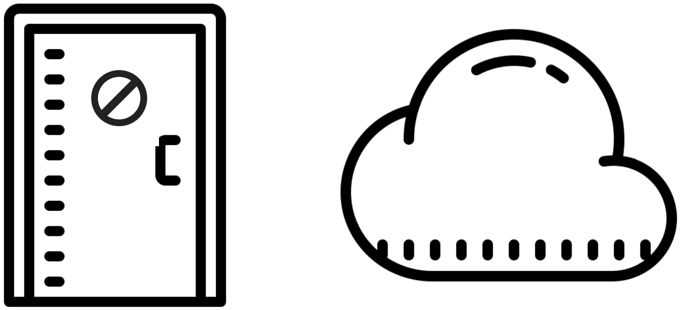


Software

Community

Neutral Hardware

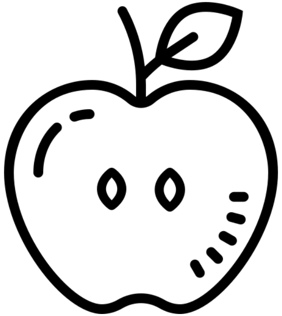
- Publicly accessible
  - Packet
- Multiple environments
  - FD.io CSIT
- Commodity hardware
  - NICs: Intel x710, Mellanox CNX-4
  - CPU: Intel Xeon Gold 5120



Private vs **Public**

Neutral Software

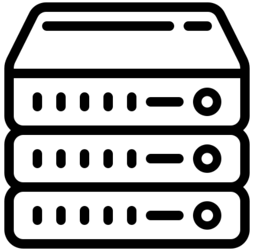
- 100% open source
- Installation best practices
  - Kubernetes Helm Charts
  - OpenStack Heat templates
- Vanilla OpenStack, Kubernetes, KVM
- Portable, open source data plane
  - VPP, DPDK
- Standard testing techniques
  - Apples-to-apples test case
  - Optimized test case



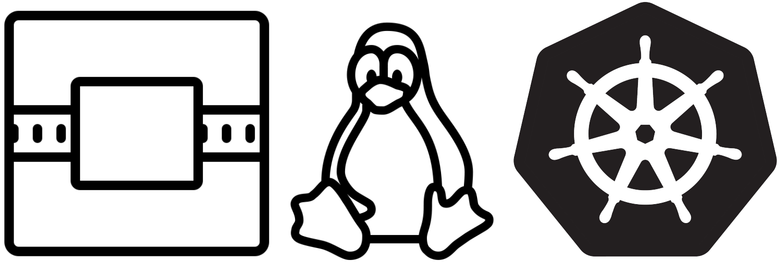
Neutral Community

- Cross-group:
  - CNCF CI Working Group, Network Service Mesh, fd.io
  - VPP, CSIT, Cross-Cloud CI
- Multi-vendor:
  - Intel, Cisco, Packet, Mellanox
- Governance:
  - In progress: Inspired by CNCF Charter

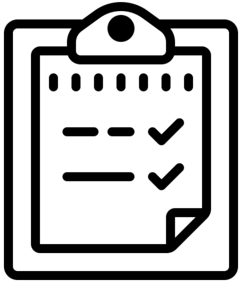
Reproducible Test Comparisons



Infrastructure



Test Environments



Test Cases

Test System Configuration

The machines running the network functions use commodity hardware and all configuration is documented.

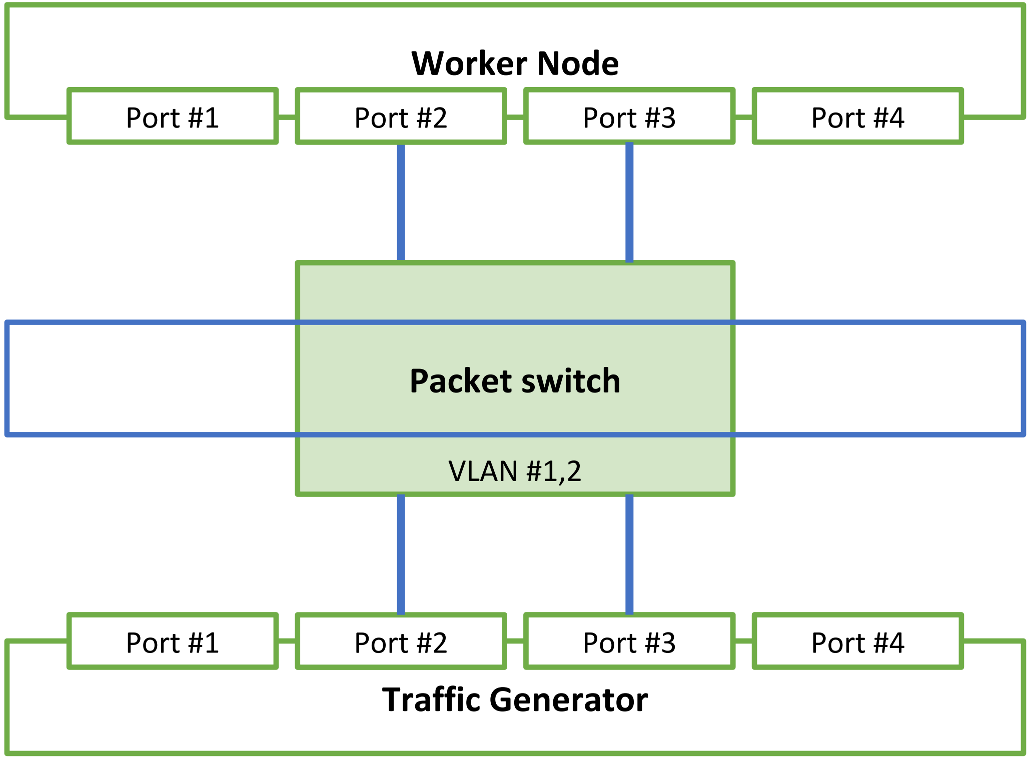
Specs at a glance:

- CPU: Dual socket Xeon Gold 5120 (2.2Ghz)
- Cores: 24 per CPU (48 total)
- Memory: 384 GB of DDR4 ECC
- Storage: 3.2 TB of NVMe Flash
- NIC: Quad port Intel x710

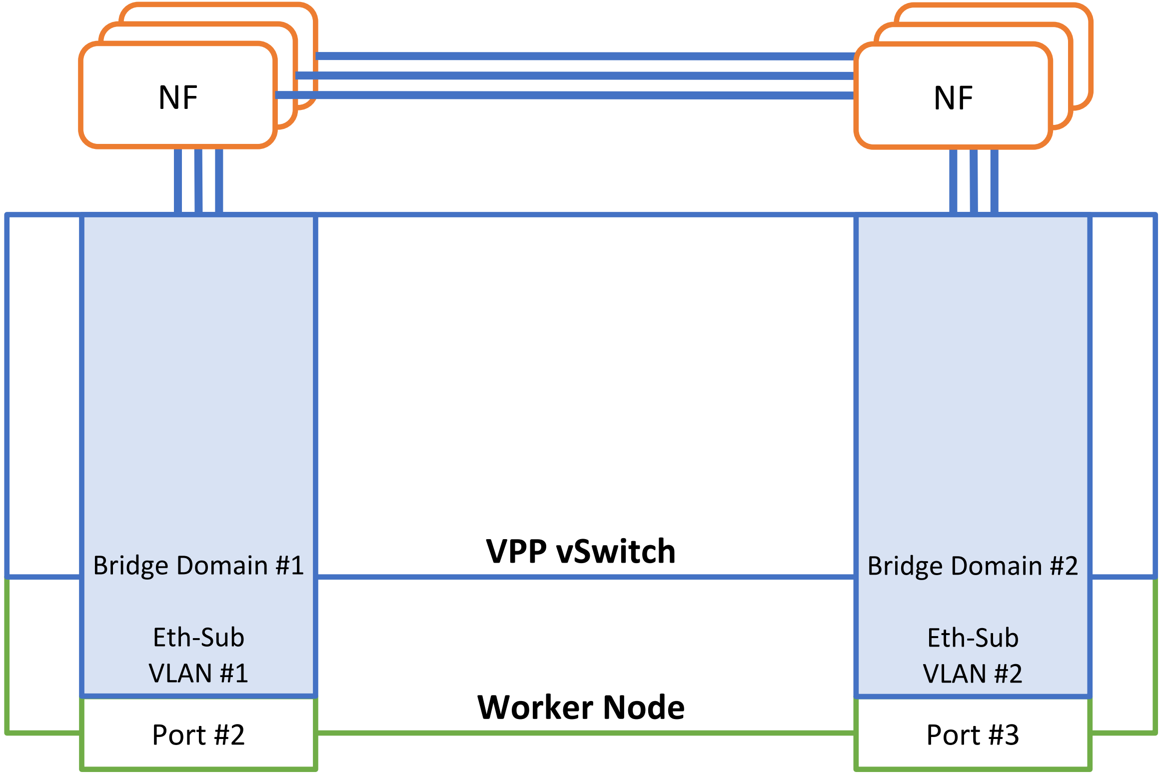
The system hardware configuration is based on the Packet m2.xlarge.x86.

Using either the default dual port Mellanox ConnectX-4 NIC or a quad port Intel x710 NIC. The NIC ports are connected to 10GbE ports on the top-of-rack switches.

Hardware Wiring Configuration



Layer 2 Configuration

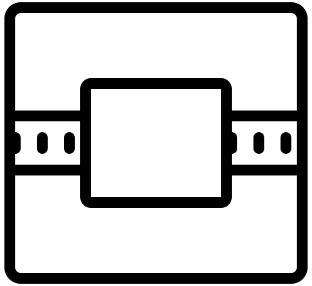


Reproducible Test Environments

KubeCon | CloudNativeCon North America 2018

OpenStack test environment:

- Deployment using the OpenStack Chef cookbook
- OpenStack services running on bare metal
- High-performance networking using the OpenStack VPP-neutron plugin
- Data plane VNFs using VPP



Reproducible Test Environments

KVM test environment:

- Deployment of KVM with Ansible
- KVM running on bare metal
- High-performance host networking using VPP as the vSwitch
- Data plane VNFs using VPP

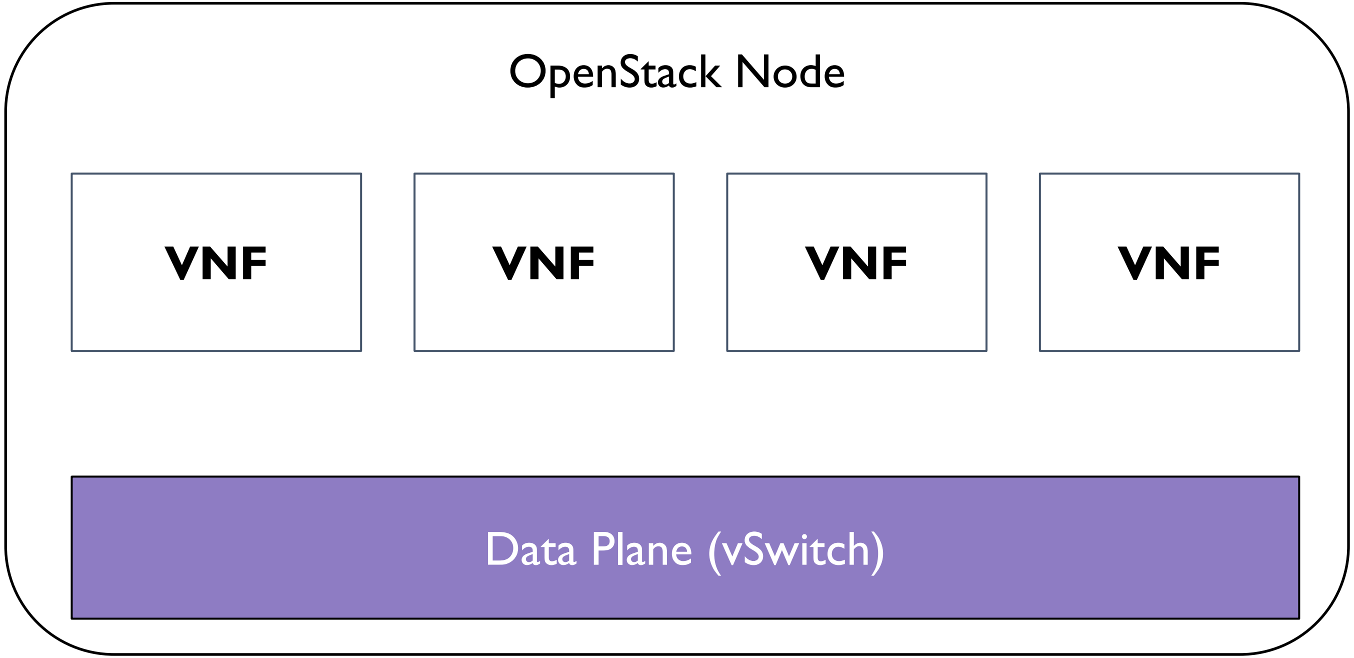
Reproducible Test Environments

Kubernetes test environment:

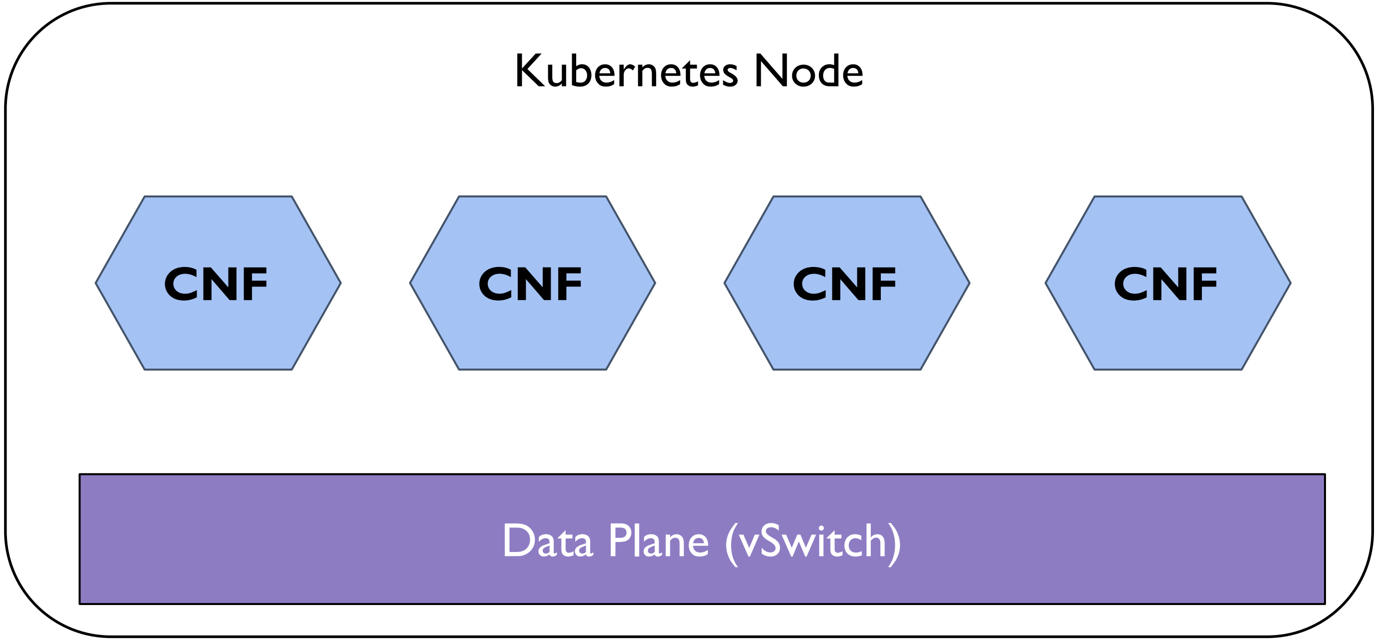
- Deployment of Kubernetes with cloud-init + Ansible
- Kubernetes services running on bare metal
- High-performance host networking using VPP as the vSwitch
- Data plane CNFs using VPP

Testing Network Functions: Service Topology & Density

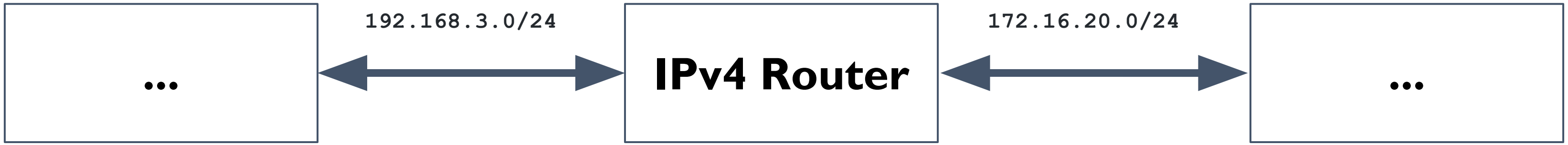
OpenStack Node Architecture



Kubernetes Node Architecture



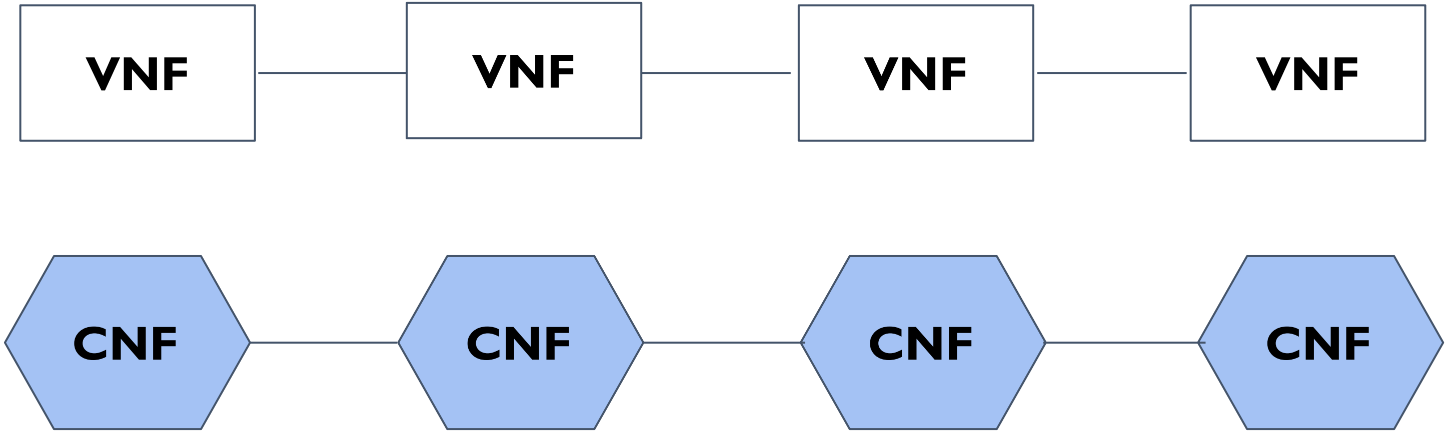
A Simple Network Function: IPv4 Router



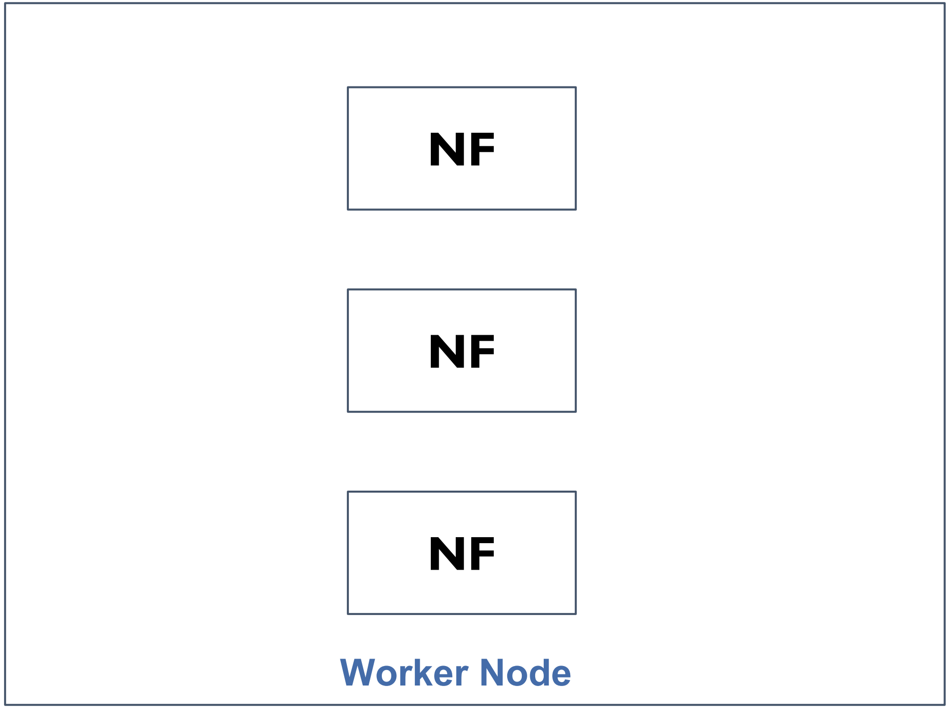
Packet Forwarding and Routing on IPv4 Networks

- This network function provides Layer-3 routing and has multiple interfaces.
- The system decides where to send a packet and forwards it over the correct interface to the next destination.

Logical Service Function Chains

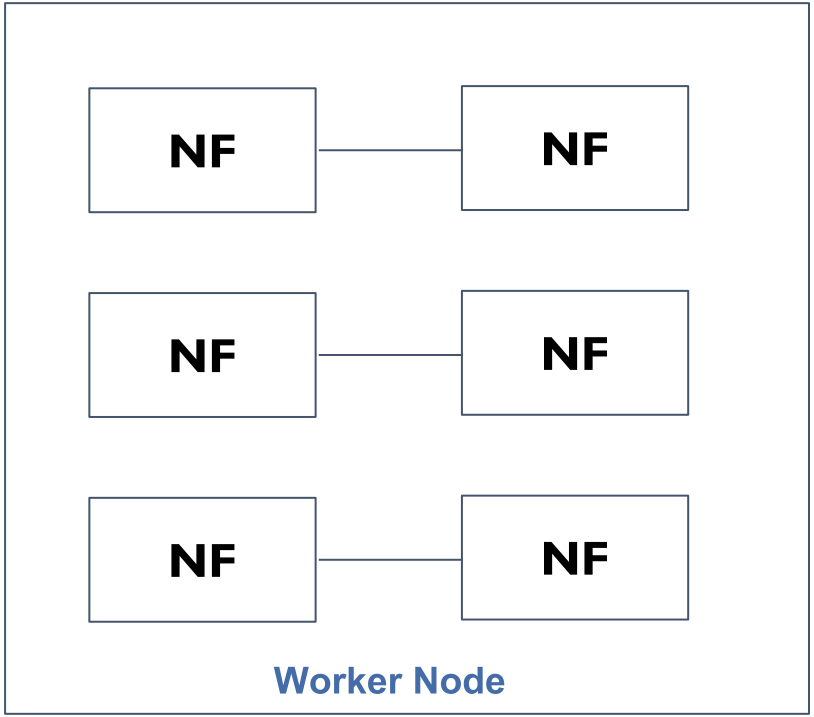


Service Density

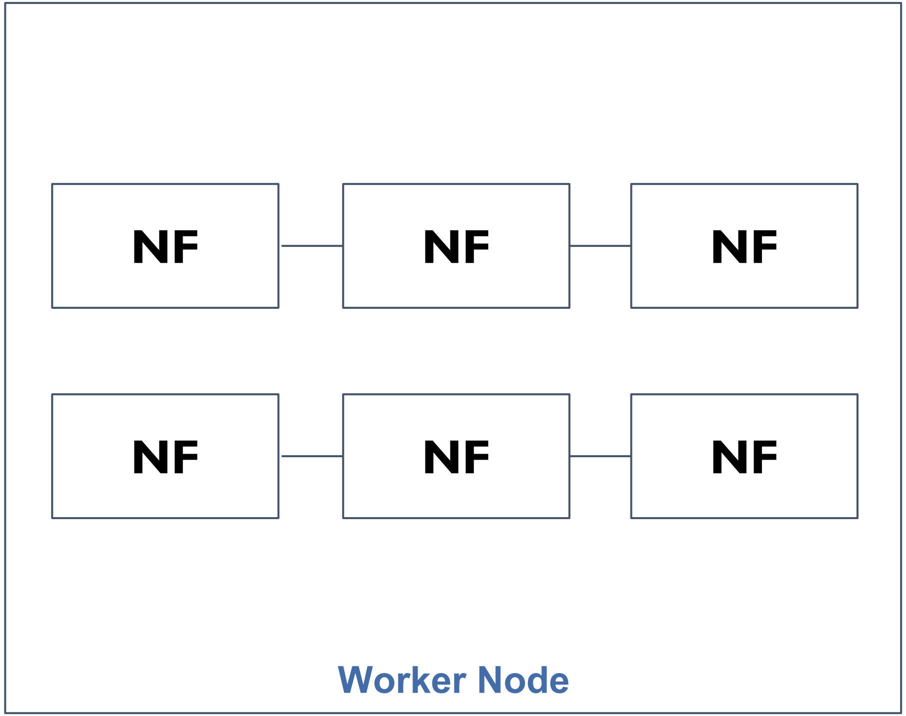


3 Chains of 1 Network Function each

Service Density



3 Chains of 2 Network Functions



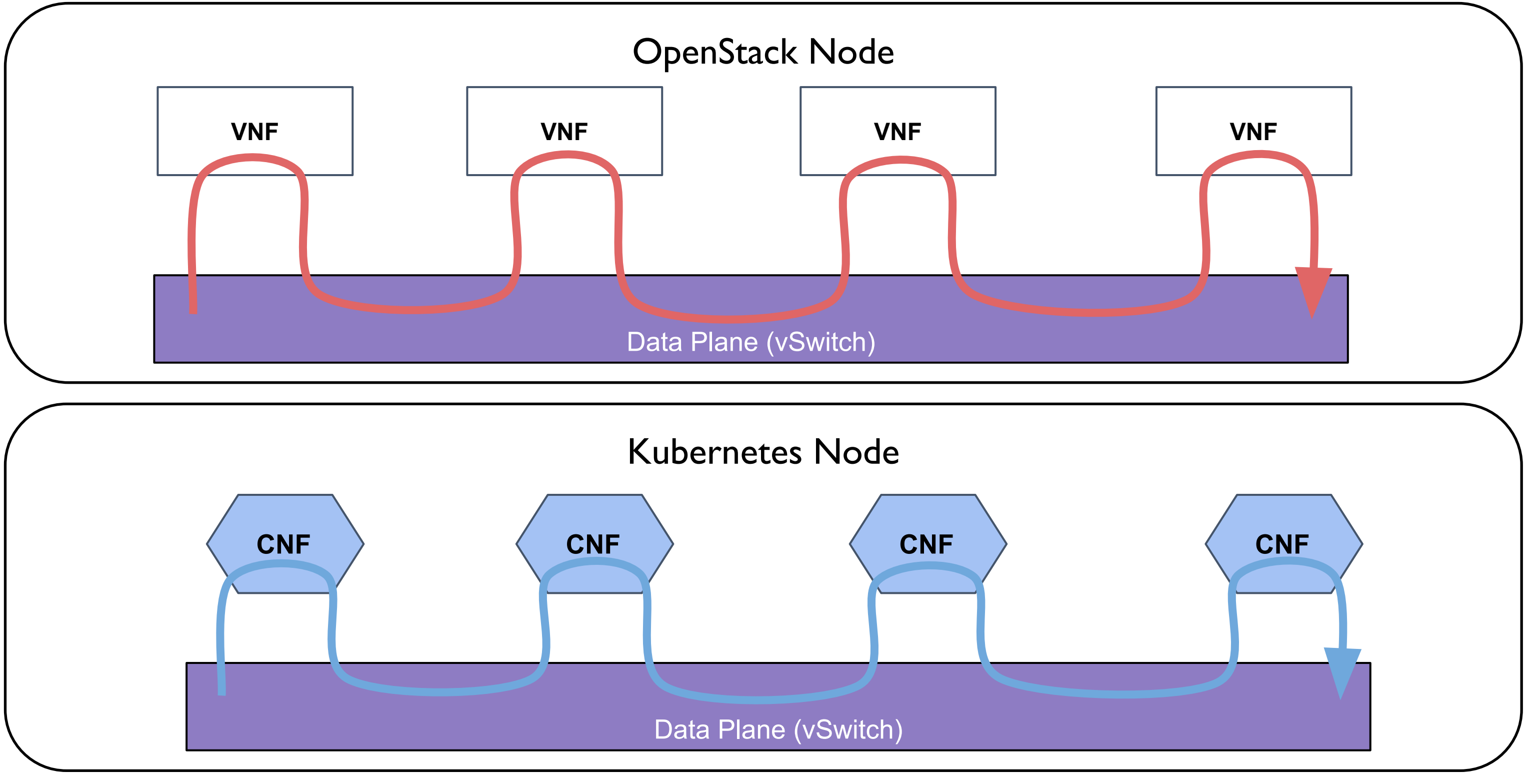
2 Chains of 3 Network Functions

Apples-to-Apples Test Case: Multichain Snake Test

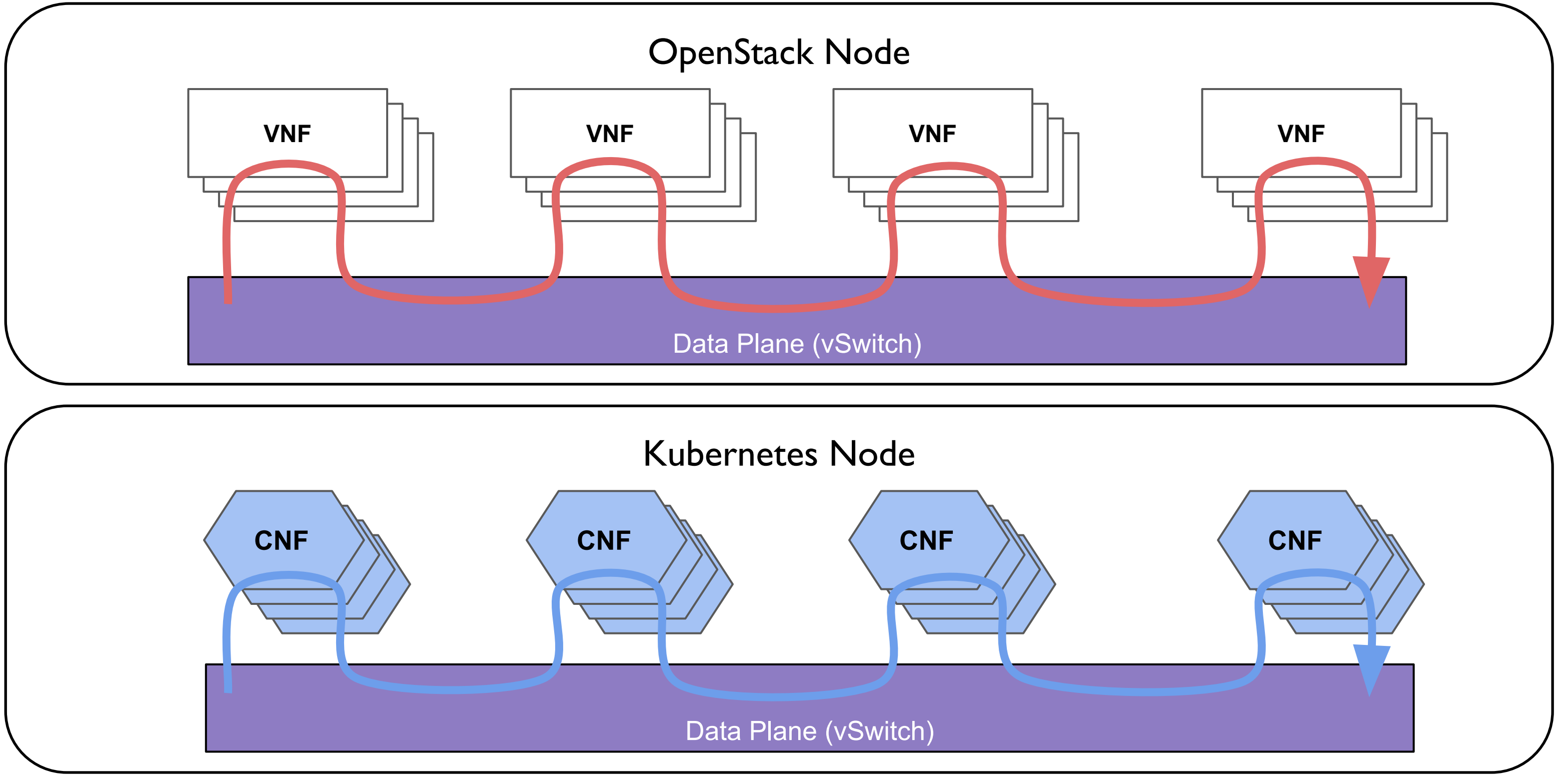
Multichain Snake Test

- The network functions run on a worker node running VPP as the vSwitch.
- Worker nodes run multiple sets of NFs **which loop out to the virtual switch (vSwitch) between each NF.**
- Benchmark tests the total packet throughput of multiple sets of NFs
- Test bed consists of two machines for the test traffic.
- A traffic generator sends packets to the target machine and collects the results.

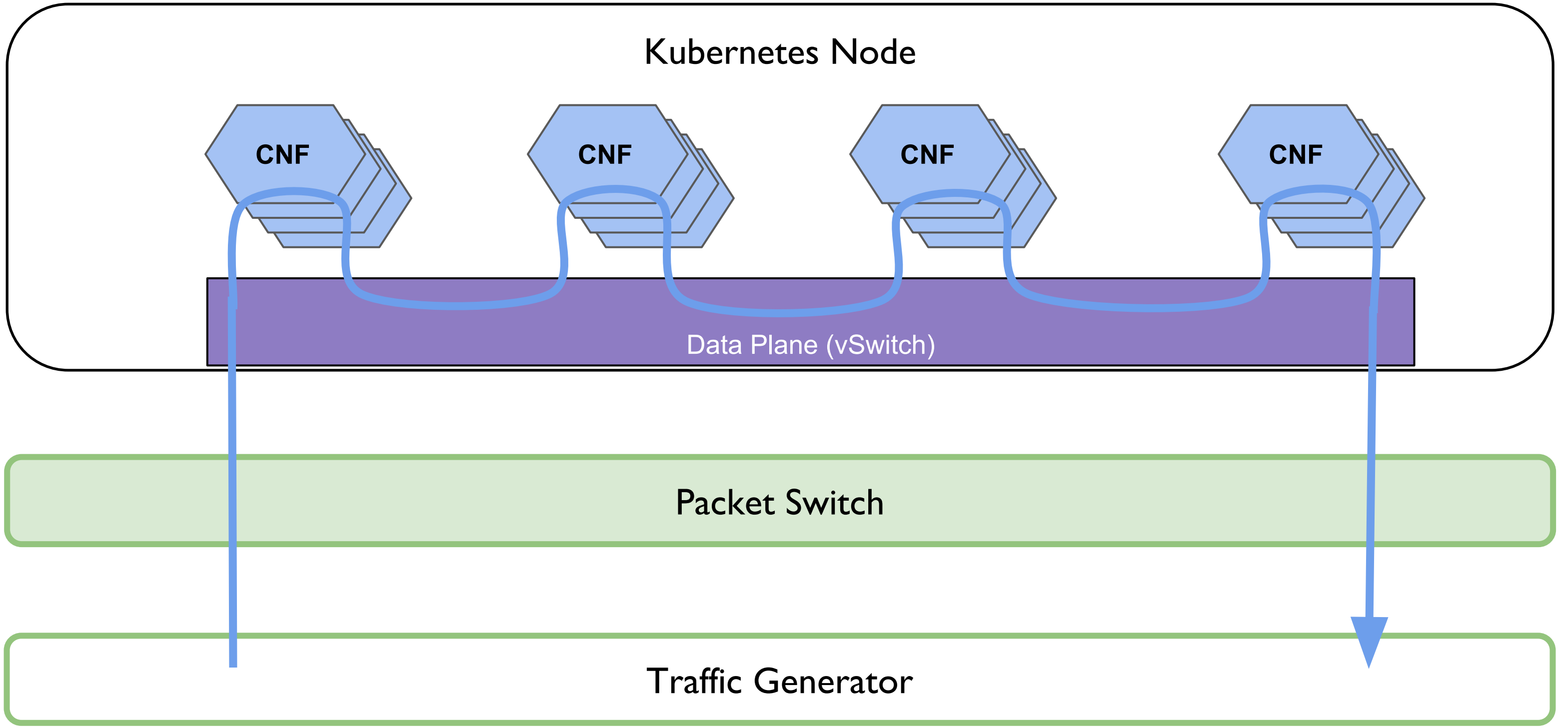
A Service Function Chain: Snake Test



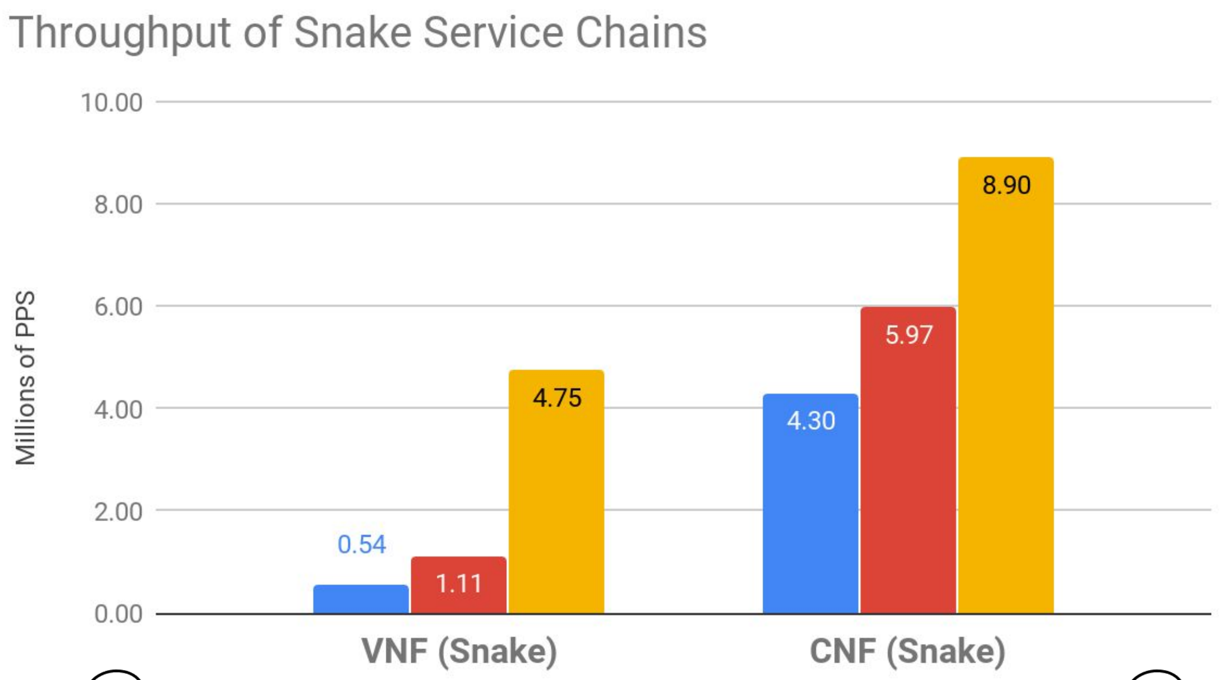
Multiple Service Function Chains: Snake Test



Running the Test



Multichain Snake Test Results



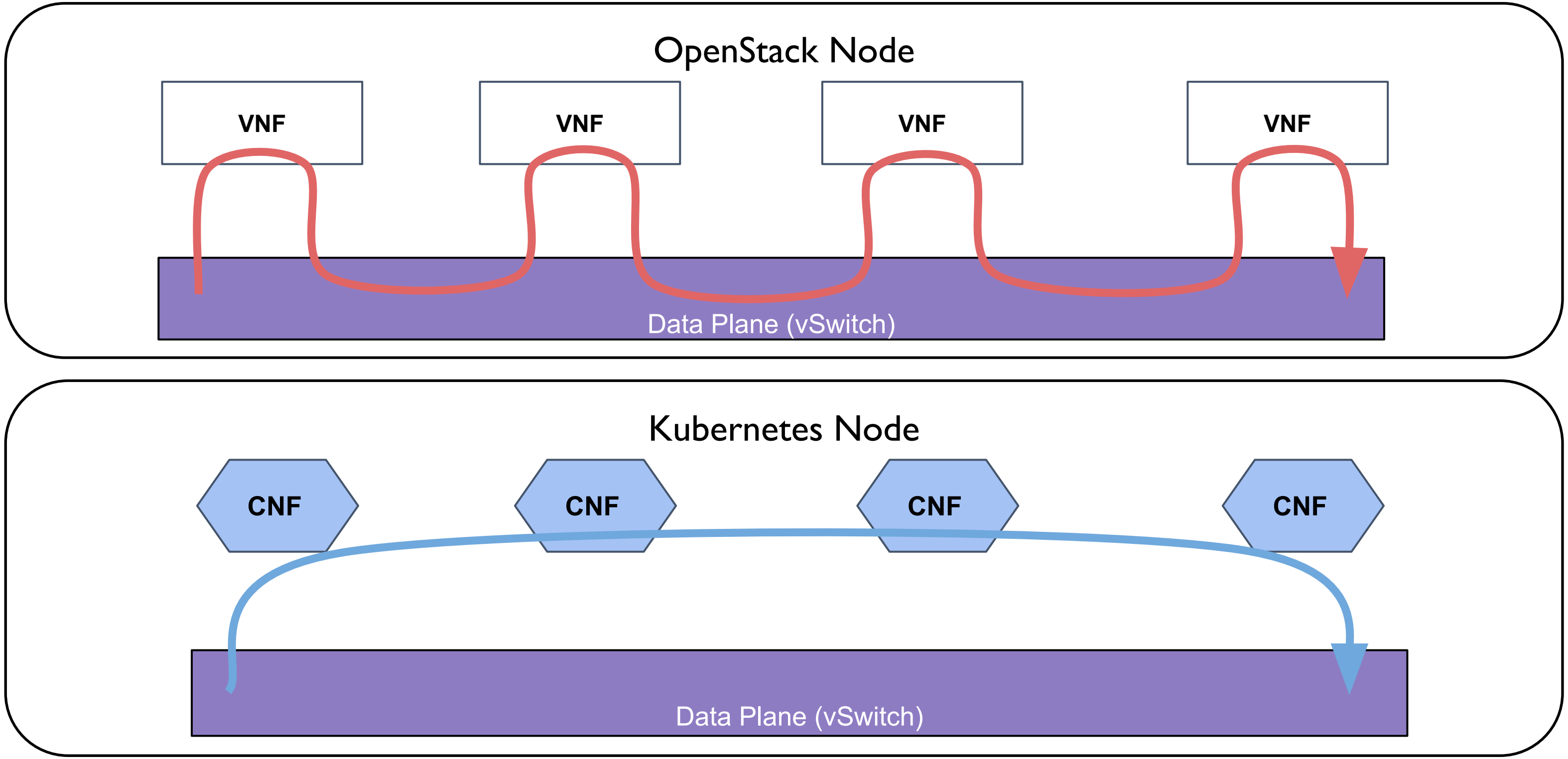
Best Test Case:
Optimal Connection Test

VS

Optimal Connection Test

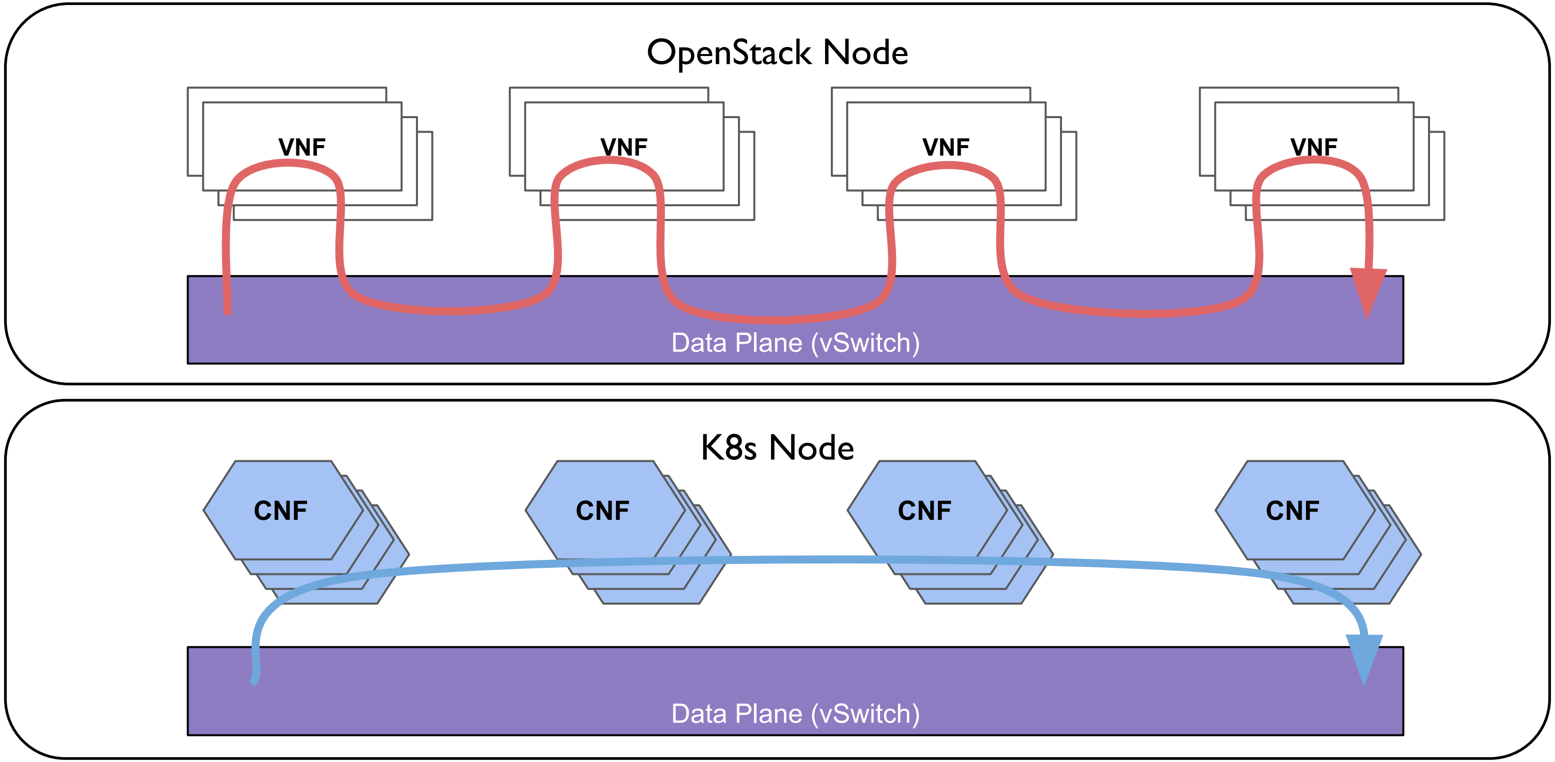
- The network functions run on a worker node running VPP as the vSwitch.
- OpenStack worker node runs multiple sets of VNFs **which loop out to the virtual switch (vSwitch) between each VNF.**
- Kubernetes worker node runs multiple sets of CNFs **which directly connect to each each other** before returning to the virtual switch (vSwitch)
- Benchmark tests the total packet throughput of multiple sets of NFs
- A traffic generator sends packets to the target machine and collects the results.

A Service Function Chain: Optimal Connection

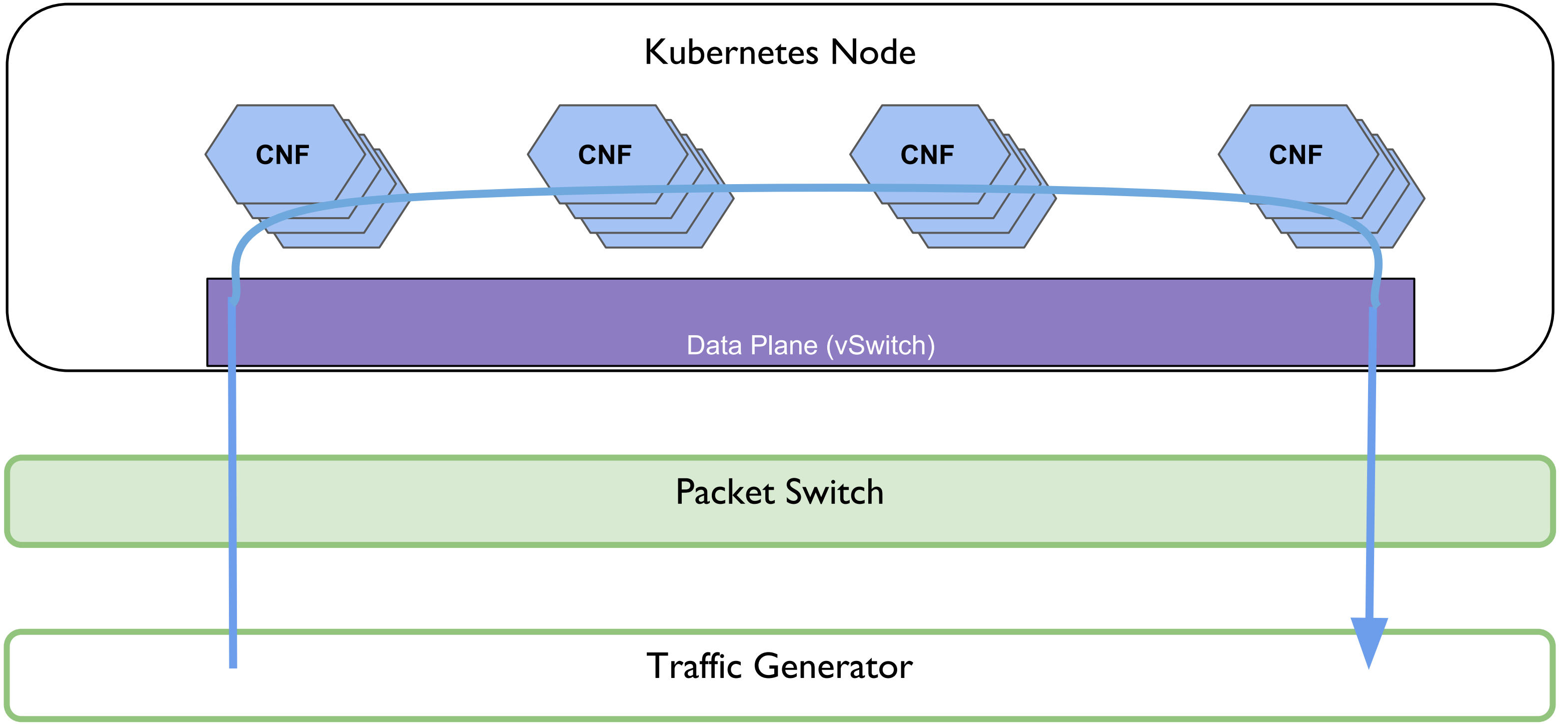


Multiple Service Function Chains: Optimal Connection

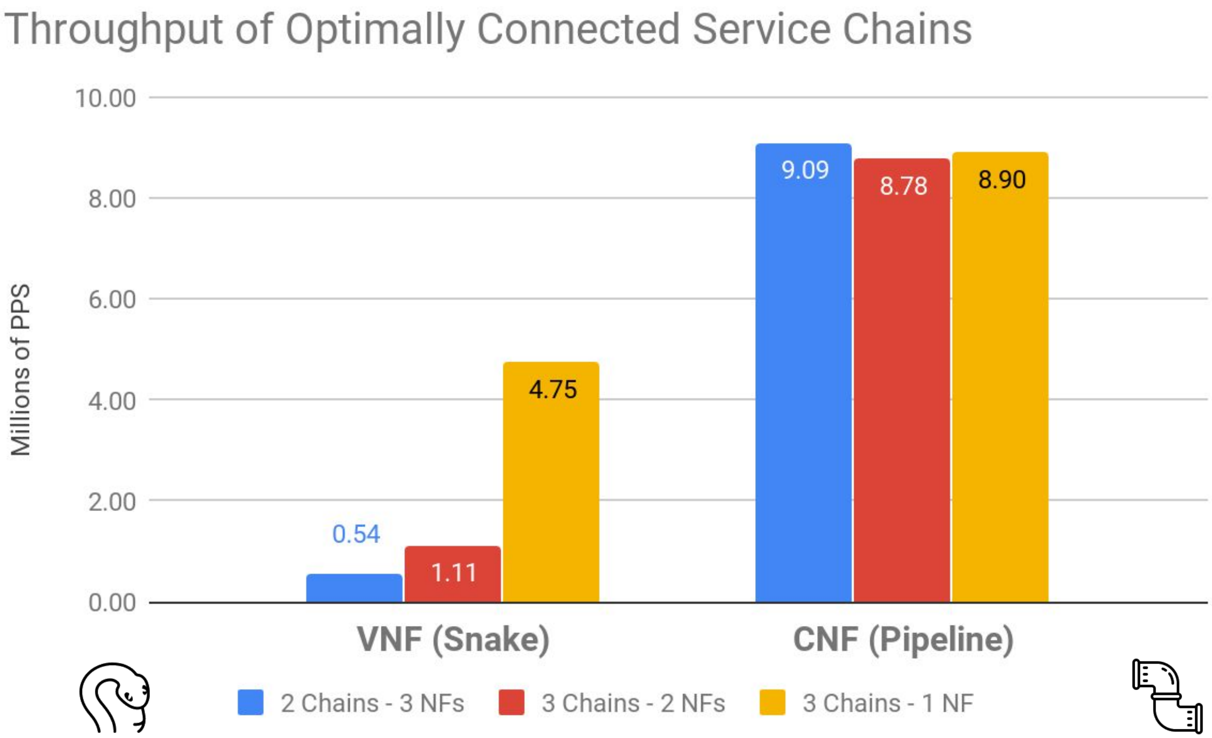
KubeCon | CloudNativeCon North America 2018



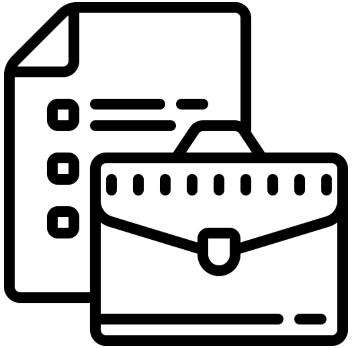
Running the Optimal Connection Test



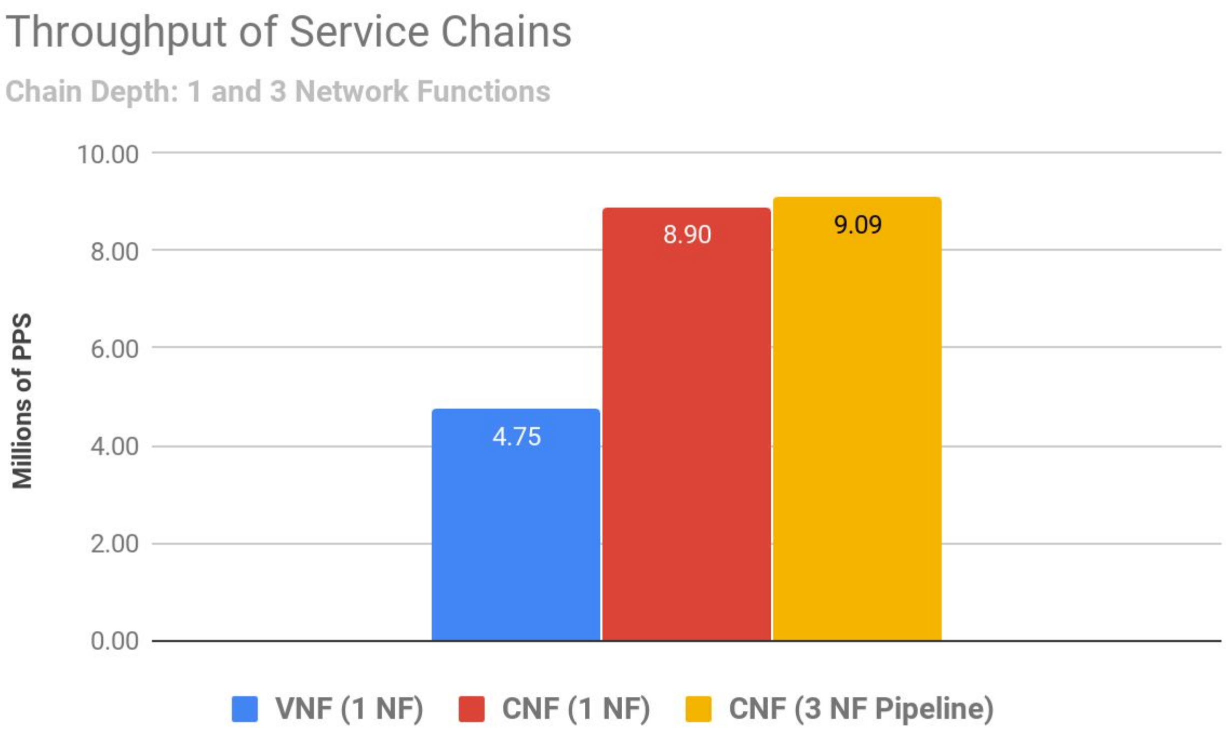
Best Test Case Results: vs



Summarizing the results



Summary of Test Results:



Verifying the Test Results

- All software is 100% Open Source
  - Available at https://github.com/cncf/cnfs
- Testing on Packet:
  - Packet Account
  - API Key
- Download code
- Follow the steps documented for the comparison

What’s Next for the CNF Project?

Next:

- Increasing collaboration with individuals and groups
- Add support to use Network Service Mesh (NSM)
- Additional use cases, including:
  - Test case with a few large optimized VNFs
  - Scenarios which use non-data plane CNFs (eg. CPE use case)
- Support other public test environments (ex. Amazon bare metal)

What's Next for CNCF CNFs?

Events and presentations:

- Cross-cloud CI Deep Dive on Wed, Dec 12 at 10:50am PT
  - https://sched.co/Greb
- KubeCon CNF BOF on Wed, Dec 12 at 2:35pm PT
  - https://sched.co/JCLS
- Mobile World Congress, Barcelona, February 25:
  - https://www.mwcbarcelona.com

Collaborate on CNFs

@cncf/cnfs

#cnf channel on CNCF Slack

cncfcnfs@vulk.coop

@vulkcoop

Q&A

Thank You!

Today's Presentation Prepared by:

taylor@vulk.coop
lucina@vulk.coop
watson@vulk.coop
denver@debian.nz


KubeCon
CloudNativeCon
North America 2018