Ligato
A Platform for Development of Cloud-Native VNFs
The way Applications are developed & deployed... has changed....
Microservices & Containers have changed many things...

- Applications are being developed and deployed very differently today.
  - Microservices allow you to split an application into many modular pieces, the network is how you stitch the pieces back together.
  - The interconnection of the pieces results in a more complex application network which consumes lots of resources.
  - The performance of the cloud native network is crucial to the behavior of the overall application.

It’s crucial we get "Container Networking" right!
Let’s not get “Openstacked”
Microservices & Containers have changed many things...

- Applications are being developed and deployed very differently today.
- Microservices allow you to split an application into many modular pieces, the network is how you stitch the pieces back together.
- The interconnection of the pieces results in a more complex application network which consumes lots of resources.
- The performance of the cloud native network is crucial to the behavior of the overall application.

It's crucial we get "Container Networking" right! Let's not get "Openstacked"
Solution #1
Move Cloud Native Networking out of the Kernel to Userspace

Container Networking moving from Kernel to Userspace
• Userspace enables rapid upgradability, highly available (doesn't bring down node), no system call overhead, no dependency on linux kernel networking community for features, higher performance and scale
• FD.io (dataplane), DPDK (network), SPDK (Storage) are examples
• Cloud Native apps are all connected by the network – lots of network end points to be managed, userspace offers lower overhead and higher performance
• Meltdown/Spectre bugs add a new tax for kernel networking
Contiv-VPP

- Kubernetes assumes seamless connectivity between pods, wherever it decides to place them. A networking plugin is needed to abstract the network.
- Contiv is a networking plugin for Kubernetes that:
  - Allocates IP addresses to Pods (IPAM)
  - Programs the underlying infrastructure it uses (Linux TCP/IP stack, OVS, VPP, …) to connect the Pods to other Pods in the cluster and/or to the external world.
  - Implements K8s network policies that define which pods can talk to each other.
  - Implements K8s services; a service exposes one or more (physical) service instances implemented as K8s pods to the other pods in the cluster and/or to external clients as a virtual instance (e.g. as a virtual “service” IP address).
- Contiv is a user-space based, high-performance, high-density networking plugin for Kubernetes - leveraging FD.io/VPP as the industry’s highest performance data plane.
Contiv-VPP Architecture

- Can deliver complete container networking solution entirely from userspace
- Replace all eth/kernel interfaces with memif/userspace interfaces.
- Apps can add VCL library for Higher Performance (bypass Kernel host stack and use VPP TCP stack)
- Legacy apps can still use the kernel host stack in the same architecture
Creating BD with BVI

Create BD

Create VXLAN Tunnel
(one per rmt node)

Create BVI
Solution #2
Cloud-Native VNFs

What Container-Networking Lacks for NFV Use-Cases:

- NFV-specific policy APIs (e.g. QoS, placement considering network resources)
- Networking:
  - HTTP or NAT-based load balancing isn’t suitable for NFV use-cases
  - No support for high-speed wiring of NFs:
    - To the outside world
    - To application containers
    - Between NFV containers
    - Creation of Service Function Chains (mixed physical and virtual – virtual a mix of VM and container)
- Management/Control:
  - Containerised NFs not really in the data plane (except for the vSwitch)
  - No support for cloud-native, high-performance NFs
- Forwarding:
  - Kernel used for forwarding – not sufficiently performance orientated (except for Contiv-VPP!)
Solution #2
Cloud-Native VNFs

- Kubernetes does not provide a way to stitch micro-services together today
- Ligato allows you to wire the data plane together into a service topology
- Network functions can now become part of the service topology
Service Function Chaining with Cloud-Native VNFs
Accelerating NFV Using Containers

- In VM case have to copy via the kernel
- With containers we use a shared memory interface (memif)
  - Key is to chain between NFs on the same server
  - Containers are “cheap” so can have dedicated chain per tenant service
Intra-Server Rendering
Point to Point – 2 options based on policy

Direct East/West Memif

Memif via vSwitch
VXLAN Rendering
Point to Point

Node 1
- CNF1
  - vSwitch
  - memif
  - policy

Node 2
- CNF2
  - vSwitch
  - memif
  - policy

VXLAN Tunnel
Dedicated VNI

Data Plane Network
Creating VXLAN xConnect

- Create memif
- Create VXLAN Tunnel
- Create xConnect
A VNF Cloud
A VNF Cloud: Data and Control Planes

Cloud-Native Control Plane

Cloud-Native Data Plane

Network

Cloud tools & services
VNFs & K8s Networking
ONS Demo

- K8s Master
- Contiv-VPP
- K8s Master
- Contiv-VPP

Nodes
Namespaces
Pods
Services
Policies

State distribution:
- K8s
- L3-L4 Service Mesh

- VPP
- x-connect
- Agent

- VPP
- x-connect
- Agent

VNF1 Overlay
VNF2 Overlay
K8s Vxlan Overlay
Putting it All Together...

IPv4/IPv6/SRv6 Network
Ligato on Github

- **cn-infra**
  A platform for developing cloud-native VNFs
  - Go: 13
  - GitHub stars: 17

- **vpp-agent**
  cn-Infra based VNF agent for VPP (FD.io)
  - Go: 18
  - GitHub stars: 23

- **sfc-controller**
  Service Function Chain (SFC) Controller for stitching virtual and physical networking
  - Go: 4
  - GitHub stars: 4

---

**Top languages**
- Go
- CSS
- Makefile

**Most used topics**
- cloud-native
- cn-infra
- vnf

**People**
18+
Ligato CN-Infra: a CNF Development Platform

www.github.com/ligato/cn-infra
Ligato VPP Agent: a CNF Management Agent

www.github.com/ligato/vpp-agent
Ligato Controller: a CNF Deployment Platform

[GitHub Link]

Diagram of Ligato Controller and its components, including APIs, Net Interfaces, ACL, L3, L2, Linux, VETH, VPP-Plugins, CN-Infra, and Core.
Backup
Network Micro-Service Use Case:
Service Function Chaining with Cloud-Native NFs