Cloud-native Container Networking
Orchestration at Cloud Scale, Data Plane with Terabit Performance
Cloud-native Container Networking
Using Best of Breed to Get Kick-Ass Cloud Networking Platform

- **Cloud-native Networking Platform**
  - Network as a Service
  - Network-as-a-Service

- **Production-Grade Container Orchestration**
  - Most portable on the planet
  - Superior extensibility and self-healing

- **Cloud-native NFV Orchestration**
  - Scalability, performance and agility
  - Marries K8s with NFV topologies

- **Fast Data Input/Output**
  - Most efficient on the planet
  - Top performance, flexibility and extensibility
Production-Grade Container Orchestration
For Native Cloud Network Services

- **PORTABILITY**
  - Public, private, hybrid, multi-cloud.

- **EXTENSIBILITY**
  - Modular, pluggable, hookable, composable.

- **SELF-HEALING**
  - Automatic: placement, restart, replication, scaling.

- **CLOUD SERVICES**
  - Foundation for cloud native network services

- **LINUX FOUNDATION**
  - Open source collaborative project in Linux Foundation
Cloud-native NFV Orchestration
For Native Cloud Network Services

Solid Foundations
- Kubernetes for automated deployment and lifecycle of containers, FD.io for efficient and performant data plane.

Scalability and Self-healing
- SCALABILITY and SELF-HEALING
  Follows Kubernetes scale and self-healing principles.

Flexible and Extensible
- SOFTWARE DEFINED NETWORKING
  Software programmable, extendable and flexible.

Cloud Native
- CLOUD SERVICES
  Foundation for cloud native network services

Open Source
- LINUX FOUNDATION
  Open source collaborative project in Linux Foundation
FD.io – Fast Data Input/Output
For Native Cloud Network Services

**EFFICIENCY**
The most efficient software data plane Packet Processing on the planet

**PERFORMANCE**
FD.io on x86 servers outperforms specialized packet processing HW

**RICH NFV FUNCTIONALITY**
Switching, routing, overlays, crypto, security, the list is very long, very network feature rich.

**CLOUD SERVICES**
Foundation for cloud native network services

**LINUX FOUNDATION**
Open source collaborative project in Linux Foundation

- Superior Performance
- Rich NFV Functionality
- Cloud Native
- Open Source
- Most Efficient on the Planet
Cloud-native Container Networking
For Native Cloud Network Services

- Kubernetes: Production-Grade Container Orchestration
- LIGATO: Cloud-native NFV Orchestration
- FD.io: Containerized Fast Data Input/Output

Enabling Production-Grade Native Cloud Network Services at Scale

Service Policy, Service Topology, Lifecycle

Production-Grade Container Orchestration

Network Function and Network Topology Orchestration

Containerized Network Data Plane

Cloud-native NFV Orchestration

Container Networking
Calico
Contiv Netmaster

Network Function and Network Topology Orchestration

Containerized Network Data Plane

Networking Plugin
Calico
Contiv Netmaster

Kubelet

Agent: Contiv Agent
FD.io VPP Agent
Agent: CNF
Agent: CNF Agent
Cloud-native Container Networking
Putting It All Together Now – The System Design

Functional Layered Diagram

Production-Grade Container Orchestration
- Kubernetes
- API Proxies

Network Function and Network Topology Orchestration
- SFC Controller

Containerized Network Data Plane
- Networking Plugin
- CNI
- Kubelet

Implementation Diagram as Demonstrated

Control and Management Plane
- Applications
- SFC Controller
- Tools (e.g., agentctl)

Inter-Process Communication
- Data Store
- Message Bus

Containerized Network Data Plane
- GoAGENT
- VPP eSwitch
- VPP CNF

Containers Lifecycle Orchestration

Putting It All Together Now – The System Design

- Cloud-native Container Networking
- Container Networking
- Networking Plugin
- CNI
- Kubelet

- Functional Layered Diagram
- Implementation Diagram as Demonstrated

- Production-Grade Container Orchestration
- Network Function and Network Topology Orchestration
- Containerized Network Data Plane

- Control and Management Plane
- Inter-Process Communication
- Containerized Network Data Plane
Demonstration

**Prepare System Environment**
1. Install Container orchestration infrastructure
2. Define Kubernetes PODs with Docker Containers in .yaml files

**Deploy Network Services**
3. Deploy defined Container PODs with Kubernetes
4. Deploy network topology with SFC_Controller via ETCD
5. Agents configure their local VPPs containerized in K8s PODs

**Verify Service is Up**
6. Verify Containerized network topology is configured correctly

**Benchmark for Service Acceptance**
7. Benchmark Containerized network topology
Demonstration – Containerized NFV Scenarios

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>memif</td>
<td>scenario1</td>
<td>scenario2</td>
<td>scenario3</td>
</tr>
<tr>
<td>af_packet</td>
<td>scenario4</td>
<td>scenario5</td>
<td>scenario6</td>
</tr>
</tbody>
</table>

I’m demonstrated
I’m not demonstrated