FD.io VPP & Ligato Use Cases
Contiv-VPP CNI plugin for Kubernetes
IPSEC VPN gateway
Packet Processing Software Platform

- High performance
- Linux user space
- Runs on compute CPUs: And “knows” how to run them well!

Shipping at volume in server & embedded products
Packet processing is decomposed into a directed graph of nodes …

- Each graph node implements a “micro-NF”, a “micro-NetworkFunction” processing packets.

… packets move through graph nodes in vectors …

… graph nodes are optimised to fit inside the instruction cache …

Makes use of modern Intel® Xeon® Processor micro-architectures.
Instruction cache & data cache always hot ➔ Minimised memory latency and usage.
Cloud-Native Network Functions with VPP & Ligato

- **cncf.io**: "Cloud native computing uses an open source software stack to deploy **applications as microservices**, packaging each part into its **own container**, and dynamically **orchestrating** those containers to optimize resource utilization."

- **CNFs**: Splitting of network functions into collection (chain) of loosely coupled services, deployed as containers, interconnected with fast data plane links.

- **VPP**: data plane part for building CNFs (memifs for interconnecting)

- **Ligato**: control plane part for building CNFs (VPP agent) and CNF chaining (SFC Controller)
Service Function Chaining with CNFs & VPP

Logical Representation

- Ingress Network
- Ingress Classifier

- NF₁
- NF₂
- NF₃
- Egress Classifier

- Egress Network

Placement (K8s) ➔ Rendering ➔ Topology

Physical Representation

Host

- CNF₁ VPP
- CNF₂ VPP
- ⋮
- CNF VPP

Ingress Classifier

Server

- CNF VPP
- CNF VPP
- ⋮
- CNF VPP

Ingress Classifier

Overlay Tunnel

github.com/ligato/sfc-controller

networkservicemesh.io
Ligato VPP Agent
(Development Platform for VPP-based / Non-VPP-based CNFs)

VPP-based CNFs require a cloud-native management agent for VPP:

- NETCONF/RESTCONF + YANG does not fit to cloud-native very well

- **Protobuf API** is used instead; via any transport:
  - gRPC / REST (for RPC-based configuration / state data retrieval)
  - Key-value datastores: ETCD, Redis, Bolt DB, ... (asynchronous configuration)
  - Message brokers: Kafka, (RabbitMQ, NATS, ...)

- Needs to be pluggable into existing cloud-native infra (k8s, Docker)
  - Implemented in **Go** (as well as Docker & k8s)
  - The agent can be used as a library from other Go applications (Contiv-VPP)

- Needs to be compact and low footprint
  - Packaged into **Docker container images** together with VPP
  - Can be running in multiple containers on the same host
  - Need to have fast startup times (containers can die and restart frequently)
  - VPP Agent is written in Go – it is a single executable (binary)
Ligato VPP Agent
(Development Platform for VPP-based / Non-VPP-based CNFs)

VPP:
• Binary API – shared memory / unix domain socket

GoVPP:
• Golang wrappers over binary APIs (generator)
• Go struct to binary API marshalling & unmarshalling, communication with VPP
• Still binary API
  • low-level (numeric references to interfaces, byte ordering issues, bit flags, etc.)
  • API message ordering / dependency issues
  • API versioning issues (API needs to match with VPP version)

VPP Agent:
• Provides protobuf-modelled NB API
  • more high-level, uses labels (names) for referencing
  • does not change frequently, allows backward-compatible API changes
  • various transports for the same API: RPC (gRPC, REST), key-value store (ETCD, Redis), …
• KVScheduler
  • transaction-based configuration processing using graph processing engine (addresses the ordering issue)
  • error state handling: handles restarts, retries on error, auto-healing (resync)
• Modular & extendable – VPP plugins, Linux plugin
• Packaged as a Docker container together with compatible VPP (no versioning issues)
Ligato VPP Agent
(Development Platform for VPP-based / Non-VPP-based CNFs)
Contiv-VPP
(Container Network Interface Plugin for Kubernetes)

Ligato VPP Agent integrated into Kubernetes ecosystem:

• Provides k8s **CNI** (Container Network Interface) functionality
  • interconnects the PODs in the cluster (TAP interfaces)
  • uses VXLAN tunnels for node interconnection or no overlay mode
  • provides kube-proxy functionality (load-balancing + NAT) on VPP
  • Implements k8s policies as ACLs on VPP

• At the same time, still exposes the same APIs as the Ligato VPP Agent
  • uses Ligato VPP Agent as a library to program VPP
  • can be easily extended with extra configuration – e.g. to create additional memif interfaces to CNF PODs

[github.com/contiv/vpp]
Contiv-VPP
(Container Network Interface Plugin for Kubernetes)
Contiv-VPP
(Container Network Interface Plugin for Kubernetes)

github.com/contiv/vpp
Contiv-VPP + CNFs

Cloud tools (DB, Mbus,...)

Contiv Control Plane

Etcld

Kubernetes

SFC Controller

Physical Device

Physical Device

Physical Device

Physical Device

Physical Device

Physical Device

Contiv-VPP

CNF

Agent

VPP

Cloud VRF

VRF X

VRF Y

Contiv-VPP

CNF

Agent

VPP

Cloud VRF

VRF X

VRF Y

Contiv-VPP

CNF

Agent

VPP

Cloud VRF

VRF X

VRF Y

Contiv-VPP

CNF

Agent

VPP

Cloud VRF

VRF X

VRF Y

Contiv-VPP

CNF

Agent

VPP

Cloud VRF

VRF X

VRF Y

Contiv-VPP

CNF

Agent

VPP

Cloud VRF

VRF X

VRF Y

Contiv-VPP

CNF

Agent

VPP

Cloud VRF

VRF X

VRF Y
k8s Services & Load Balancing with Contiv-VPP
k8s Services & Load Balancing with Contiv-VPP

```yaml
apiVersion: apps/v1beta2
kind: ReplicaSet
metadata:
  name: nginx
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx
template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx
        ports:
        - name: http
          containerPort: 80
```

```yaml
apiVersion: v1
kind: Service
metadata:
  name: nginx
spec:
  type: LoadBalancer
  ports:
  - name: http
    port: 80
    protocol: TCP
    targetPort: 80
  selector:
    app: nginx
```

```yaml
apiVersion: v1
kind: ConfigMap
metadata:
  namespace: metallb-system
  name: config
data:
  config: |
    address-pools:
      - name: my-ip-space
        protocol: layer2
        addresses:
        - 192.168.16.150-192.168.16.250
```

MetalLB (https://metallb.universe.tf) used for external load balancing

```bash
vagrant@k8s-master:$ kubectl get svc nginx
NAME     TYPE           CLUSTER-IP       EXTERNAL-IP   PORT(S)        AGE
nginx    LoadBalancer  10.107.167.81   192.168.16.150 80:32676/TCP   1m

vagrant@k8s-master:$ kubectl get pods -o wide
NAME              READY STATUS    RESTARTS AGE  IP              NODE          NOMINATED NODE
nginx-bnh78       1/1     Running   0        1m  10.1.3.2       k8s-worker2  <none>
nginx-rghrf       1/1     Running   0        1m  10.1.4.2       k8s-worker3  <none>
nginx-w57kx       1/1     Running   0        1m  10.1.2.2       k8s-worker1  <none>
```

github.com/contiv/vpp
k8s Services & Load Balancing with Contiv-VPP

vagrant@k8s-worker1:$ sudo vppctl

vpp# sh inter addr
GigabitEthernet0/8/0 (up):
  L3 192.168.16.2/24
local0 (dn):
  loop0 (up):
    L3 10.1.2.1/24 ip4 table-id 1 fib-idx 1
loop1 (up):
  L2 bridge bd-id 1 idx 1 shg 1 bvi
  L3 192.168.30.2/24 ip4 table-id 1 fib-idx 1
tap0 (up):
  L3 172.30.2.1/24
tap1 (up):
  unnumbered, use loop0
    L3 10.1.2.1/24 ip4 table-id 1 fib-idx 1
vxlan_tunnel0 (up):
  L2 bridge bd-id 1 idx 1 shg 1
vxlan_tunnel1 (up):
  L2 bridge bd-id 1 idx 1 shg 1
vxlan_tunnel2 (up):
  L2 bridge bd-id 1 idx 1 shg 1

vagrant@k8s-worker1:$ sudo vppctl

vpp# sh nat44 static mappings
tcp external 10.107.167.81:80 self-twice-nat out2in-only
  local 10.1.2.2:80 vrf 1 probability 1
  local 10.1.3.2:80 vrf 1 probability 1
  local 10.1.4.2:80 vrf 1 probability 1
tcp external 192.168.16.150:80 twice-nat out2in-only
  local 10.1.2.2:80 vrf 1 probability 1
  local 10.1.3.2:80 vrf 1 probability 1
  local 10.1.4.2:80 vrf 1 probability 1

vagrant@k8s-worker1:$ sudo vppctl

vpp# sh ip arp
Proxy arps enabled for:
  Fib_index 0 192.168.16.150 - 192.168.16.150
k8s Services & Load Balancing with Contiv-VPP

L2 load-balancing (failover) 
externalTrafficPolicy: Cluster

BGP load-balancing 
externalTrafficPolicy: Local

Load Balancer
(router)

100%
0%

33,3%

VPP

POD 33,3%

POD 33,3%

Node 1

VPP

POD 33,3%

Node 2

VPP

POD 25%

POD 25%

Node 1

VPP

50%

Node 2

github.com/contiv/vpp
IPSEC in fd.io VPP

Forwarding Plane:

- ESP tunnel/transport modes
- IPv4/IPv6
- SHA up to 512-256 and AES-CBC 128/192/256

Control Plane:

- IKEv2 Initiator and Responder

Tested/Documented Interop with other stacks (e.g. StrongSwan)
Migration from VM/StrongSwan to Contiv/Ligato VNFs

- **VM**
  - Server running a VM with StrongSwan and Linux kernel
  - StrongSwan Daemon
    - User Space
    - Linux kernel
    - Bare Metal or HyperVisor

- **VM**
  - Server running StrongSwan with a plugin to VPP for higher IPSEC pps throughput
  - StrongSwan Daemon
    - User Space
    - Linux kernel
    - Bare Metal or HyperVisor

- **Ligato**
  - Ligato CNF running StrongSwan, a Ligato agent, and a VPP for IPSEC data plane - with a memIf to the Ligato vSwitch
  - Kubernetes is optional
  - Agent strongSwan Daemon
    - User Space
    - Linux kernel
    - VPP
    - Vswitch
    - Bare Metal or HyperVisor

- **Ligato**
  - Full K8s/Contiv-vpp integration with ligato CNF, dynamic re-wiring of CNFs using sfc-controller based on k8s placement policies
  - Agent strongSwan Daemon
    - User Space
    - Linux kernel
    - VPP
    - Vswitch
    - Bare Metal or HyperVisor
Multiple StrongSwan CNFs may be deployed on a host. K8s handles CNF placement/lifecycle.

N/B VXLAN tunnel between the host’s vSwitch and the north bound router towards the secure apps.

S/B VXLAN tunnel between the host’s vSwitch and the south bound router towards the remote IPSEC clients.

sswan-controller configures memifs and VXLAN tunnels, sends BGP updates via sfc-controller plugins, and sends security config to the StrongSwan CNFs.
Summary

- Cloud Native Container Networking
  - Contiv-VPP, VPP – Container networking and policy fabric with scale and performance
  - Ligato – Cloud-native container control and wiring
  - Kubernetes – Container orchestration and scheduling
Want to dig deeper into and try VPP:

- https://wiki.fd.io/view/VPP

- Ligato:
  - https://github.com/ligato
  - https://github.com/ligato/vpp-agent

- More on Contiv VPP:
  - https://contivpp.io
  - Getting started: https://github.com/contiv/vpp
Backup
Network Micro-Service Use Case:
Service Function Chaining with Cloud-Native NFs