Join us for KubeCon + CloudNativeCon Virtual

Event dates: August 17-20, 2020

Schedule: Now available!

Cost: $75

Register now!

Securing and Accelerating Kubernetes CNI

Integrating Project Antrea and NVIDIA Mellanox ConnectX SmartNICS

Antonin Bas
Maintainer of Project Antrea and Staff Engineer at VMware

Moshe Levi
Sr. Staff Engineer at NVIDIA

July 14, 2020

Antonin Bas

Maintainer of Project Antrea and
Staff Engineer at VMware
abas@vmware.com

Cody McCain

Sr. Product Manager Container
Networking at VMware
mmccain@vmware.com

Moshe Levi

Sr. Staff Engineer at NVIDIA
moshele@nvidia.com

Itay Ozery

Director, Product Management
for Networking at NVIDIA
itayo@nvidia.com

Agenda

Securing and Accelerating the Kubernetes CNI Data Plane

Kubernetes Cluster Networking

Project Antrea Deep Dive

Hardware Acceleration

Demo

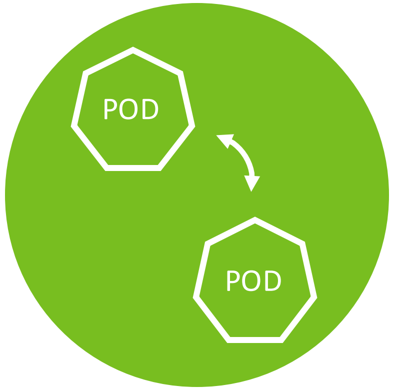
Roadmap

Get Involved

Kubernetes
Cluster Networking

Kubernetes Cluster Networking

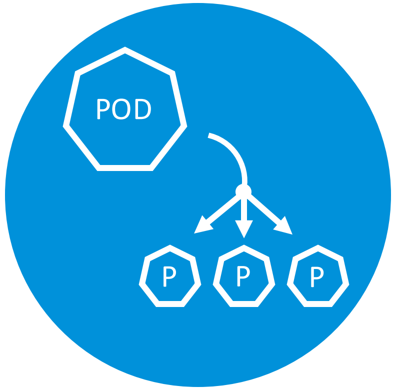
Three connectivity scenarios must be enabled.



Pod

-to-

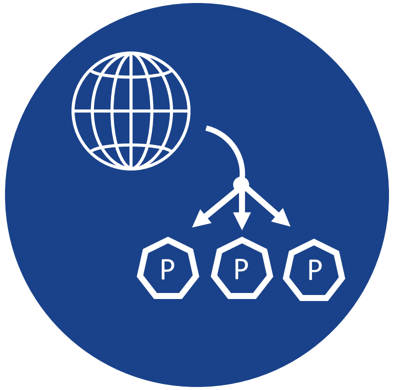
Pod



Pod

-to-

Service

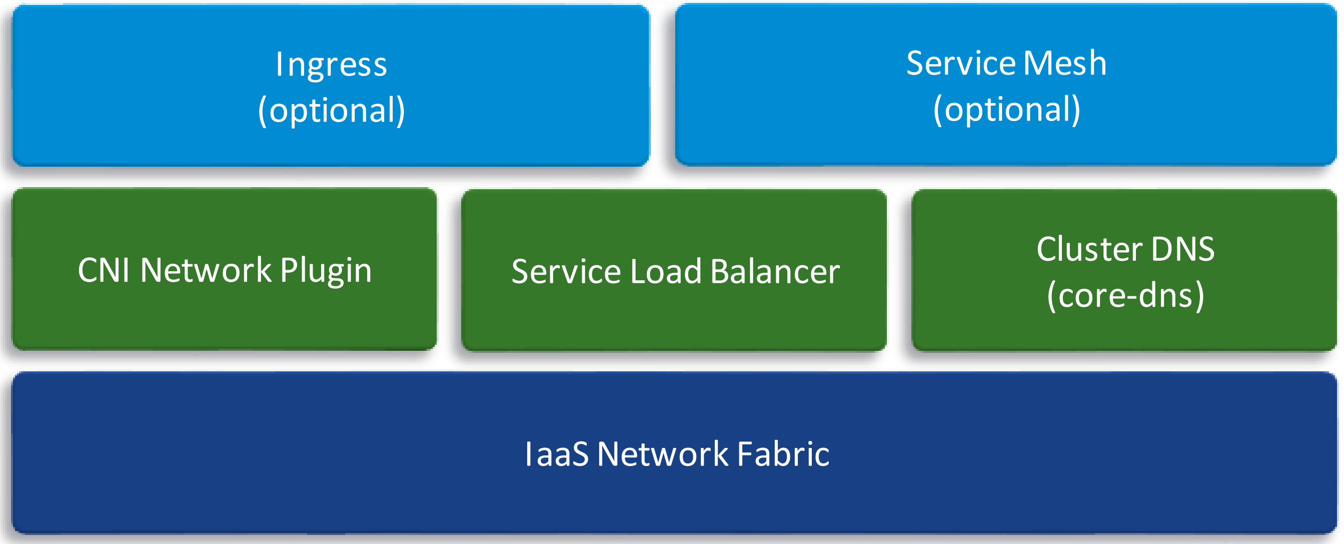


External

-to-

Service

Kubernetes Networking in Layers



Ingress (optional)

Service Mesh (optional)

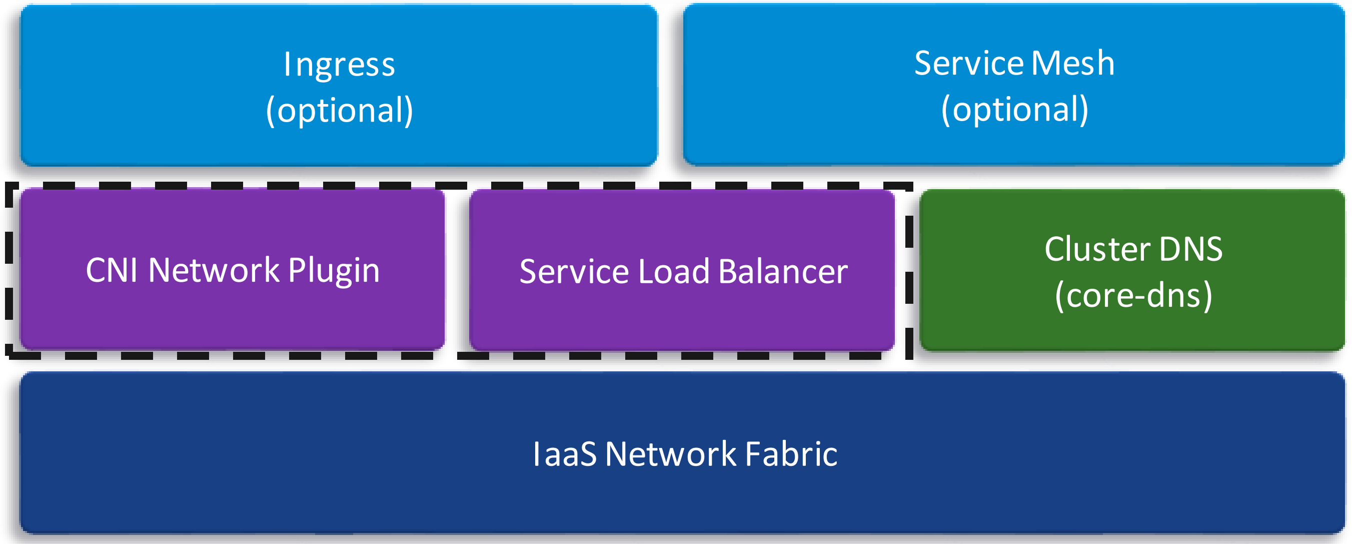
CNI Network Plugin

Service Load Balancer

Cluster DNS (core-dns)

IaaS Network Fabric

Kubernetes Networking in Layers



What is a

Kubernetes CNI Network Plugin

responsible for?

Pod Connectivity

Plumbing eth0 (network interface) into Pod network (encapsulated or non-encapsulated)
Pod egress to world – SNAT

IP Address Management (IPAM)

Service Load Balancing

Make traffic available to upstream kube-proxy, or
Implement native service load balancing – VIP DNAT

NetworkPolicy Enforcement (optional)

Enforcing Kubernetes Network Policy
Source Spoof Prevention
Connection Tracking (Stateful Firewall)

hostPort Support

Traffic Shaping Support

(experimental)

Project Antrea
Deep Dive

ANTREA

https://antrea.io

https://github.com/vmware-tanzu/antrea

@ProjectAntrea

Kubernetes Slack – #antrea

Project Antrea is an open source CNI network plugin providing pod connectivity and network policy enforcement with Open vSwitch in Kubernetes.

ANTREA = CNI + OvS Open vSwitch + kubernetes

ANTREA

https://antrea.io

https://github.com/vmware-tanzu/antrea

@ProjectAntrea

Kubernetes Slack – #antrea

Antrea is a community driven project focused on

- simplifying usability and diagnostics,
- adapting any network topology, and
- improving scaling and performance

for container networking in Kubernetes.

```
kubectl apply -f \
https://github.com/vmware-tanzu/antrea/releases/download/v0.8.0/antrea.yml
```

661

GitHub Stars

111

GitHub Forks

29

Contributors

Where can I run Antrea?

Our goal is to run anywhere Kubernetes runs.

Private Cloud

Public Cloud

Edge

Linux

Windows

What is Open vSwitch (OVS)?

And why use it for K8s networking?

A high-performance programmable virtual switch

- Connects to VMs (tap) and containers (veth)

Linux foundation project, very active

Portable: Works out of the box on all Linux distributions and supports Windows

Programmability: Supports many protocols, build your own forwarding pipeline

High-performance

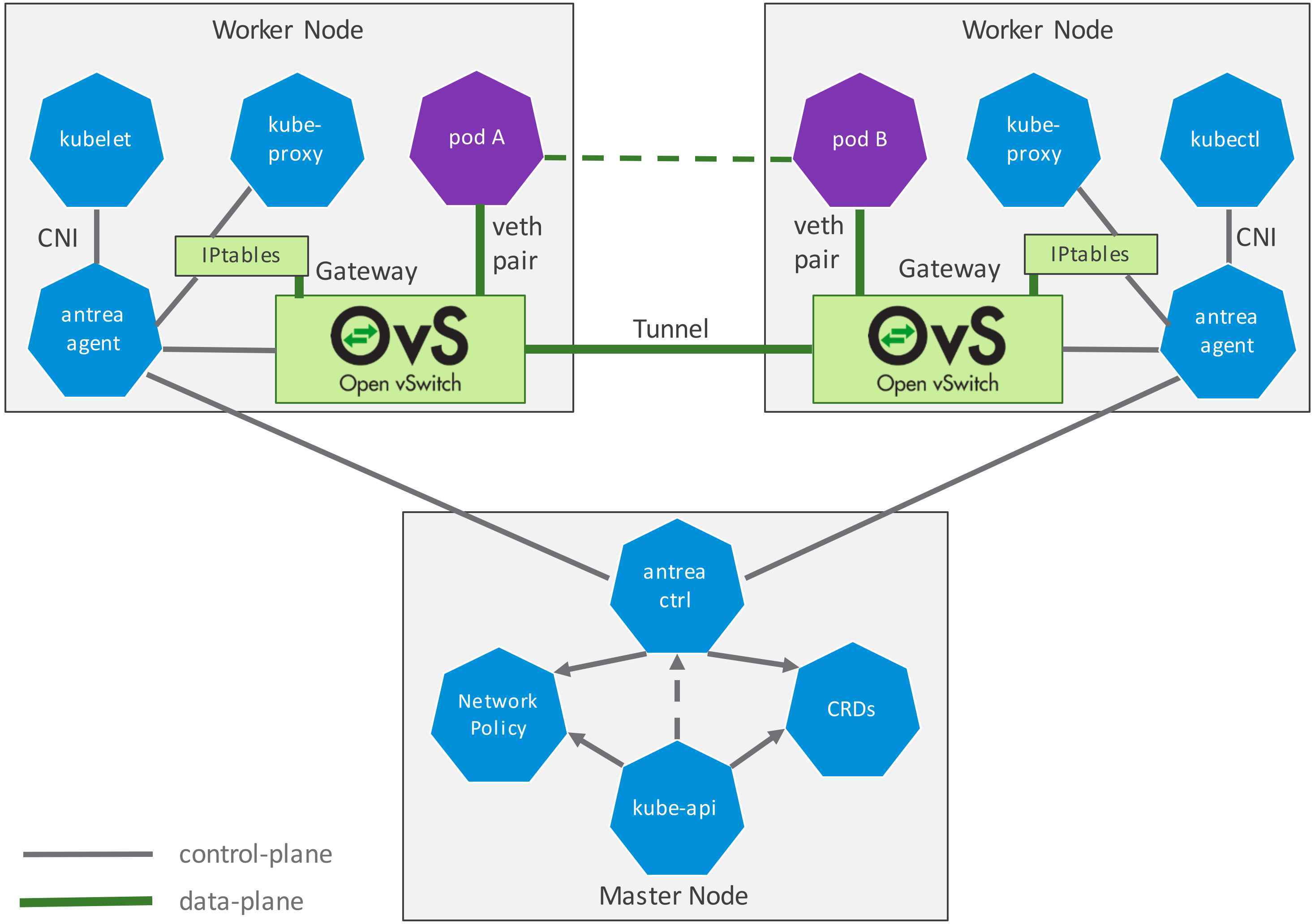
- DPDK, AF_XDP
- Hardware offload available across multiple vendors

Rich feature set:

- Advanced CLI tools
- Statistics, QoS
- Packet tracing

Project Antrea Architecture

Open vSwitch provides a flexible and performant data plane.



Supports K8S cluster networking

Antrea Agent

- Manages Pod network interfaces and OVS bridge.
- Creates overlay tunnels across Nodes.
- Implements NetworkPolicies with OVS.

Antrea Controller

- Computes K8s NetworkPolicies and publishes the results to Antrea Agents.

Open vSwitch as dataplane

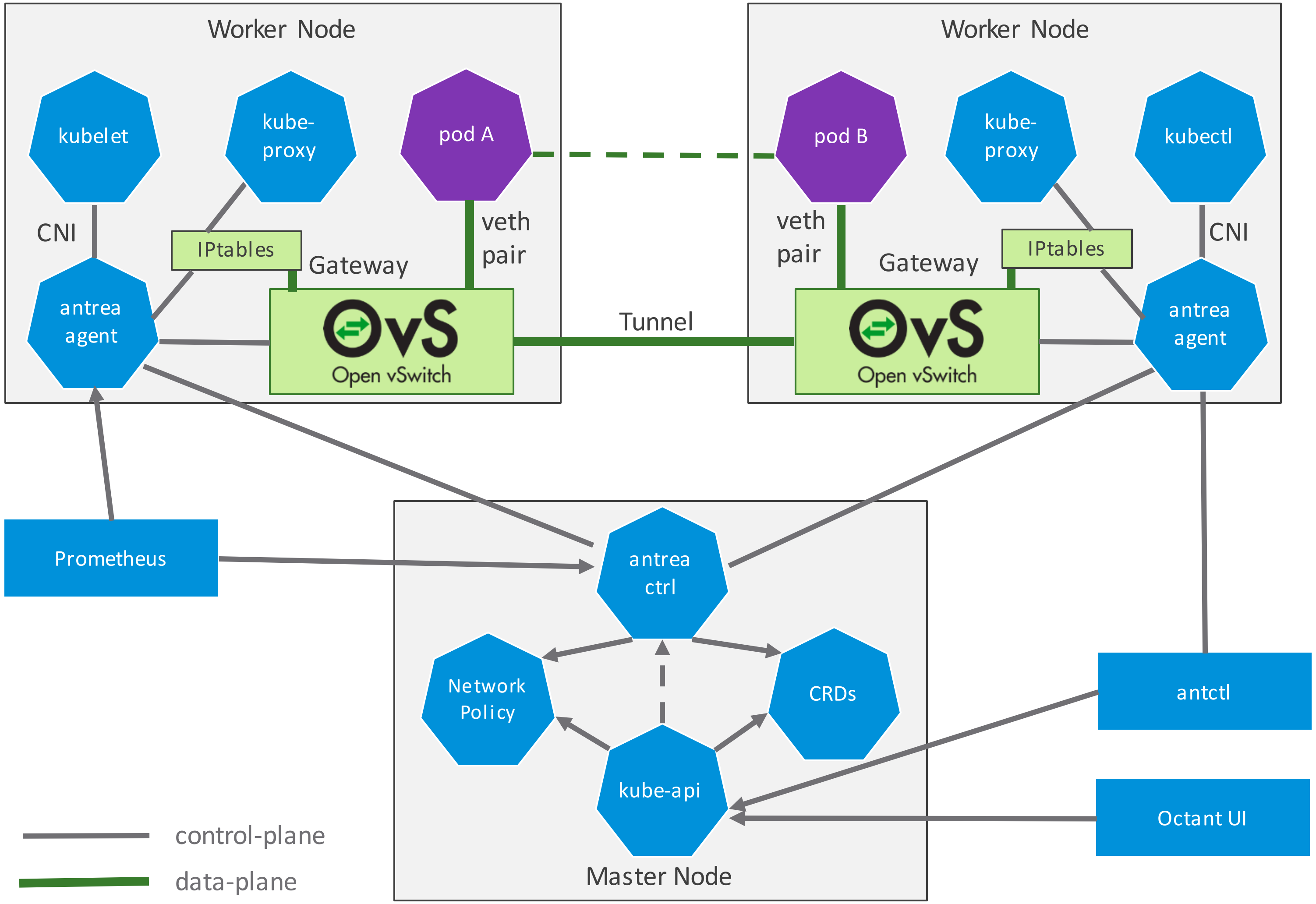
- Antrea Agent programs Open vSwitch with OpenFlow flows.
- Geneve, VXLAN, GRE, or STT tunnel between nodes
- Also supports policy-only and no-encap modes

Built with K8S technologies

- Leverages K8S and K8S solutions for API, UI, deployment, control plane, and CLI.
- Antrea Controller and Agent are based on K8S controller and apiserver libs.

Project Antrea Architecture

Component Review



Octant UI Plugin

- Shows Antrea runtime information (CRDs).
- Diagnostic traceflow visualization.

antctl – CLI for debugging

- Connects to Agent Agent or Controller.
- Packet tracing / Support bundle / etc.

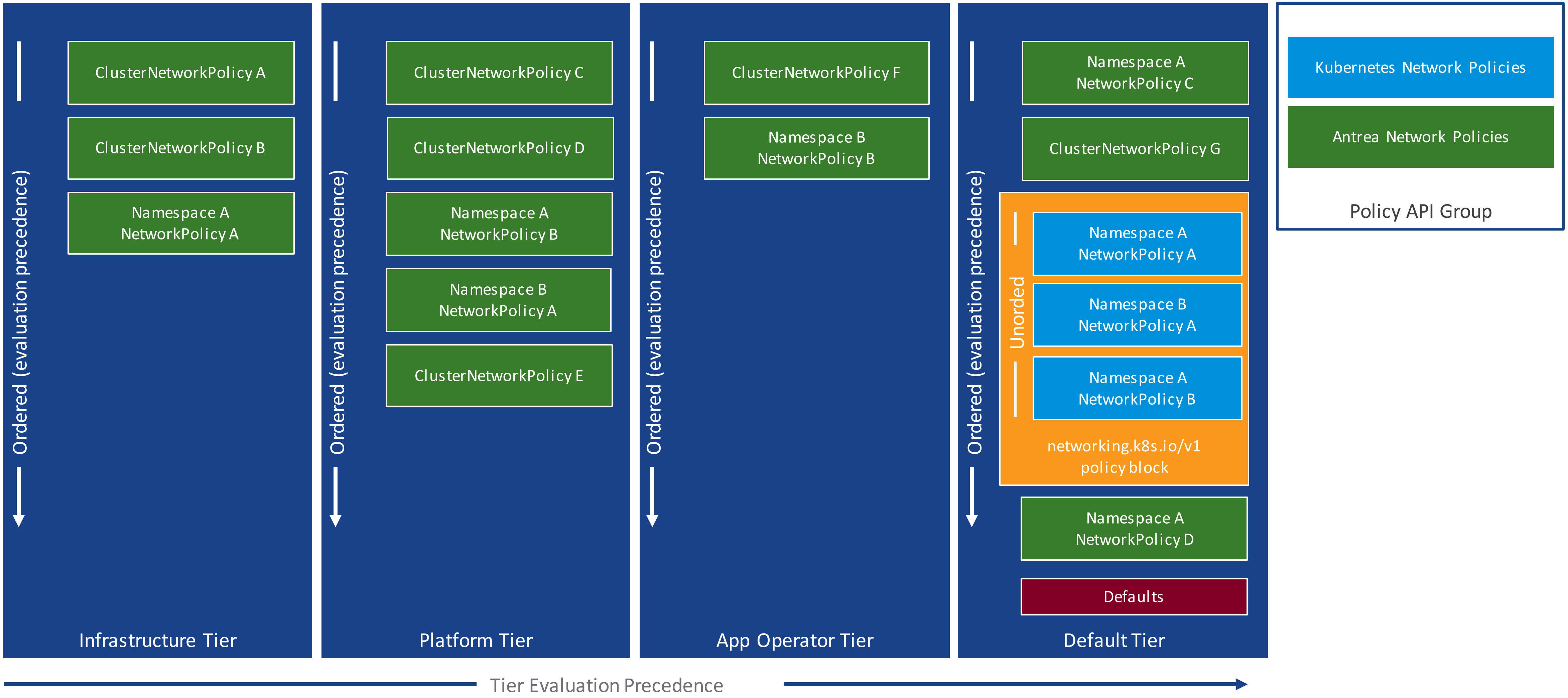
Prometheus metrics available

All bits (including OVS daemons) in a Docker image.

All components are deployed using K8S manifests.

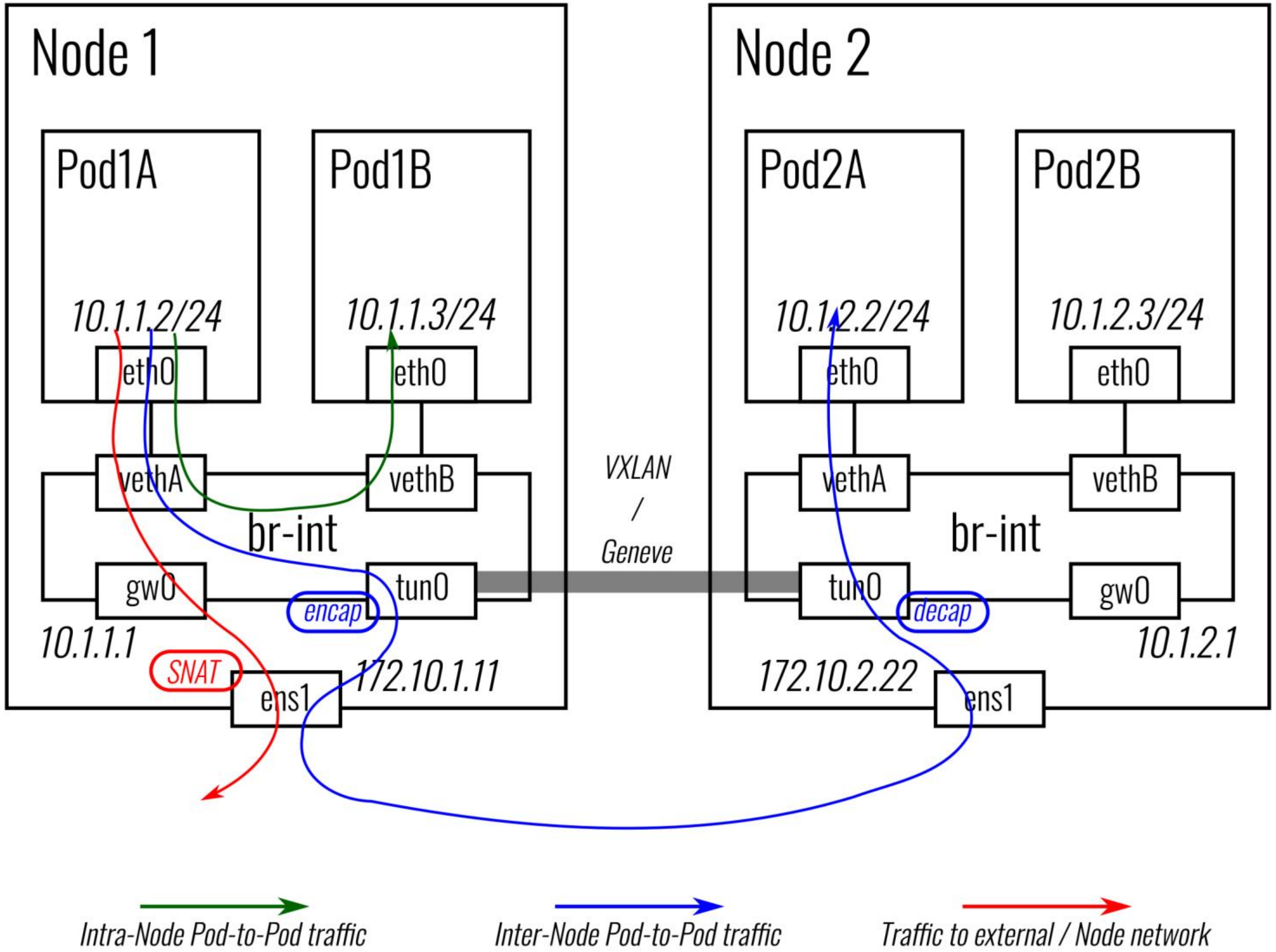
Policy Model

Antrea will allow native and Kubernetes policies to co-exist.



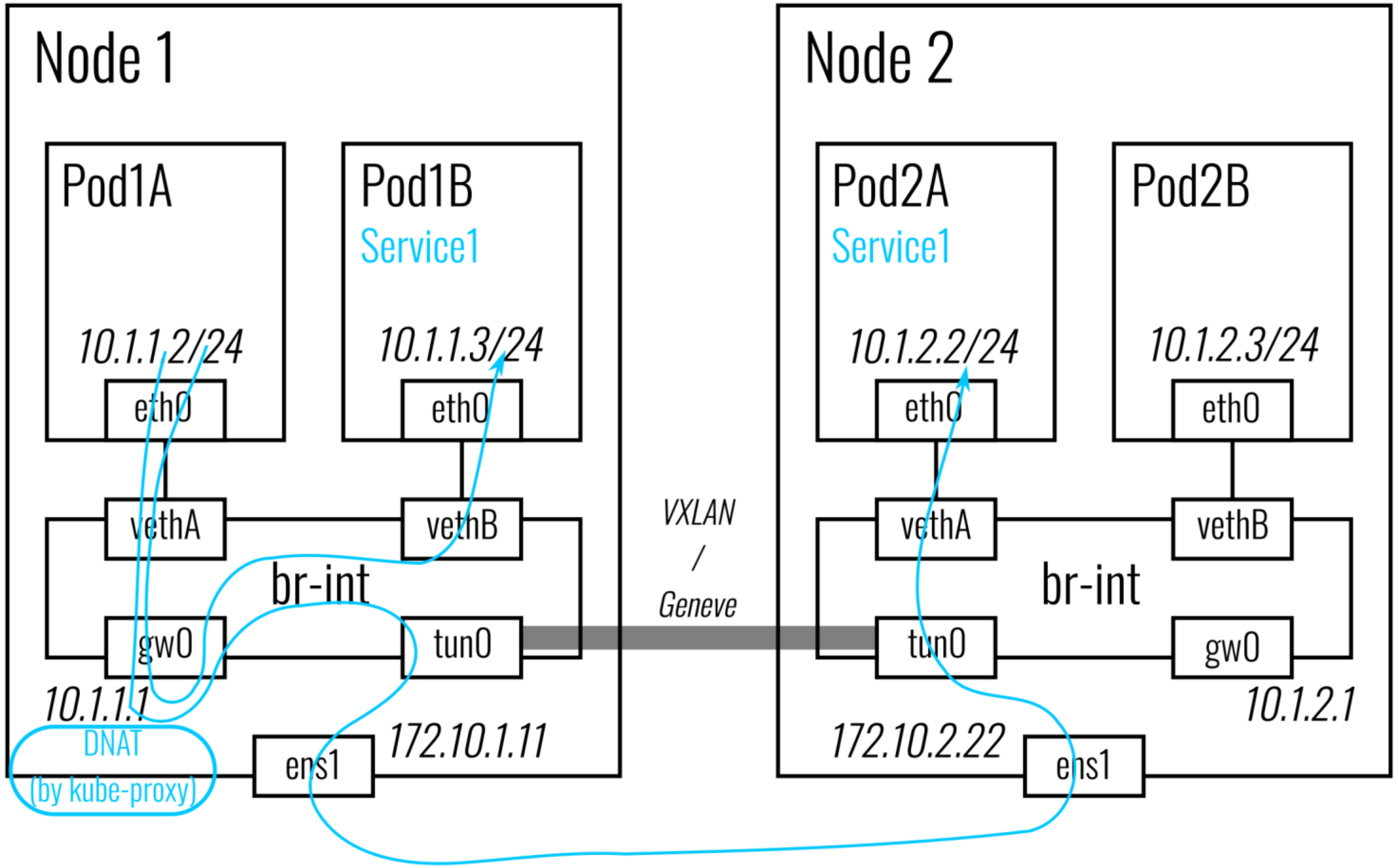
Ordered (evaluation precedence)
ClusterNetworkPolicy A
ClusterNetworkPolicy B
Namespace A NetworkPolicy A
Infrastructure Tier
Ordered (evaluation precedence)
ClusterNetworkPolicy C
ClusterNetworkPolicy D
Namespace A NetworkPolicy B
Namespace B NetworkPolicy A
ClusterNetworkPolicy E
Platform Tier
Ordered (evaluation precedence)
ClusterNetworkPolicy F
Namespace B NetworkPolicy B
App Operator Tier
Ordered (evaluation precedence)
Namespace A NetworkPolicy C
ClusterNetworkPolicy G
Unorded
Namespace A NetworkPolicy A
Namespace B NetworkPolicy A
Namespace A NetworkPolicy B
networking.k8s.io/v1 policy block
Namespace A NetworkPolicy D
Defaults
Default Tier
Kubernetes Network Policies
Antrea Network Policies
Policy API Group
Tier Evaluation Precedence

Traffic Walk (in “encap” mode)



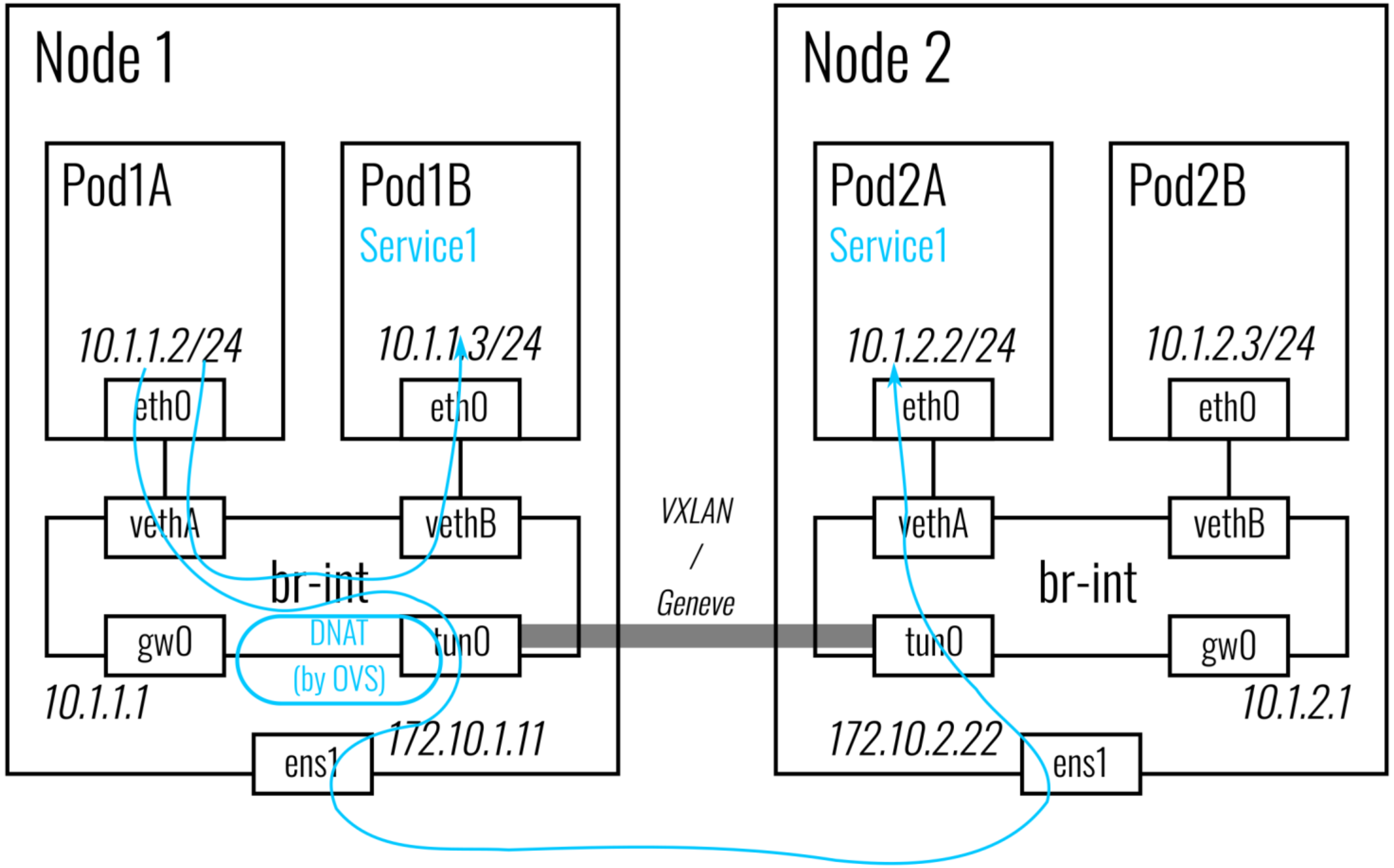
Traffic Walk: ClusterIP Services

Delegating to kube-proxy



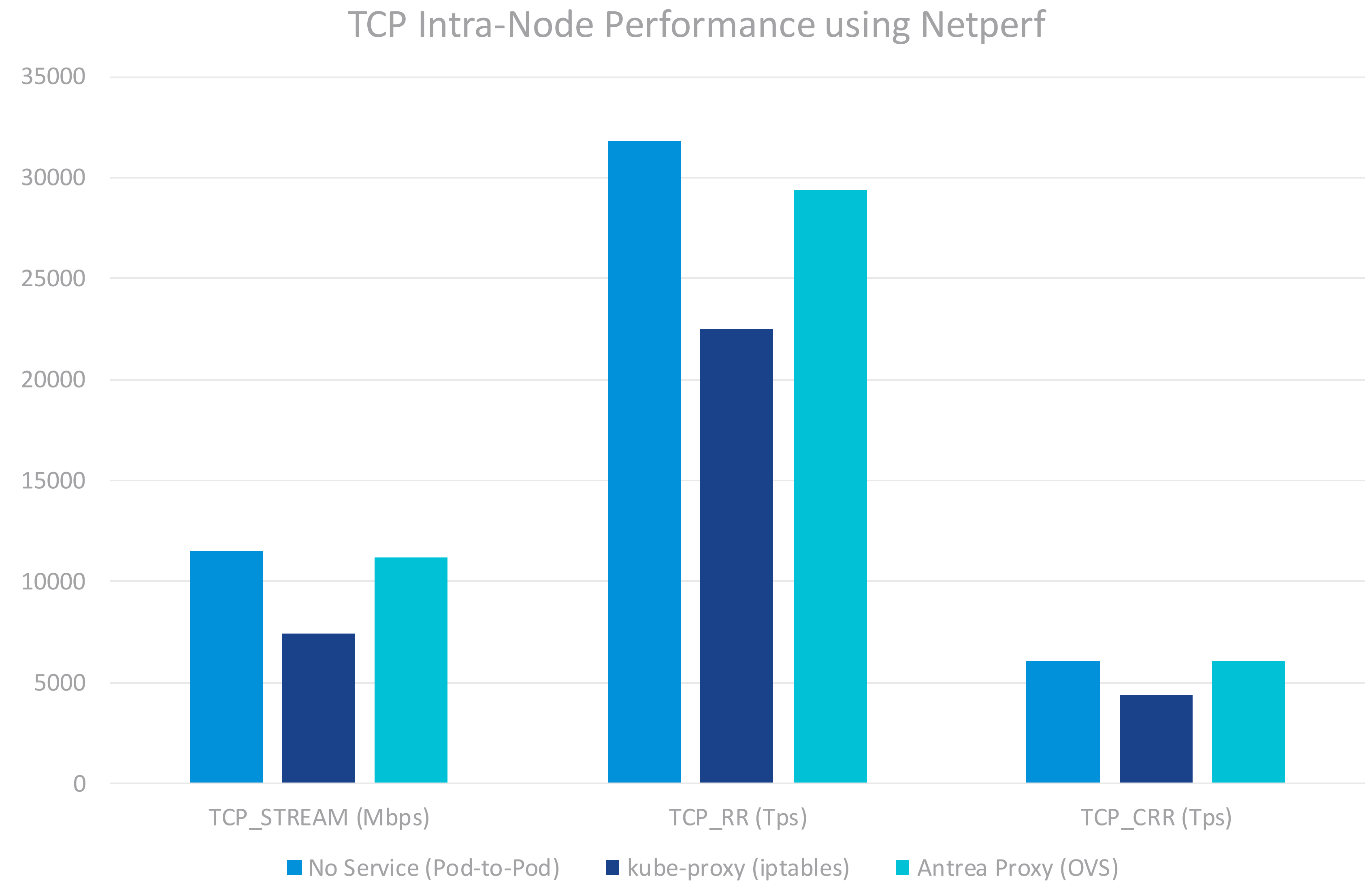
Traffic Walk: ClusterIP Services in OVS

New in v0.8.0: ClusterIP without kube-proxy



ClusterIP Services in OVS

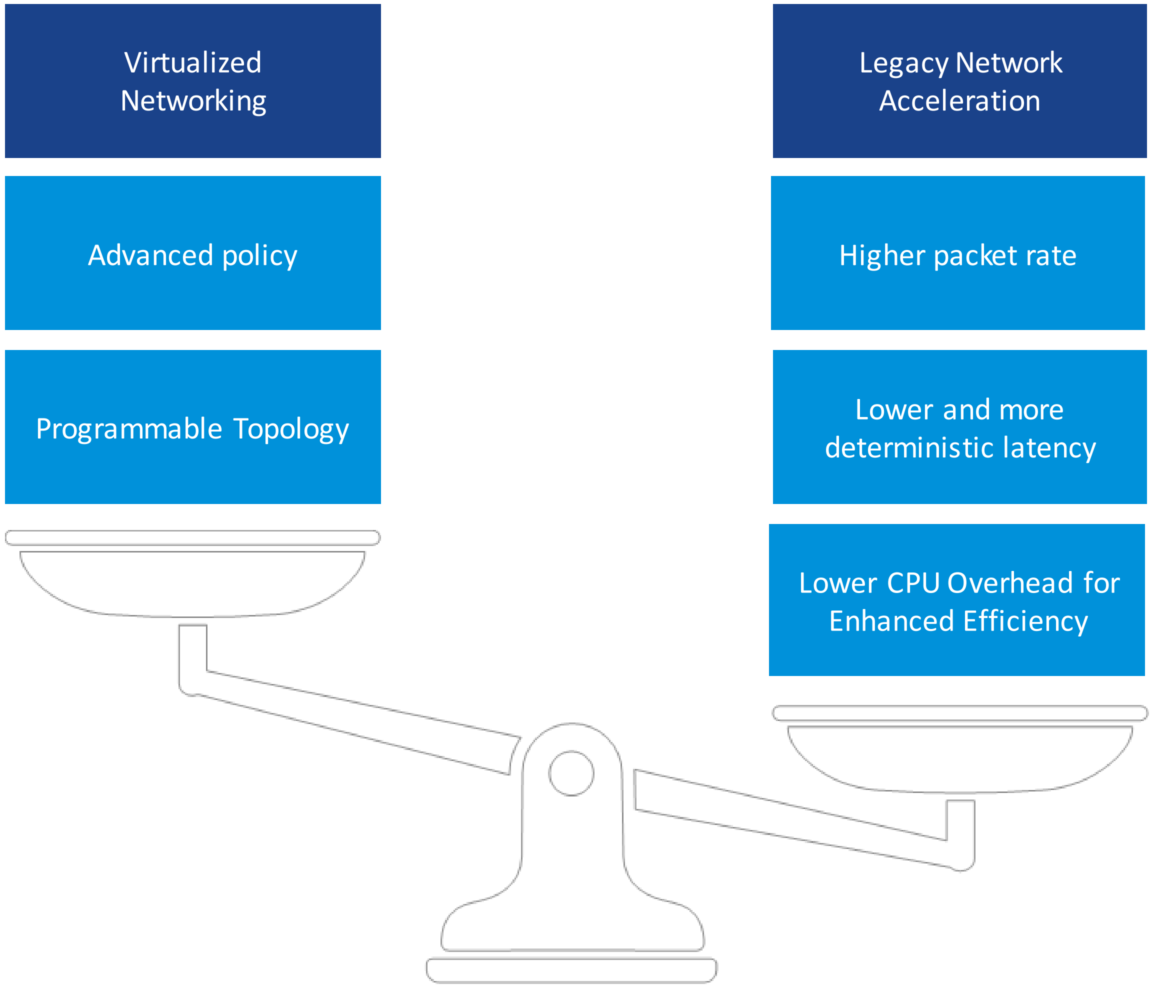
”Antrea Proxy”



Hardware Acceleration

No Tradeoff between Virtualized and Accelerated Networking

Decision used to be Either/Or



Virtualized Networking

Advanced policy

Programmable Topology

Legacy Network Acceleration

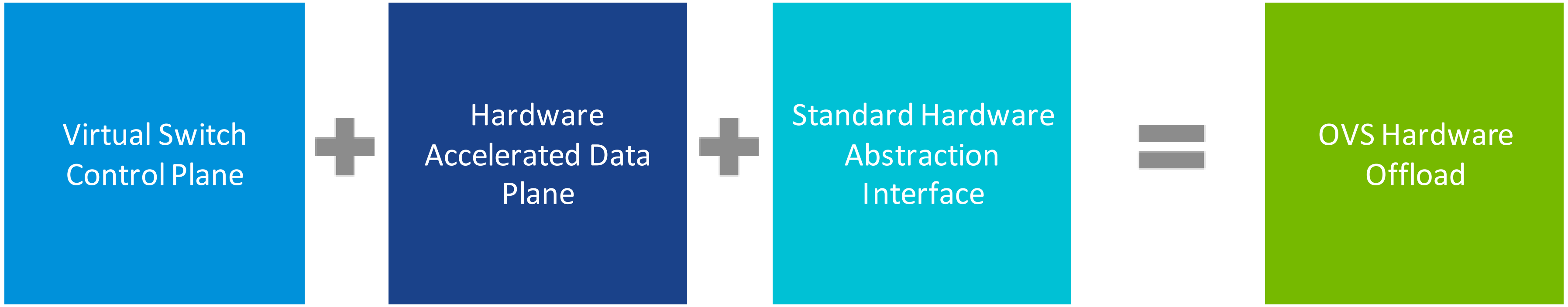
Higher packet rate

Lower and more deterministic latency

Lower CPU Overhead for Enhanced Efficiency

Introducing OVS Hardware Offload

Now we can have Both/And

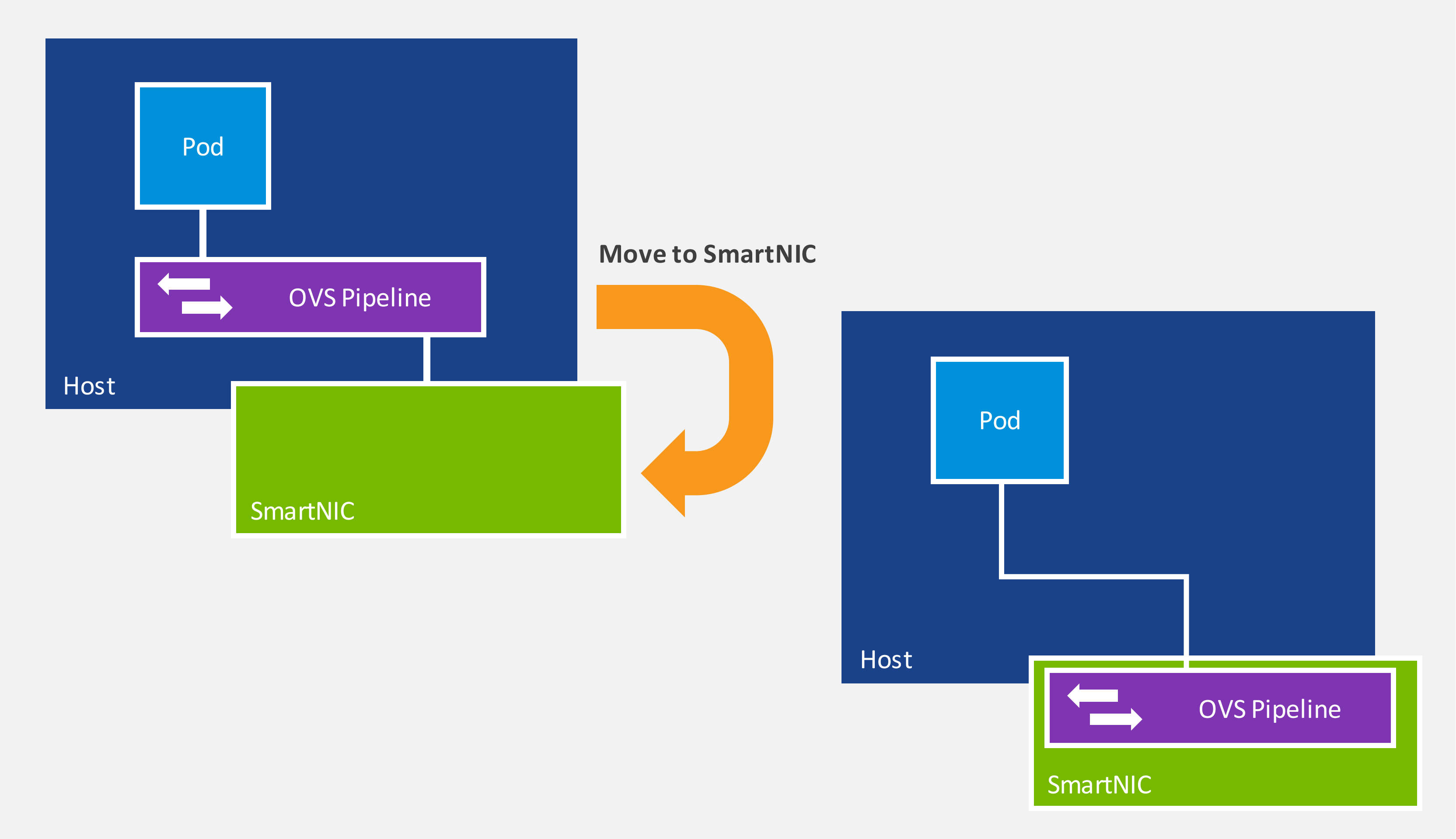


Virtual Switch Control Plane + Hardware Accelerated Data Plane + Standard Hardware Abstraction Interface = OVS Hardware Offload

- ✓ Best of both worlds: Enable hardware-accelerated networking data plane with programmable control plane
- ✓ Up to 10X network performance with practically zero CPU utilization

OVS Hardware Offload

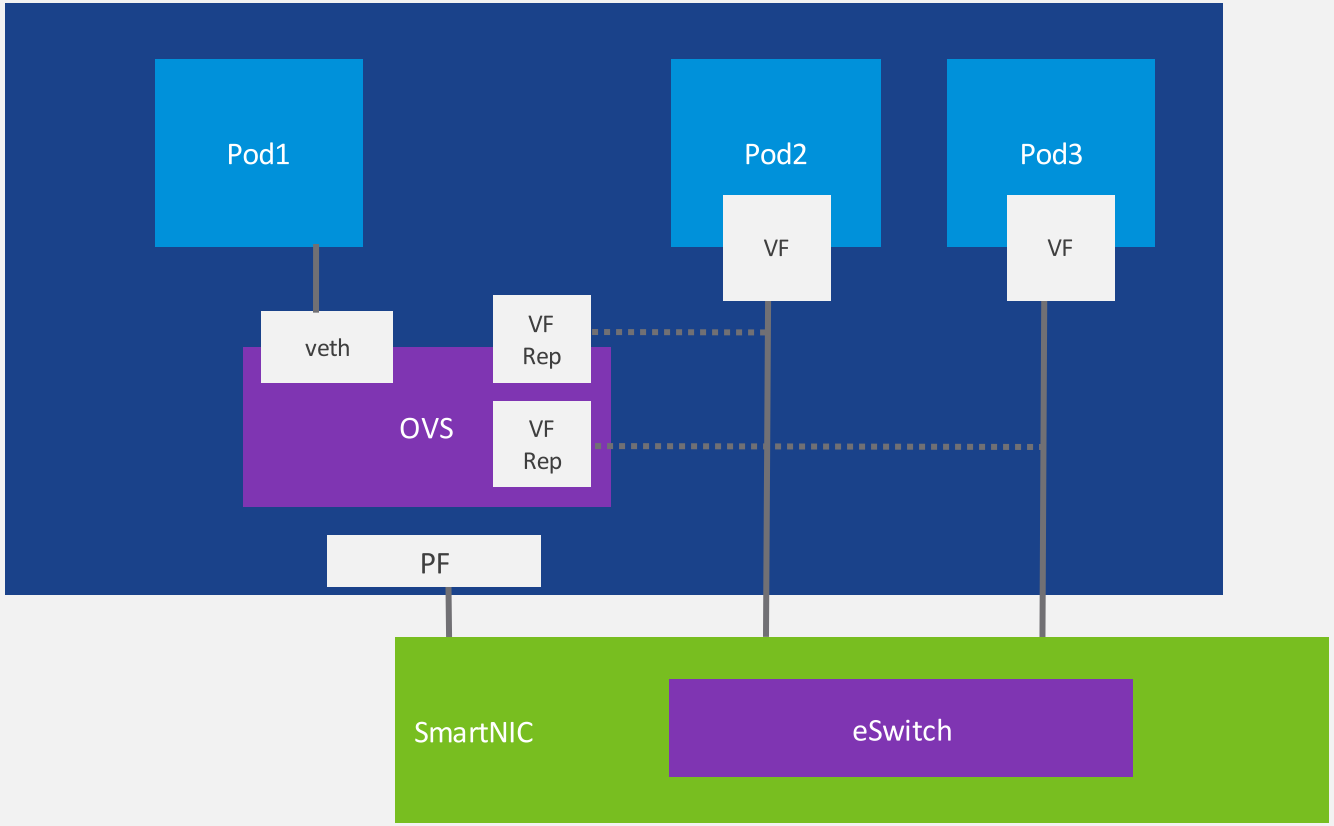
Move OVS OpenFlow Processing to a SmartNIC



Typically, OVS flows are processed on a bare metal host, VM or hypervisor.

- The OVS kernel or user space component consumes CPU
- Less CPU resources available for apps
- Moving OVS processing to the SmartNIC frees up CPU

SR-IOV Definitions



SR-IOV – Single Root I/O Virtualization

PF – Physical Function. The physical Ethernet controller that supports SR-IOV.

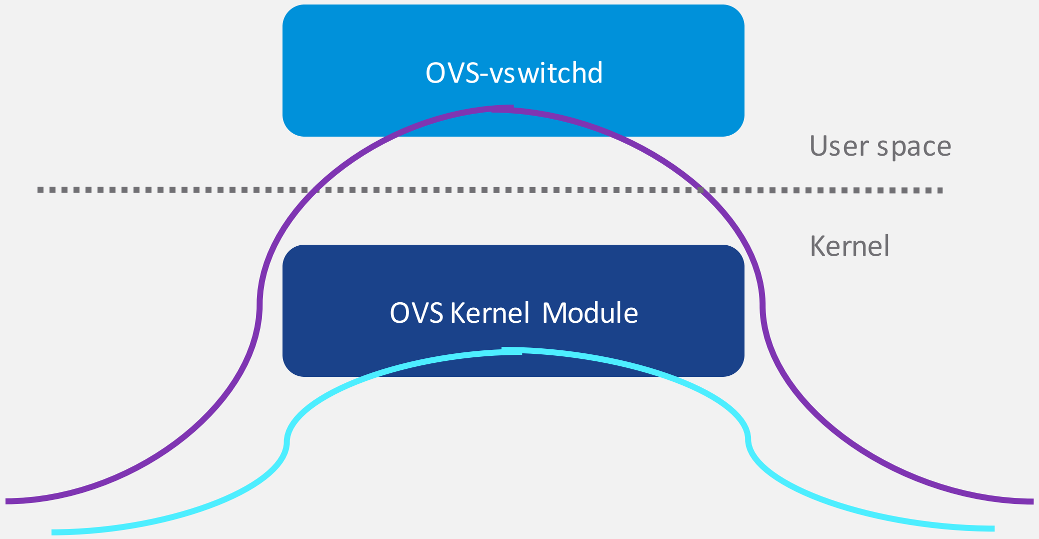
VF – Virtual Function. The virtual PCIe device created from a physical Ethernet controller.

VF Representor – Port representor of the Virtual Function

How OVS Hardware Offload Works

Software only OVS Implementation

High latency, low bandwidth, CPU intensive



OVS-vswitchd

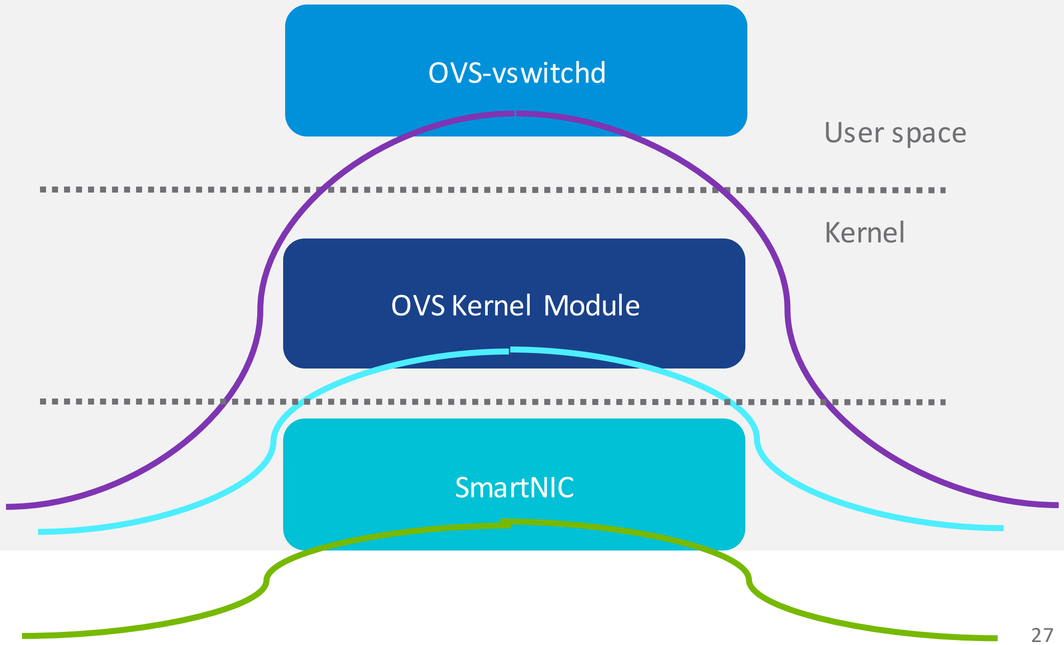
User space

Kernel

OVS Kernel Module

Software-defined, Hardware-accelerated

Low latency, high bandwidth, CPU efficient



OVS-vswitchd

User space

Kernel

OVS Kernel Module

SmartNIC

First flow packet

Fallback FRWD packet

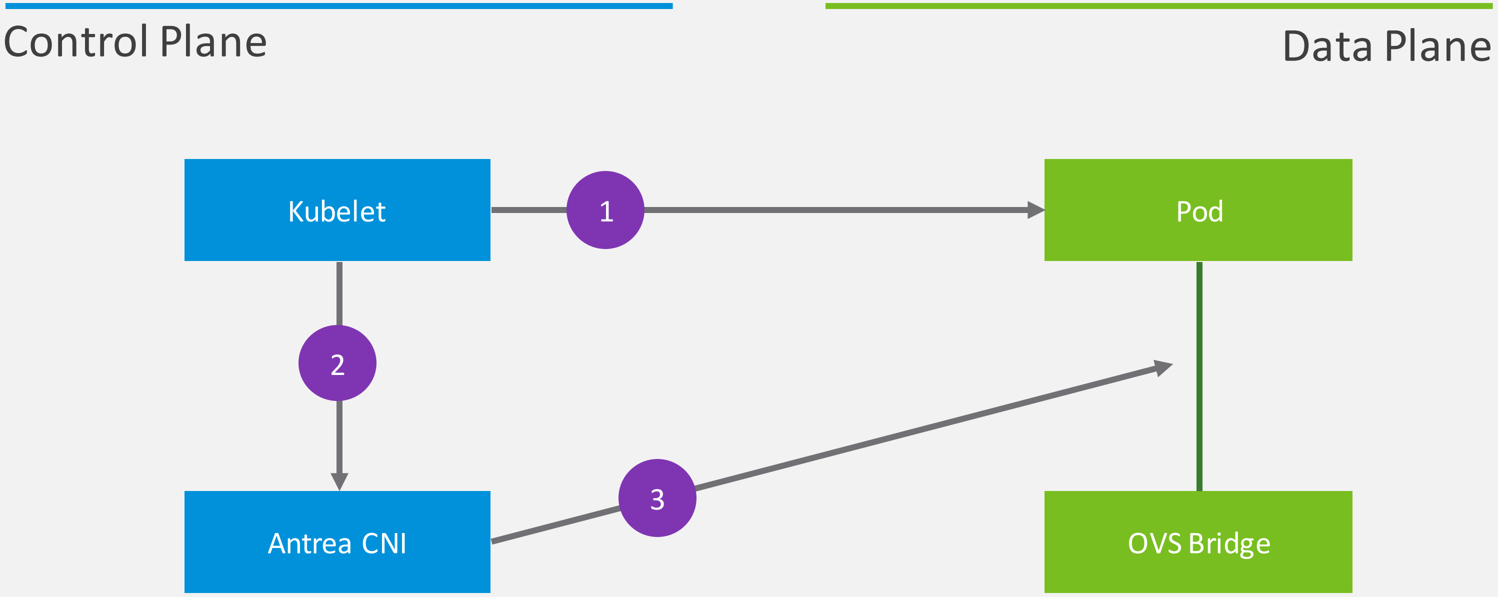
Hardware offloaded packets

OVS Hardware Offload

Requires additional CNI plugins and SR-IOV VF enablement on NIC

- Multus
- SR-IOV Network Device Plugin
- Antrea

Antrea CNI Plumbing Without Offload

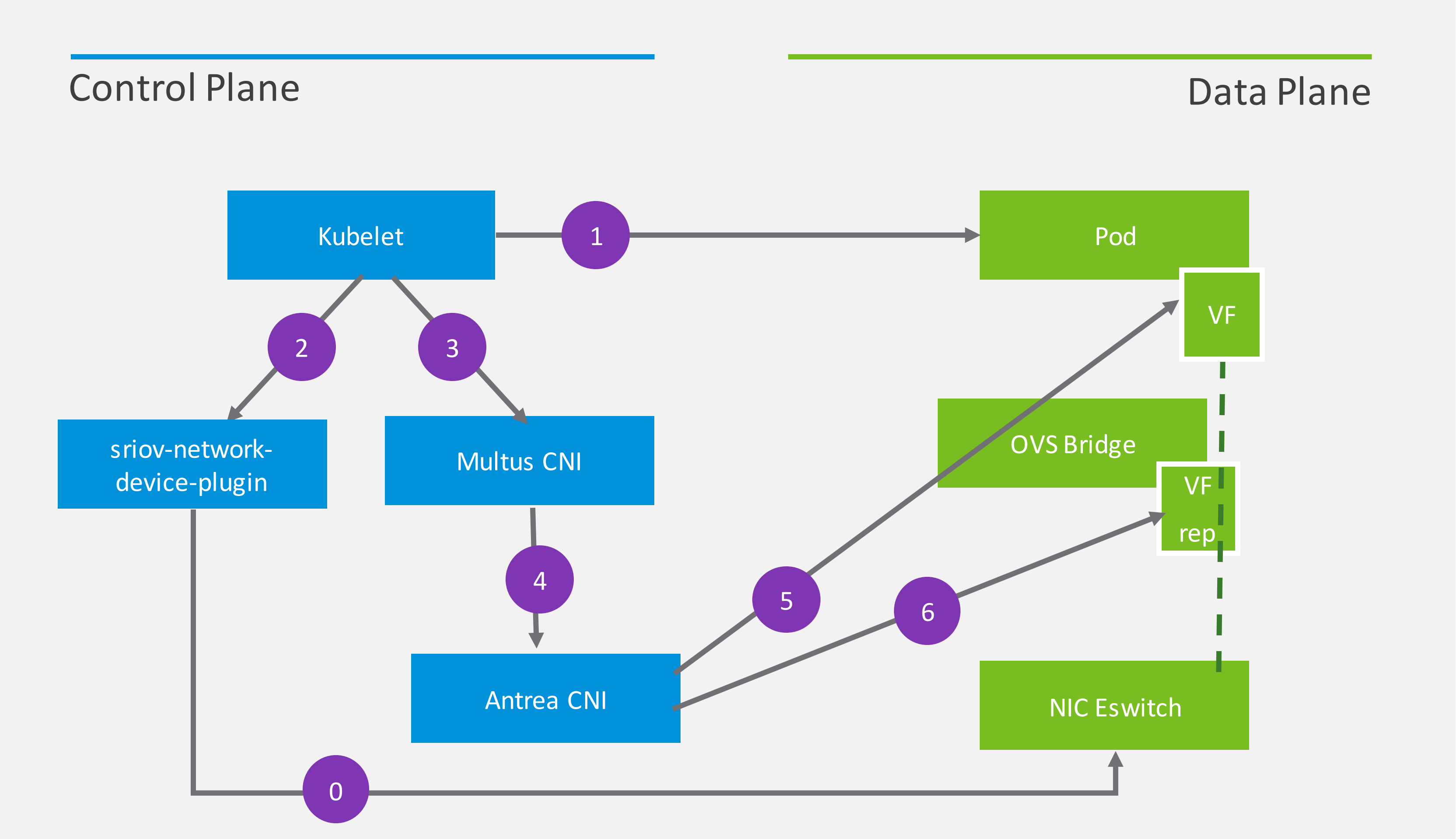


1. Kubelet creates pod

2. Kubelet calls CNI to add pod to network

3. Antrea CNI provisions veth pair
   - eth0 in pod network namespace
   - connect other end to OVS bridge port

Antrea CNI Plumbing With Offload

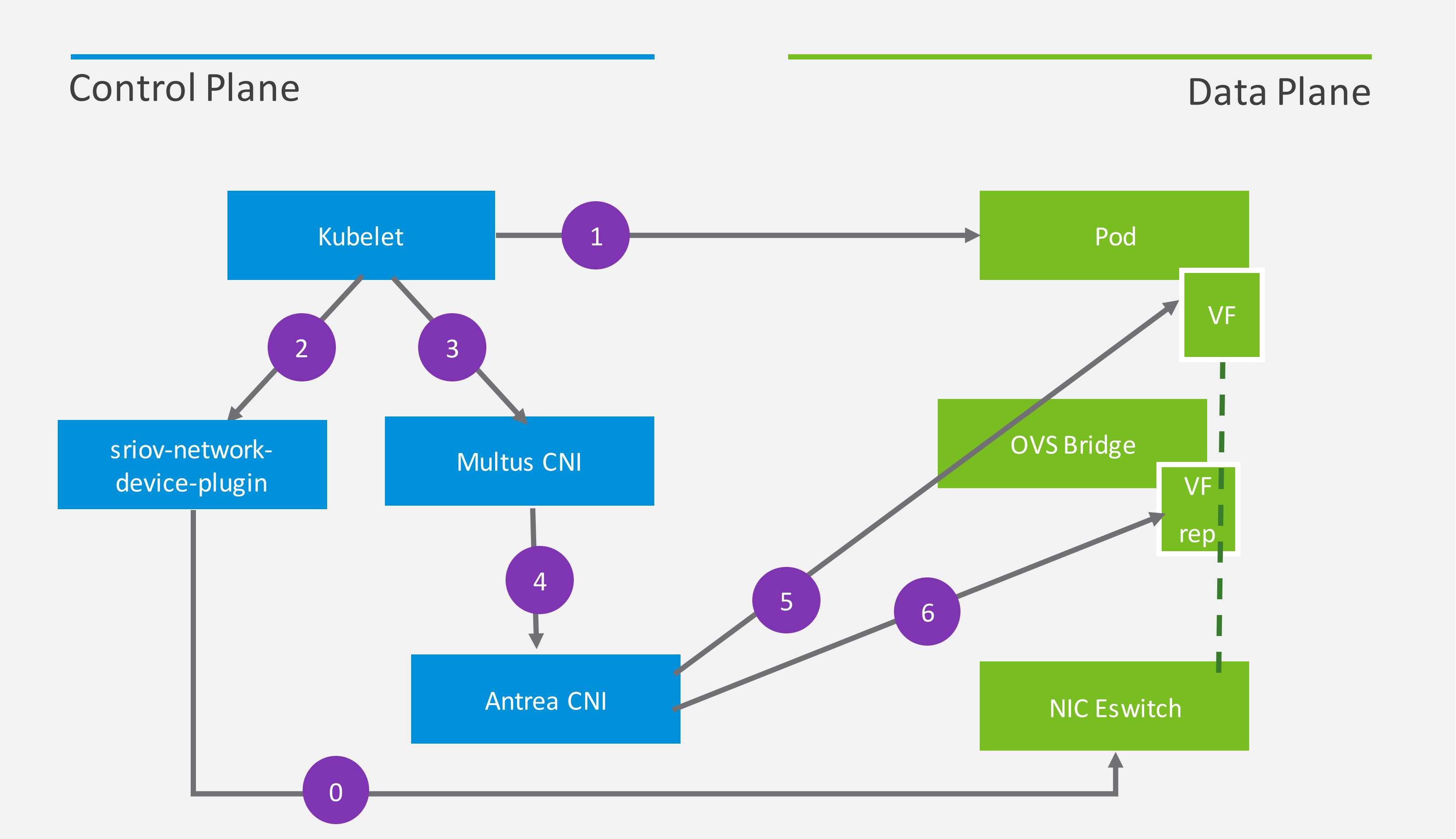


0. VF pool initialization

1. Kubelet creates pod

2. SR-IOV Device Plugin allocates VF PCI address from VF pool to satisfy resource request on pod creation (exposed as environment variable)

Antrea CNI Plumbing With Offload



3. Kubelet calls CNI (Multus) to add pod to network

4. Multus CNI looks up the allocated SR-IOV VF PCI Address and passes it as extra CNI args to Antrea CNI

5. Antrea CNI moves the VF netdevice to the pod network namespace and renames to eth0

6. Antrea CNI plugs the VF representor intoto the OVS br-int bridge

Demo - Setup Details

- 3 servers – 1 master and 2 workers
- Linux CentOS 7.7
- Kubernetes 1.18
- Linux 5.7 kernel
- Antrea v0.8.0 with offload patches
- NVIDIA Mellanox ConnectX-5 SmartNICs

Demo – Flow

- Deploy SR-IOV network device plugin
- Deploy Multus CNI
- Deploy Antrea
- Create veth Pod
- Create offload Pod
- Run iperf3 between 2 veth pods
- Run iperf3 between 2 offload pods

Demo

Antrea Roadmap

Features Available Through v0.8.0

Overlay Modes

Geneve, VXLAN,
STT, GRE

Policy-only
(CNI chaining)

No-encap

Hybrid

Clouds

Private Cloud:
bare metal, vSphere, other VM, kind

Public Cloud:
Azure – AKS Engine
AWS – EC2, EKS (beta)
Google – GKE (alpha)

Service Load Balancing

kube-proxy support in IPVS and IPtables modes

OVS based kube-proxy implementation

Features Available Through v0.8.0

Network Policy

networking.k8s.io NetworkPolicy v1 (upstream)

Native Policy: ClusterNetworkPolicy

Security

Server certificate verification for Controller APIs (user provided or generated)

Spoof Guard

IPsec over GRE

Visibility

Prometheus Metrics & Monitoring CRDs

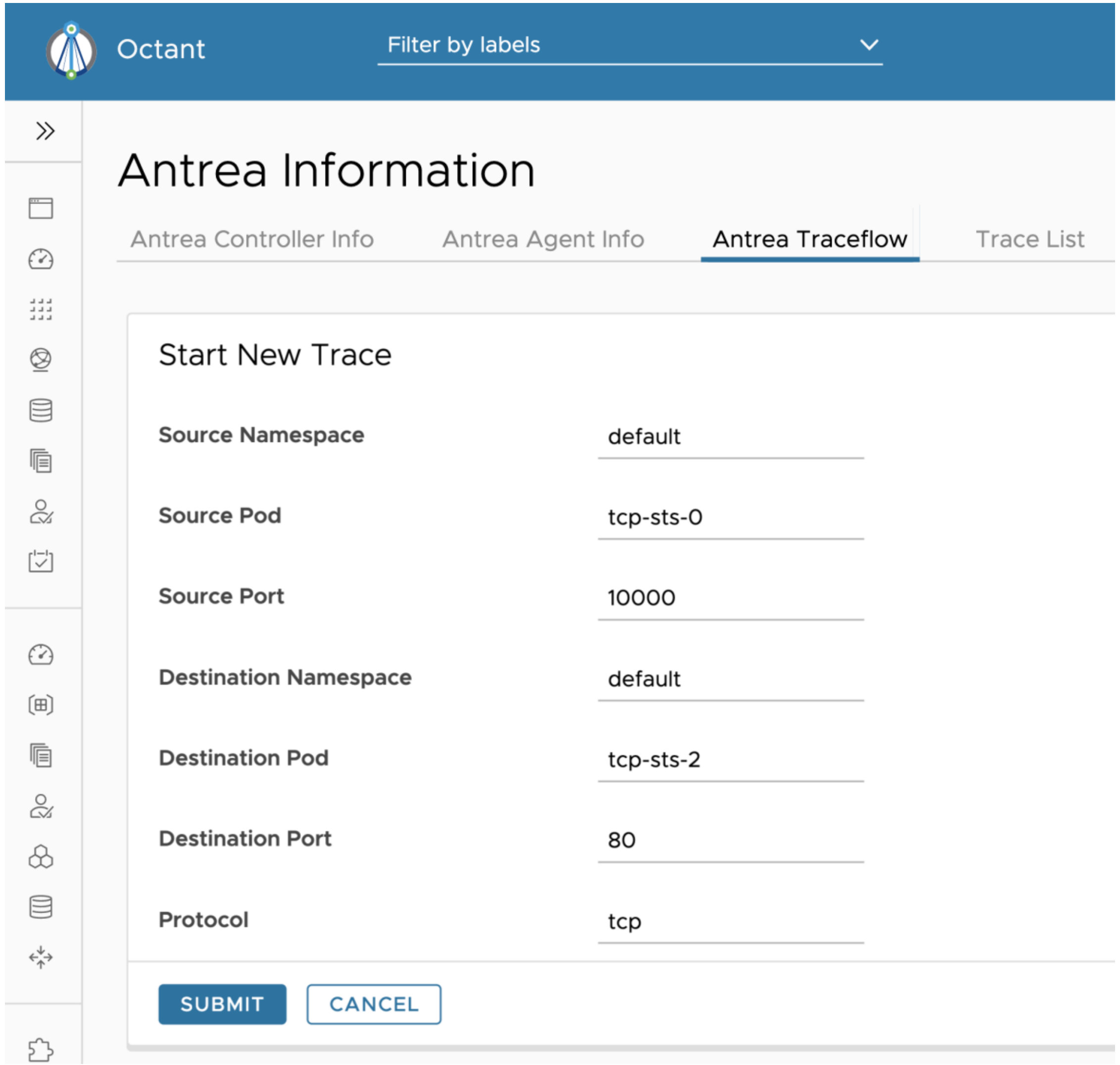
Traceflow

Support bundle generation

antctl CLI & Octant UI Plugin

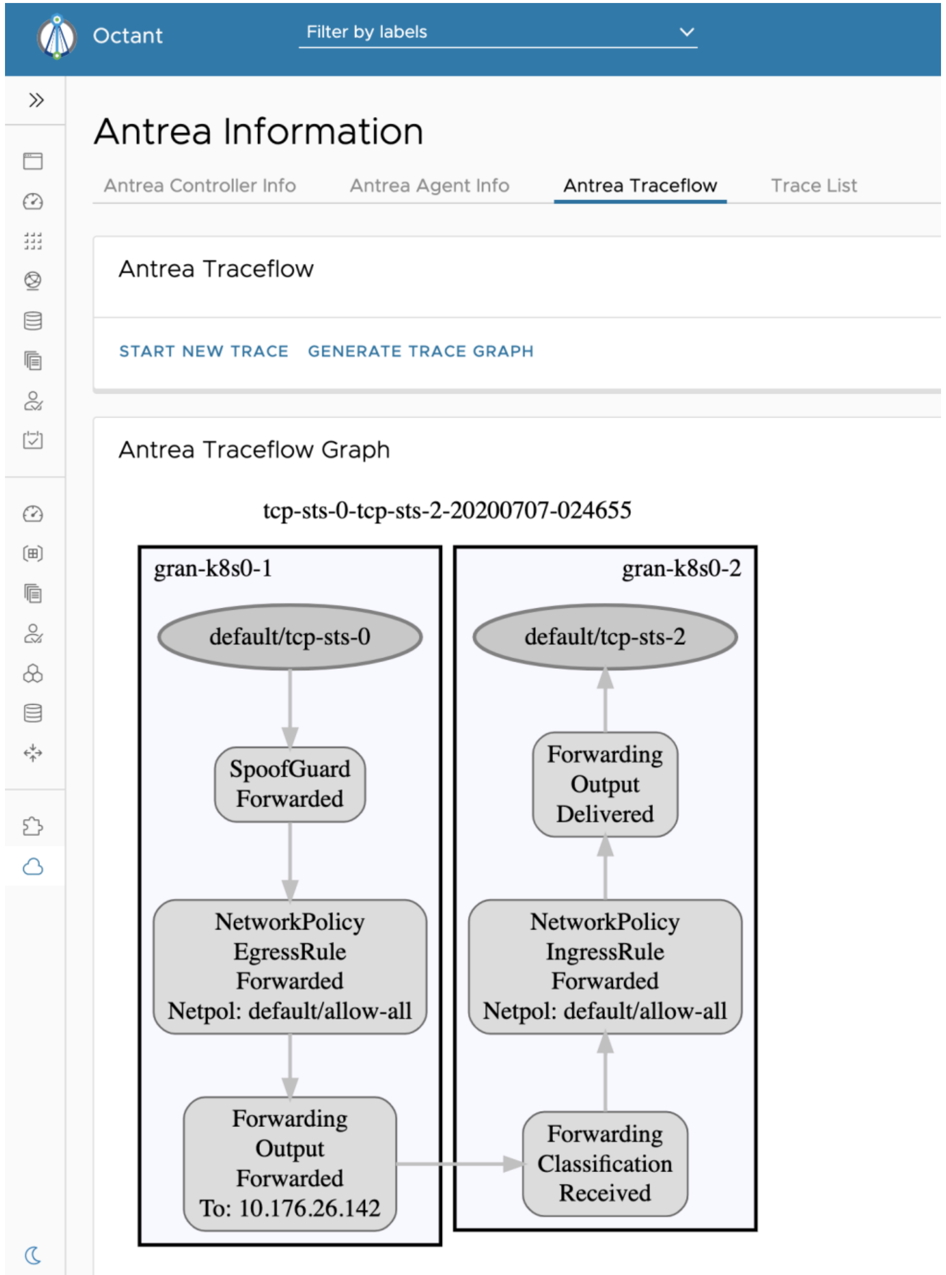
Traceflow

Request 1: traffic is allowed



Start New Trace

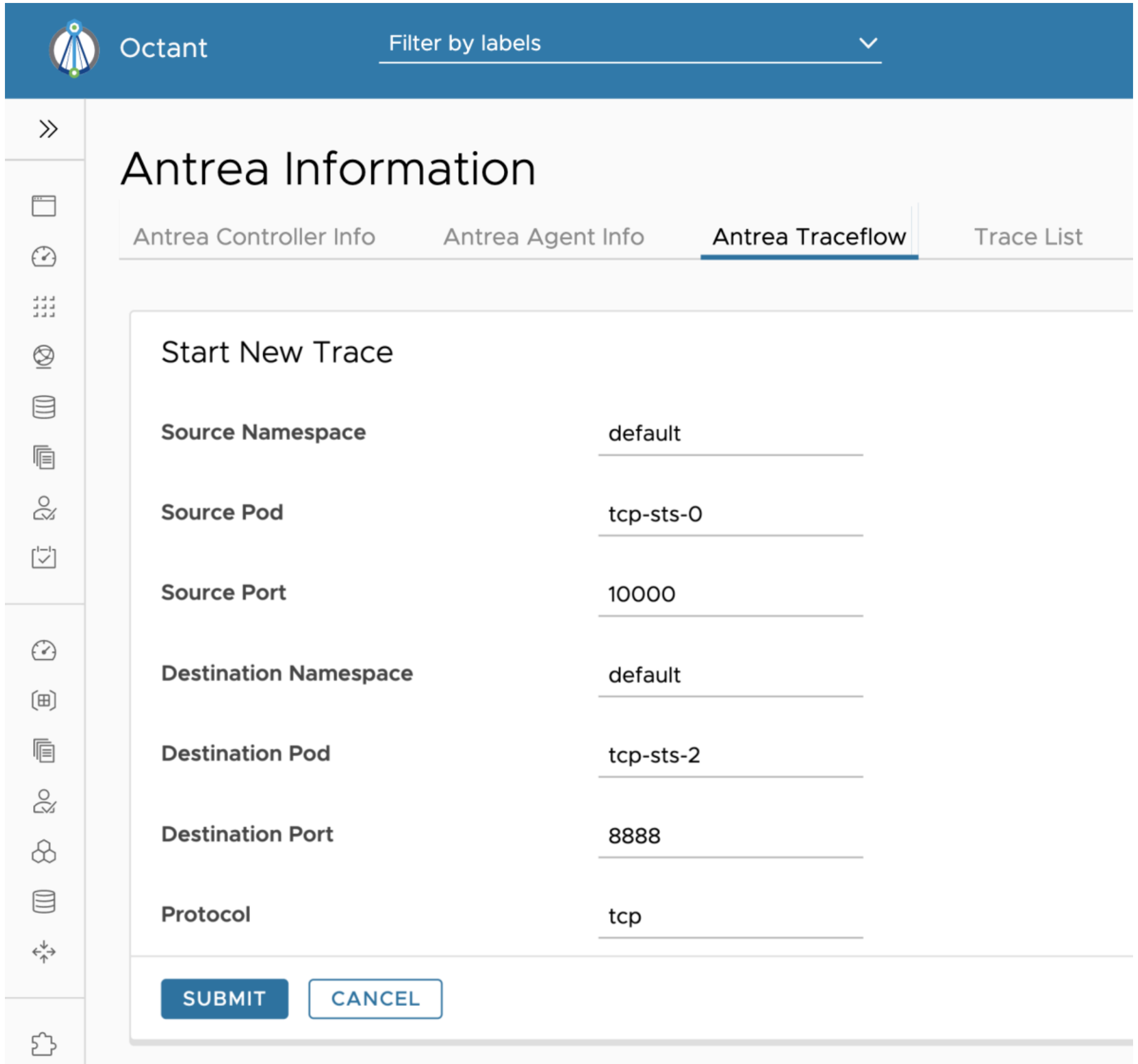
Source Namespace	default
Source Pod	tcp-sts-0
Source Port	10000
Destination Namespace	default
Destination Pod	tcp-sts-2
Destination Port	80
Protocol	tcp



Octant
Filter by labels
Antrea Information
Antrea Controller Info
Antrea Agent Info
Antrea Traceflow
Trace List
Antrea Traceflow
START NEW TRACE
GENERATE TRACE GRAPH
Antrea Traceflow Graph
tcp-sts-0-tcp-sts-2-20200707-024655
gran-k8s0-1
default/tcp-sts-0
SpoofGuard Forwarded
NetworkPolicy EgressRule Forwarded Netpol: default/allow-all
Forwarding Output Forwarded To: 10.176.26.142
gran-k8s0-2
default/tcp-sts-2
Forwarding Output Delivered
NetworkPolicy IngressRule Forwarded Netpol: default/allow-all
Forwarding Classification Received

Traceflow

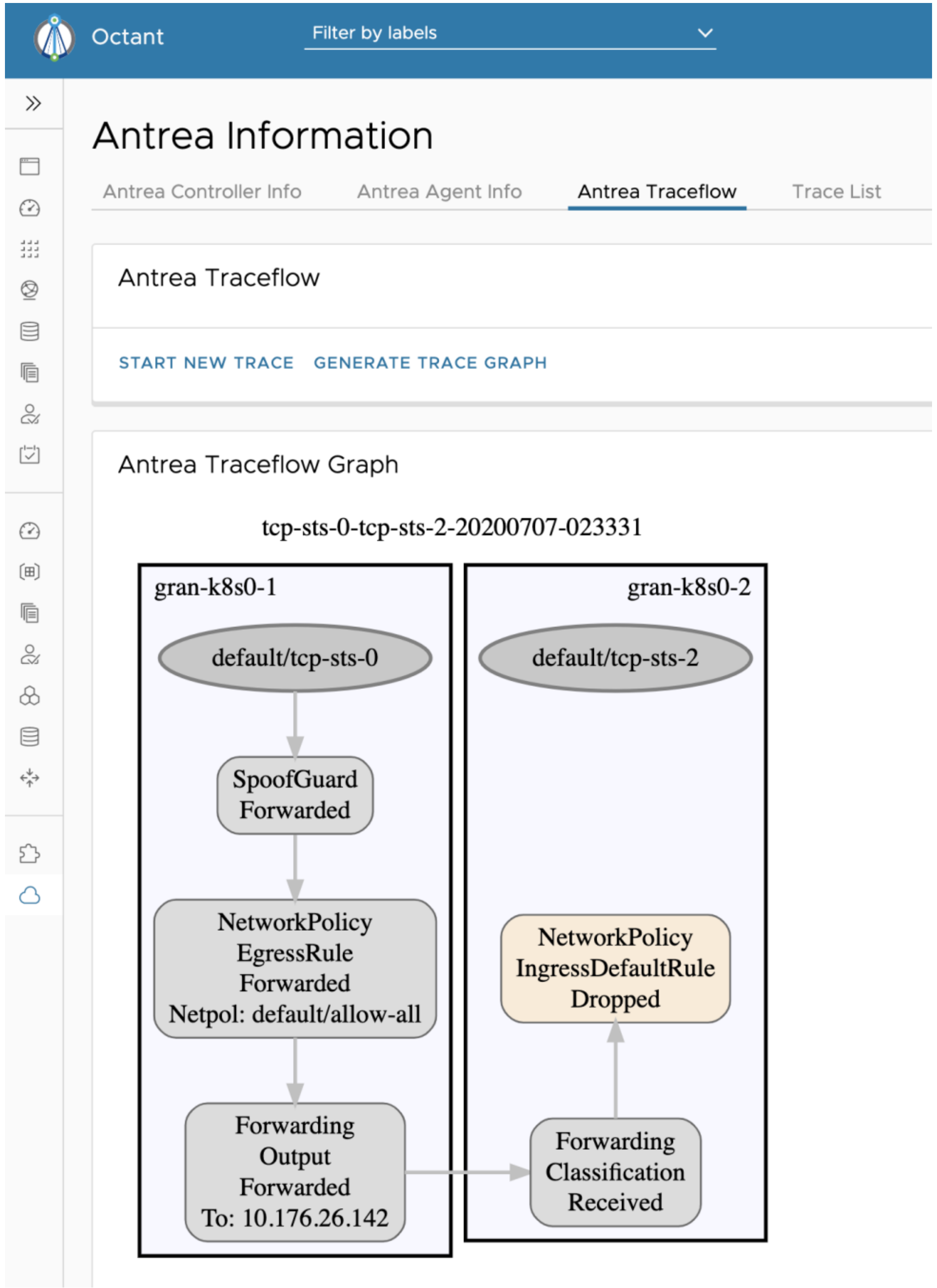
Request 2: traffic is denied



Antrea Information

Start New Trace

Source Namespace	default
Source Pod	tcp-sts-0
Source Port	10000
Destination Namespace	default
Destination Pod	tcp-sts-2
Destination Port	8888
Protocol	tcp


Octant
Filter by labels
Antrea Controller Info
Antrea Agent Info
Antrea Traceflow
Trace List

Antrea Information

Antrea Traceflow

START NEW TRACE
GENERATE TRACE GRAPH

Antrea Traceflow Graph

tcp-sts-0-tcp-sts-2-20200707-023331
gran-k8s0-1
default/tcp-sts-0
SpoofGuard Forwarded
NetworkPolicy EgressRule Forwarded Netpol: default/allow-all
Forwarding Output Forwarded To: 10.176.26.142
gran-k8s0-2
default/tcp-sts-2
NetworkPolicy IngressDefaultRule Dropped
Forwarding Classification Received

Features Available Through v0.8.0

Operating Systems

Linux

Windows Server 2019 (alpha)

Planned Features This Year

IPFIX flow data export

Advanced traffic matching and pod binding

Tiering to support multi-tenancy and delegation.

IPv6 dual-stack support

IPsec Offload

Expand support for KaaS and Cluster API providers

Enhanced data path including:
DPDK, SR-IOV, AF_XDP, VPP, and XDP

DNS egress filtering

Advanced IP Address Management

Named external endpoints with metadata

Extension mechanisms

Flow information export and visualization

Track all cluster traffic

- Number of connections
- Bandwidth for each connection
- Inter-Node bandwidth
- Aggregated Service bandwidth

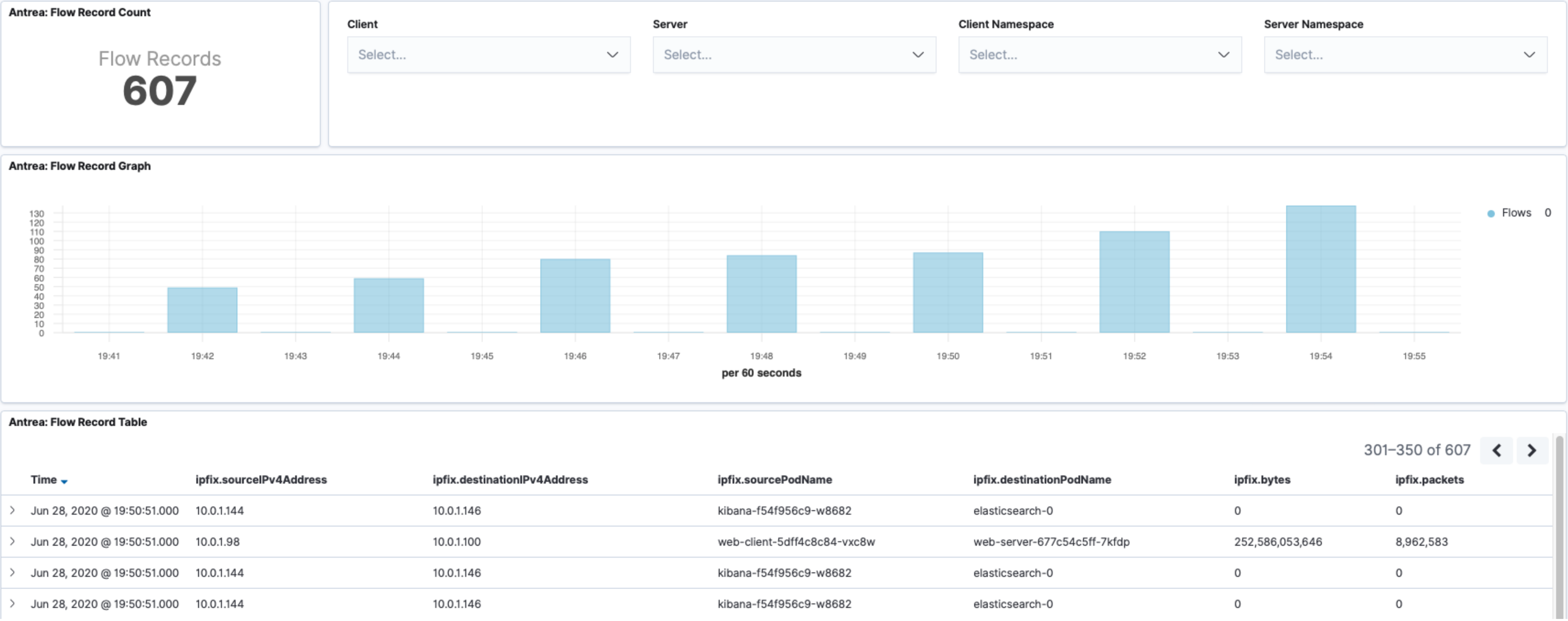
Complements Prometheus metrics

IPFIX records with K8s context (Namespace, Name, Labels, …)

Visualization using Elastic Stack

Flow information export

IPFIX Records



Antrea: Flow Record Count

Flow Records
607

Client: Select...

Server: Select...

Client Namespace: Select...

Server Namespace: Select...

Antrea: Flow Record Graph

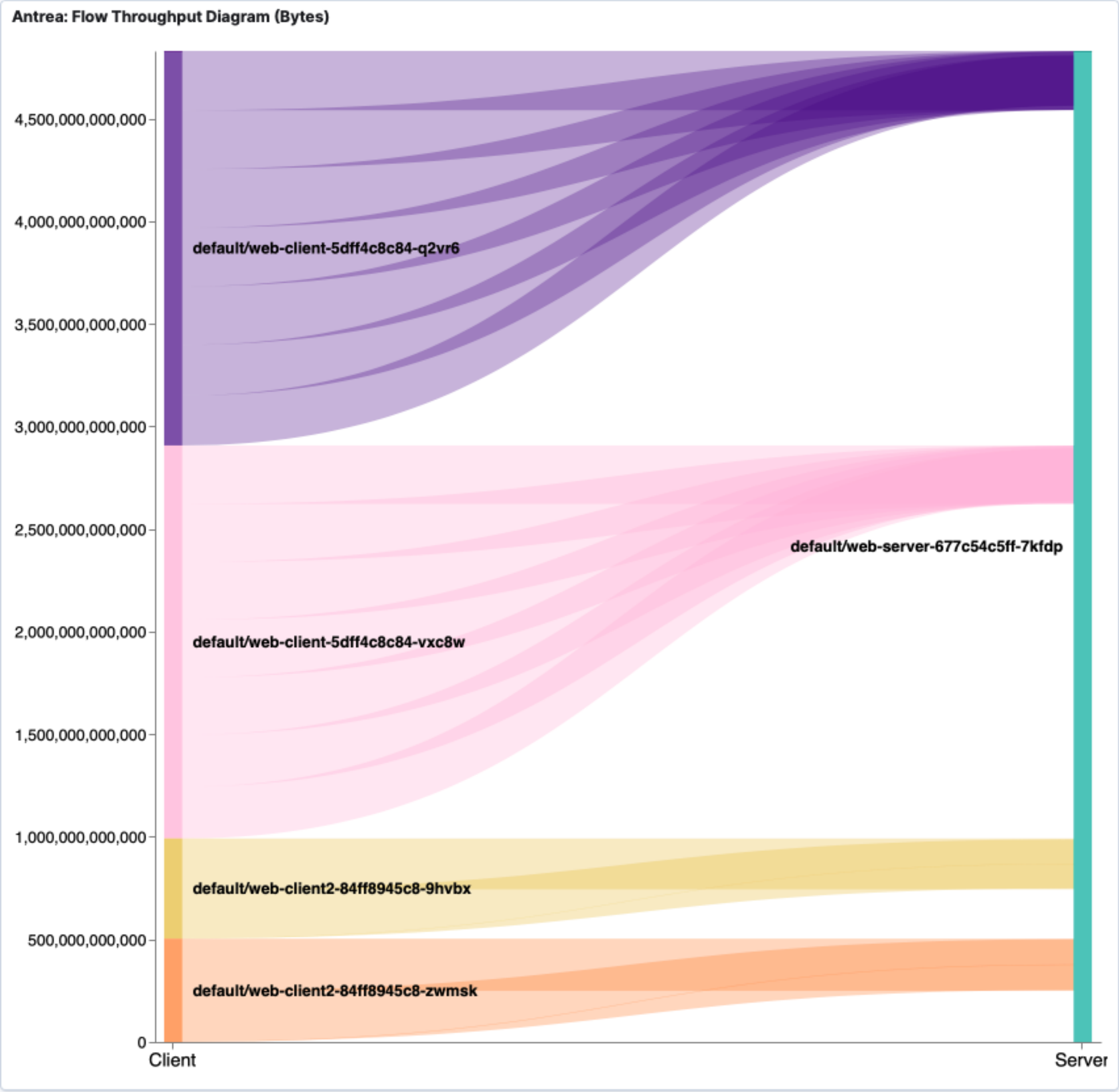
Antrea: Flow Record Table

301–350 of 607

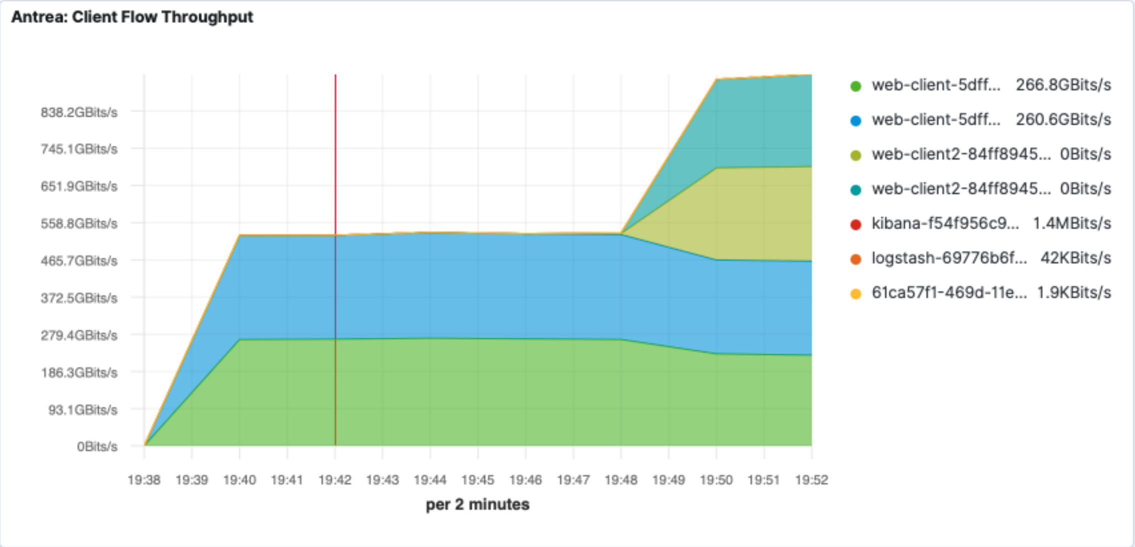
	Time	ipfix.sourceIPv4Address	ipfix.destinationIPv4Address	ipfix.sourcePodName	ipfix.destinationPodName	ipfix.bytes	ipfix.packets
›	Jun 28, 2020 @ 19:50:51.000	10.0.1.144	10.0.1.146	kibana-f54f956c9-w8682	elasticsearch-0	0	0
›	Jun 28, 2020 @ 19:50:51.000	10.0.1.98	10.0.1.100	web-client-5dff4c8c84-vxc8w	web-server-677c54c5ff-7kfdp	252,586,053,646	8,962,583
›	Jun 28, 2020 @ 19:50:51.000	10.0.1.144	10.0.1.146	kibana-f54f956c9-w8682	elasticsearch-0	0	0
›	Jun 28, 2020 @ 19:50:51.000	10.0.1.144	10.0.1.146	kibana-f54f956c9-w8682	elasticsearch-0	0	0

Flow information visualization

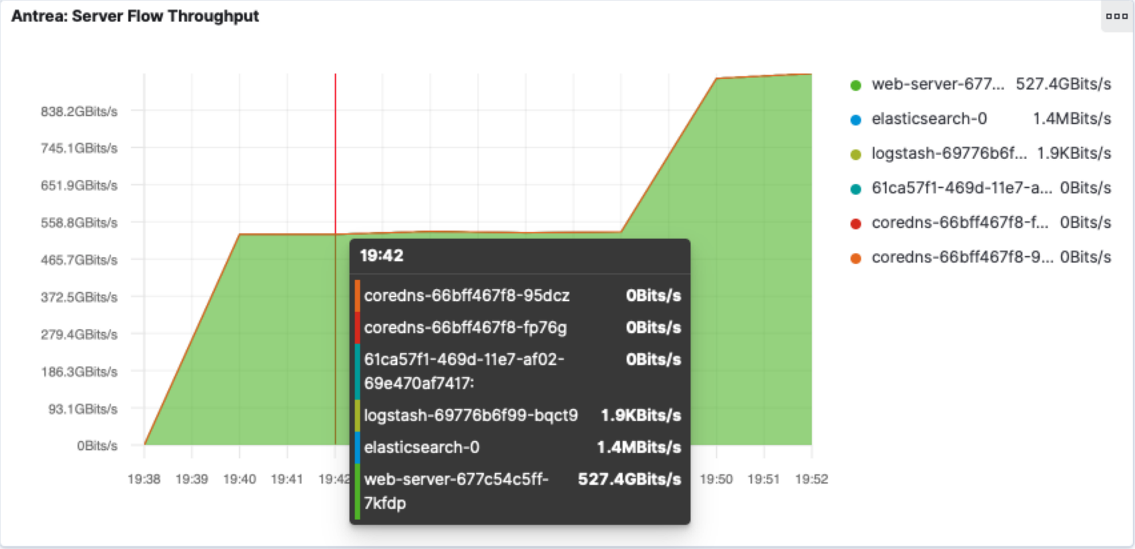
With Elastic Stack



Antrea: Flow Throughput Diagram (Bytes)



Antrea: Client Flow Throughput



Antrea: Server Flow Throughput

19:42	
coredns-66bff467f8-95dcz	0Bits/s
coredns-66bff467f8-fp76g	0Bits/s
61ca57f1-469d-11e7-af02-69e470af7417:	0Bits/s
logstash-69776b6f99-bqct9	1.9KBits/s
elasticsearch-0	1.4MBits/s
web-server-677c54c5ff-7kfdp	527.4GBits/s

Get Involved

ANTREA

Come help us continually improve Kubernetes Networking!

@ projectantrea-announce

projectantrea

projectantrea-dev

(Google Groups)

@ProjectAntrea

Kubernetes Slack
#antrea

Community Meeting, Mondays @ 9PM PT
Zoom ID: 823-654-111

https://github.com/vmware-tanzu/antrea

- Good first issues
- Help us improve our documentation
- Propose new features
- File Bugs

https://antrea.io

- Documentation
- Blogs

Thank You

Backup Slides

Antrea in Public Cloud

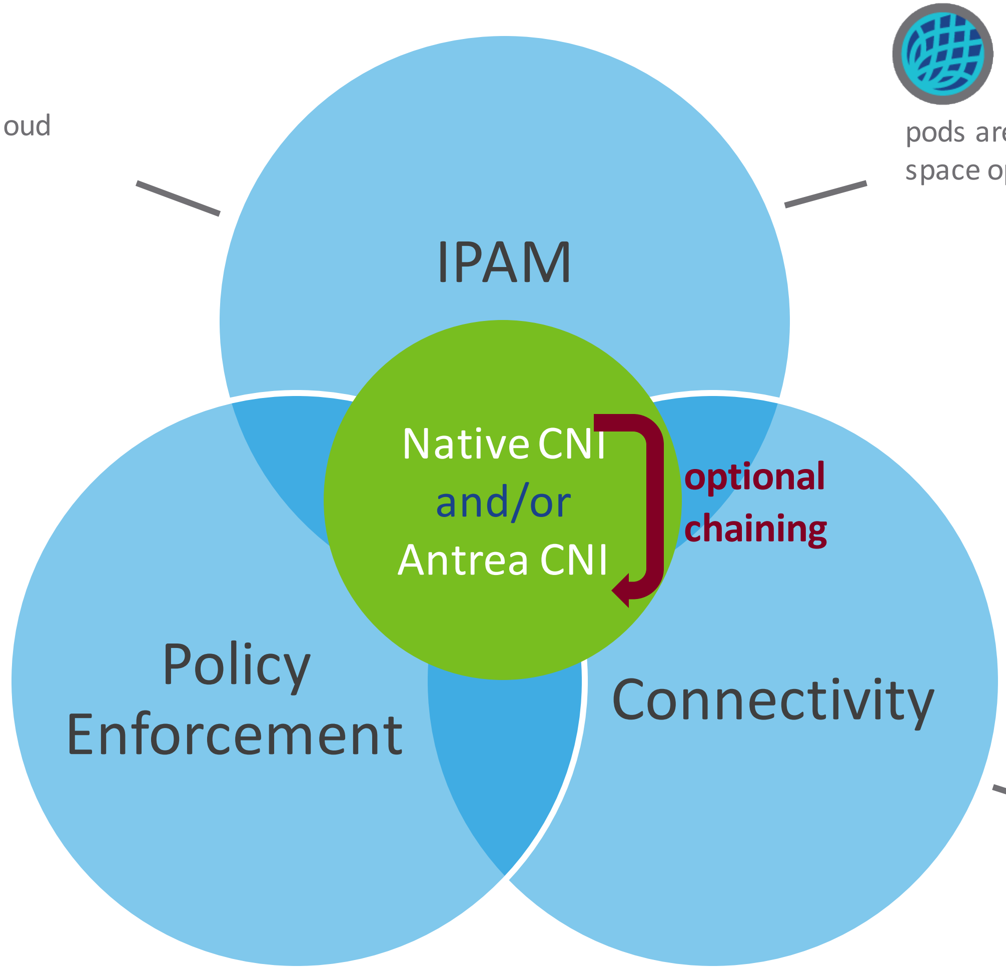
The Antrea CNI provides both pod connectivity and network policy enforcement and is flexible to use in either cloud native or overlay IP addressing schemes.

● Native CNI

IP addresses are assigned from cloud native private network (VPC)

ANTREA

pods are assigned addresses from IP space opaque to the cloud



ANTREA

Enforces network policy and filters traffic to/from pods

ANTREA

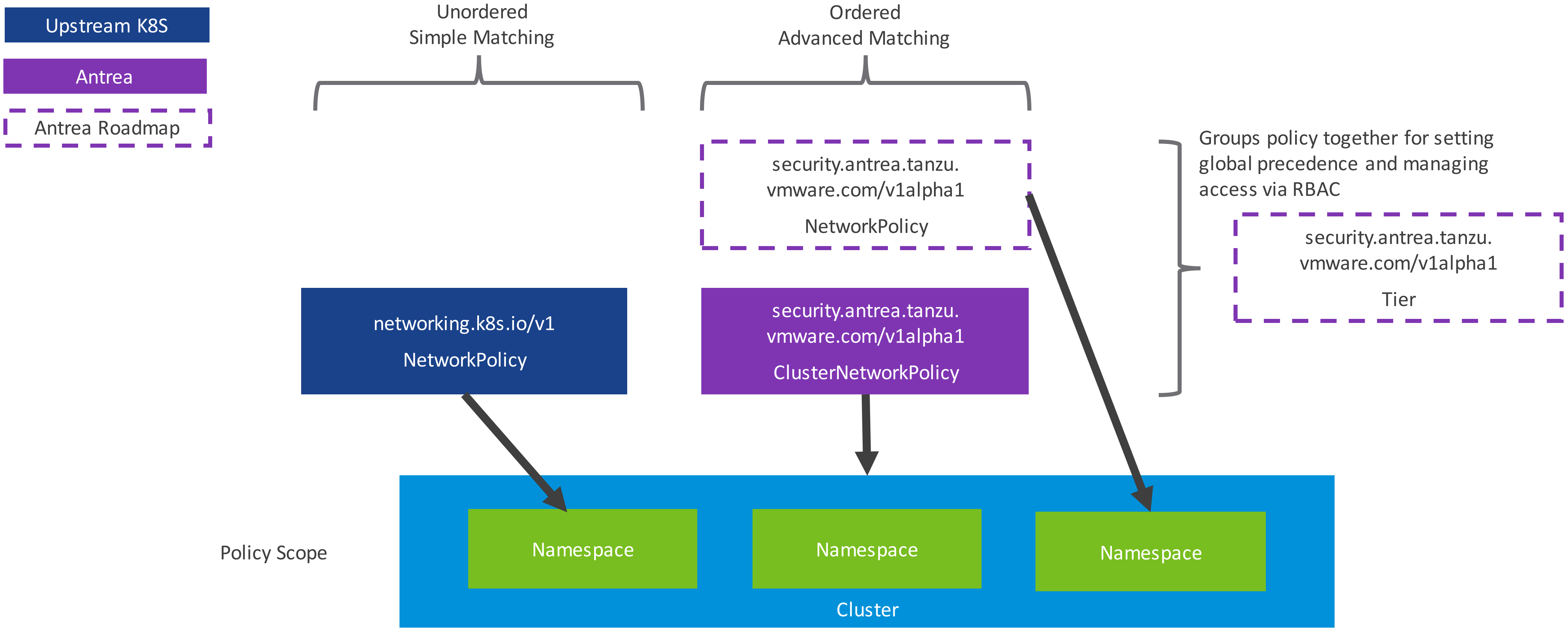
can provide overlay encapsulation and encryption when connecting pods

● Native CNI

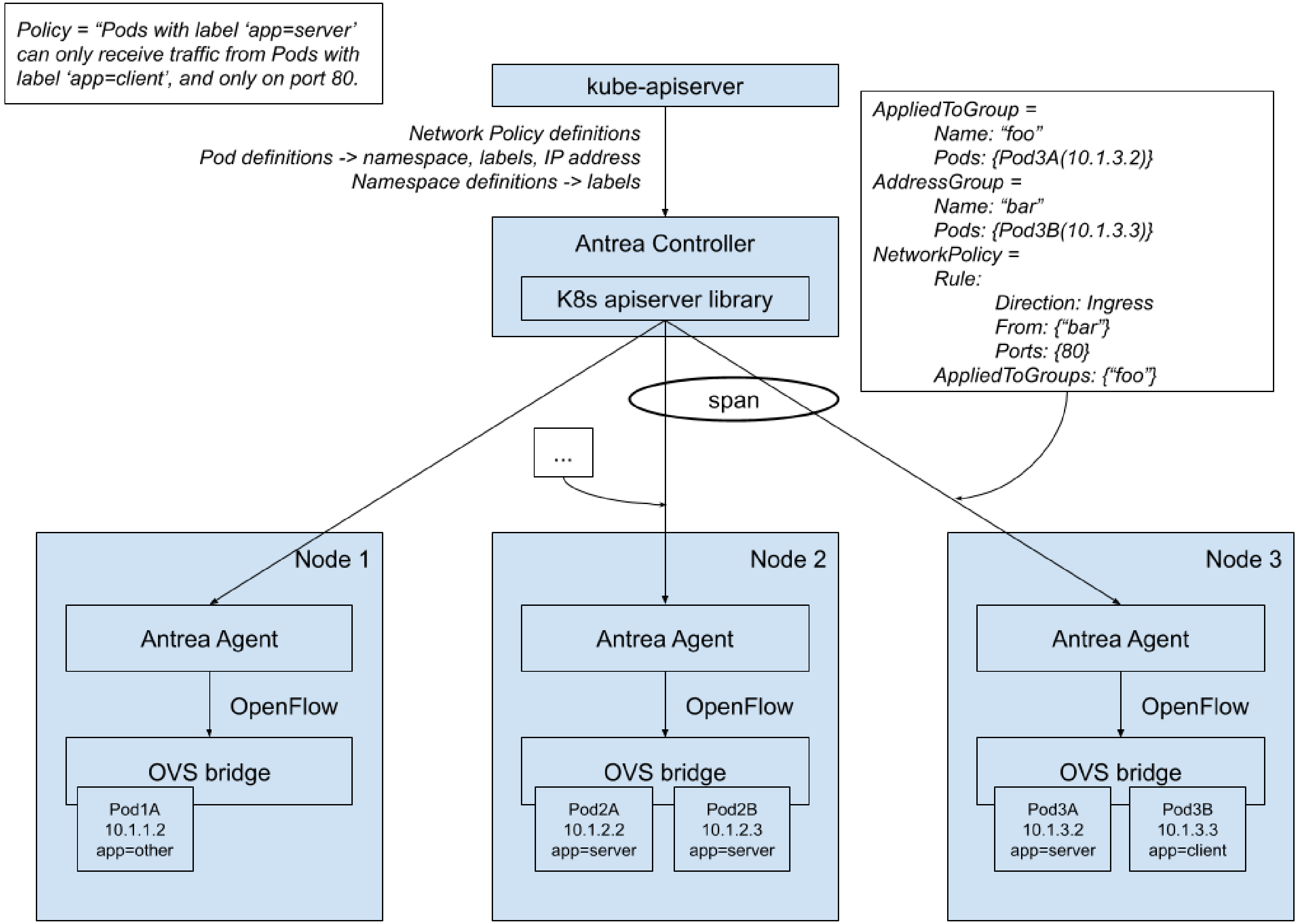
pods are routable on cloud fabric

Network Policy Resources

Antrea supports both upstream K8S and native policy primitives



NetworkPolicy Implementation



Policy = "Pods with label 'app=server' can only receive traffic from Pods with label 'app=client', and only on port 80.
kube-apiserver
Network Policy definitions
Pod definitions -> namespace, labels, IP address
Namespace definitions -> labels
Antrea Controller
K8s apiserver library
span
...
AppliedToGroup =
Name: "foo"
Pods: {Pod3A(10.1.3.2)}
AddressGroup =
Name: "bar"
Pods: {Pod3B(10.1.3.3)}
NetworkPolicy =
Rule:
Direction: Ingress
From: {"bar"}
Ports: {80}
AppliedToGroups: {"foo"}
Node 1
Antrea Agent
OpenFlow
OVS bridge
Pod1A 10.1.1.2 app=other
Node 2
Antrea Agent
OpenFlow
OVS bridge
Pod2A 10.1.2.2 app=server
Pod2B 10.1.2.3 app=server
Node 3
Antrea Agent
OpenFlow
OVS bridge
Pod3A 10.1.3.2 app=server
Pod3B 10.1.3.3 app=client

Centralized controller for Network Policy computation

Each Node's Agent receives only the relevant data

Very lightweight for the Node's Agent (simple conversion to flows)

Controller = single source of truth

- Easier to debug

Multiple controllers possible

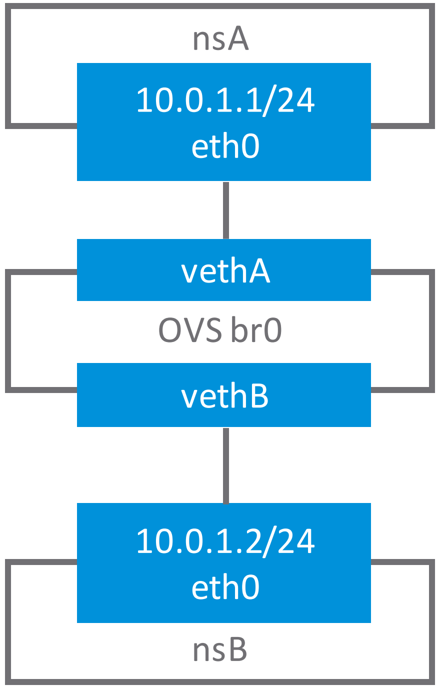
- HA
- Controller scale-out

Use OVS flow conjunction

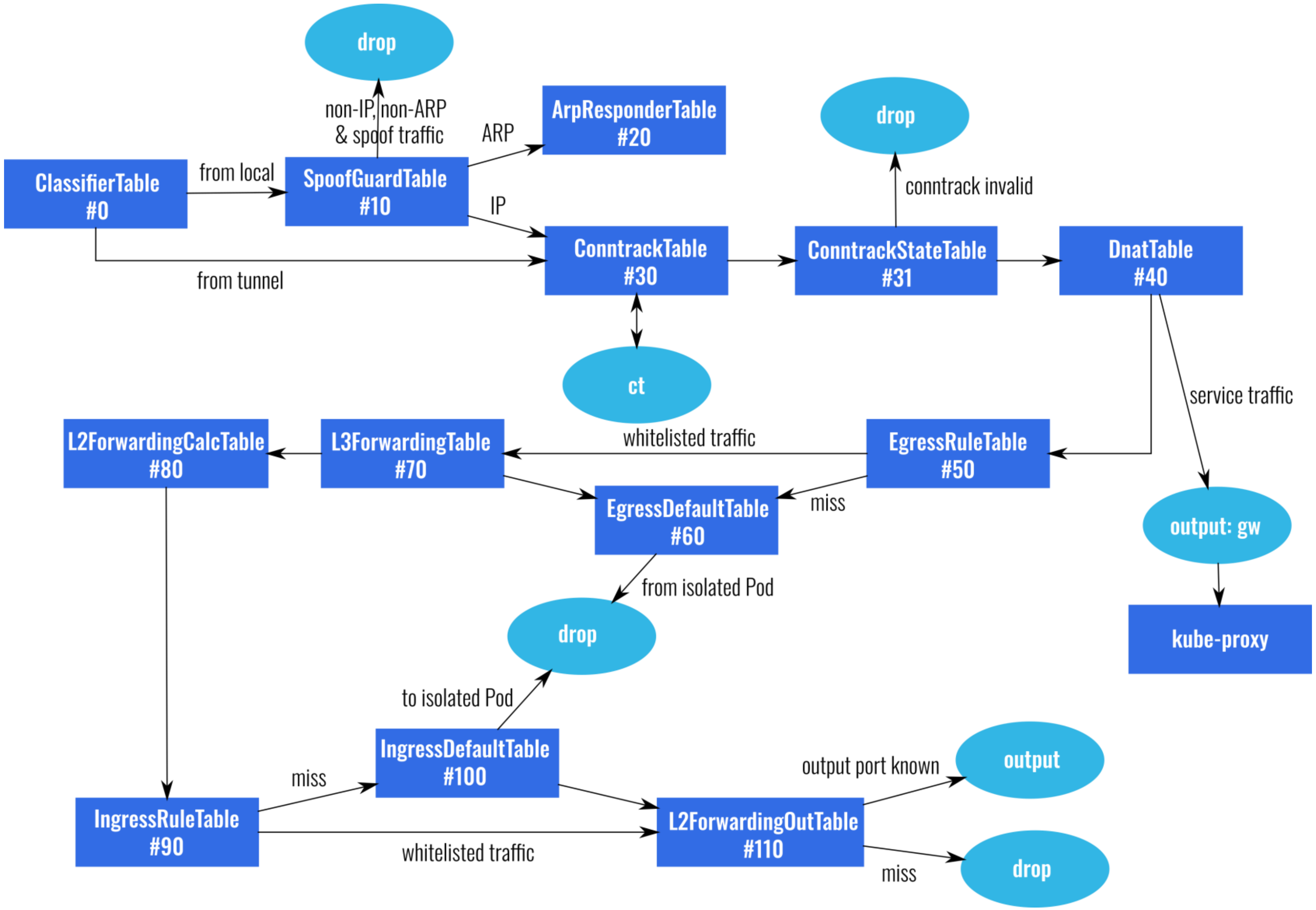
- Reduces number of flows

OVS Hello World

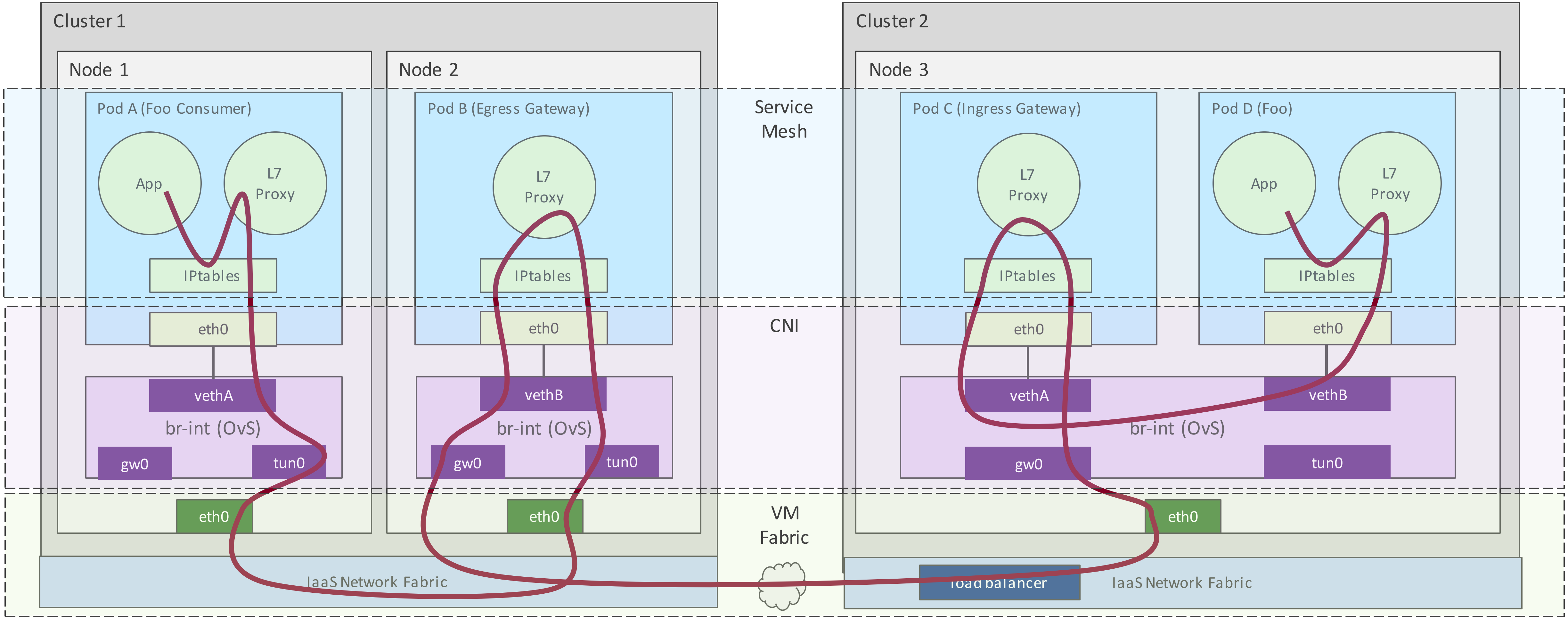
```
> ovs-vsctl add-br br0
> ovs-vsctl add-port br0 vethA
> ovs-vsctl add-port br1 vethB
> ip netns exec nsA ping -c 1 -W 1 10.0.1.2 && echo “SUCCESS”
SUCCESS
>
> ovs-ofctl add-flow br0 priority=100,icmp,actions=drop
> ip netns exec nsA ping -c 1 -W 1 10.0.1.2 || echo “FAILED”
FAILED
>
> ovs-ofctl dump-flows br0
table=0, n_packets=1, n_bytes=98, priority=100,icmp actions=drop
table=0, n_packets=18, n_bytes=1092, priority=0 actions=NORMAL
```



OVS Pipeline



Antrea Packet Walk Across Network Layers


Cluster 1
Node 1
Node 2
Pod A (Foo Consumer)
App
L7 Proxy
IPtables
eth0
vethA
br-int (OvS)
gw0
tun0
eth0
Pod B (Egress Gateway)
L7 Proxy
IPtables
eth0
vethB
br-int (OvS)
gw0
tun0
eth0
IaaS Network Fabric
Service Mesh
CNI
VM Fabric
Cluster 2
Node 3
Pod C (Ingress Gateway)
L7 Proxy
IPtables
eth0
vethA
gw0
Pod D (Foo)
App
L7 Proxy
IPtables
eth0
vethB
tun0
br-int (OvS)
eth0
load balancer
IaaS Network Fabric