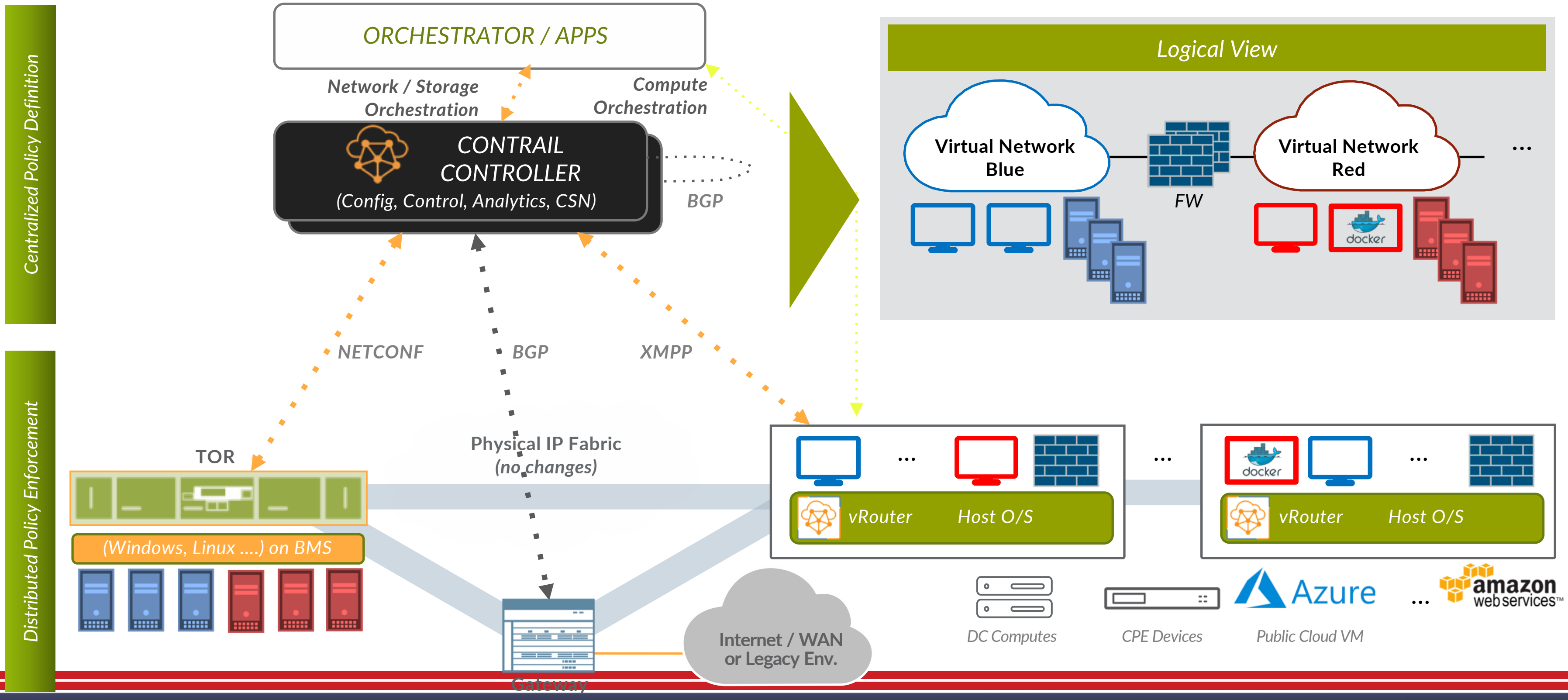
TungstenFabric + Kubernetes

Pragash Vijayaragavan
Yuvaraja Mariappan
Sachchidanand Vaidya

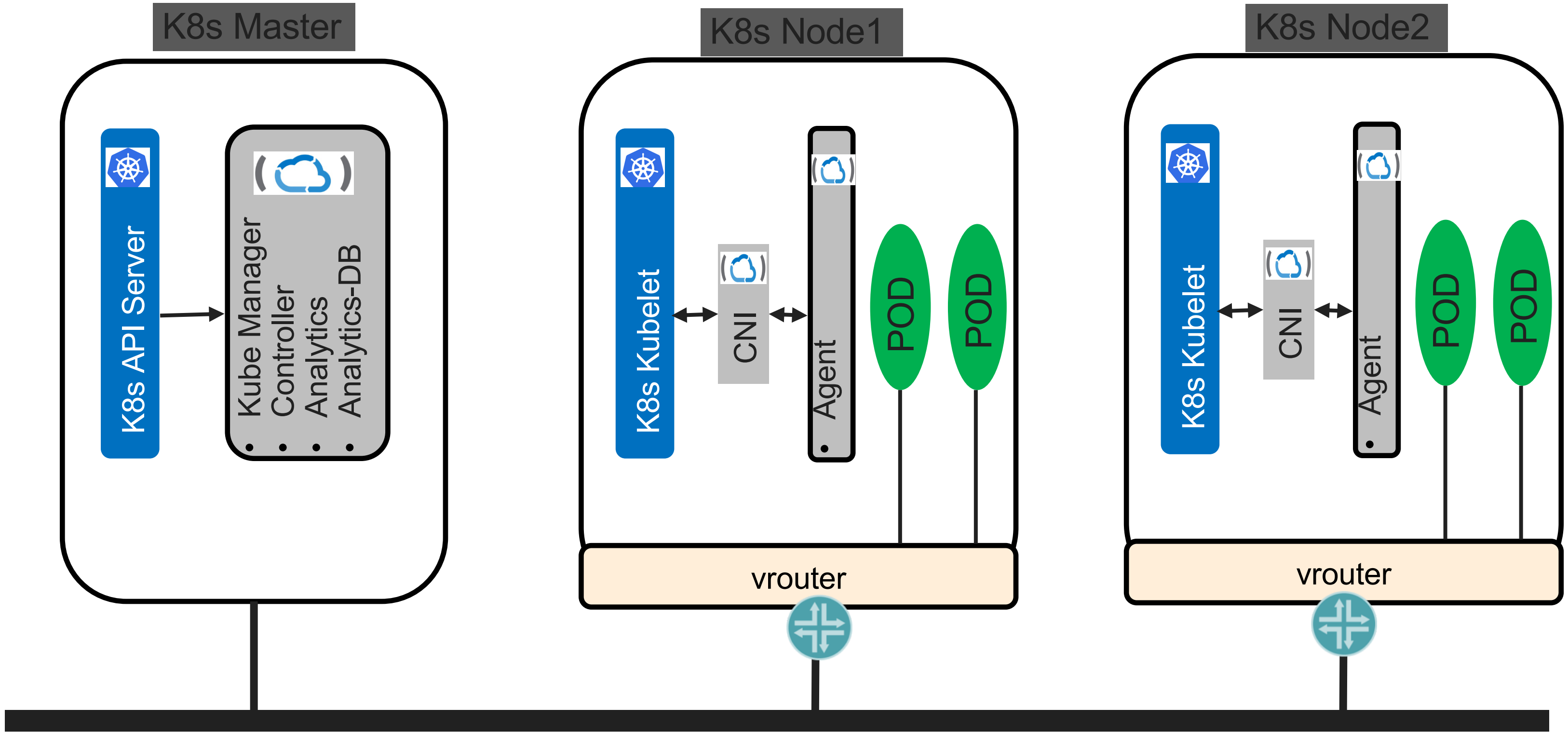
Agenda

- Tungsten Fabric – Kubernetes Integration – Architecture View
- Tungsten Network Policy
- Tungsten Service Chaining in kubernetes

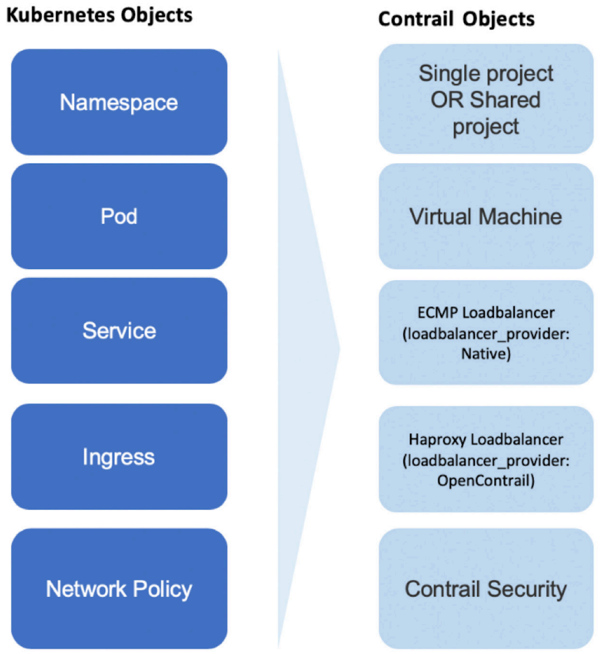
Architecture Overview



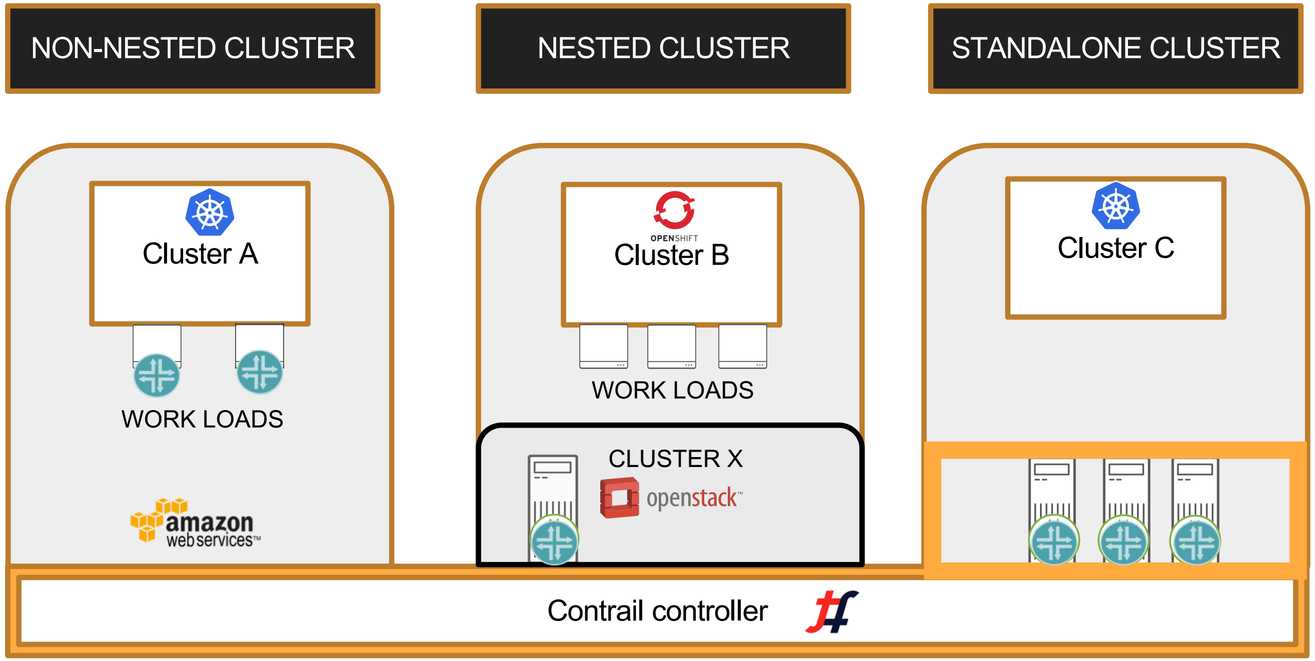
Tungsten Fabric with Kubernetes on baremetal



Kubernetes to Tungsten Object Mapping

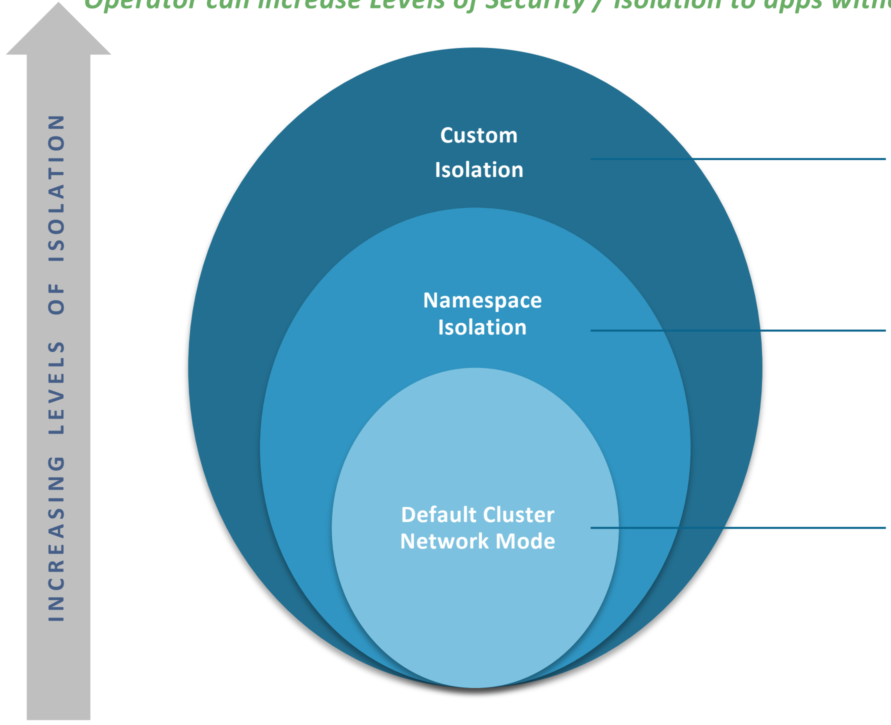

Kubernetes Objects
Contrail Objects
Namespace
Single project OR Shared project
Pod
Virtual Machine
Service
ECMP Loadbalancer (loadbalancer_provider: Native)
Ingress
Haproxy Loadbalancer (loadbalancer_provider: OpenContrail)
Network Policy
Contrail Security

Tungsten Fabric Multi Cluster



Tungsten Fabric with Kubernetes

Operator can increase Levels of Security / Isolation to apps without changing App developer / deployer workflow

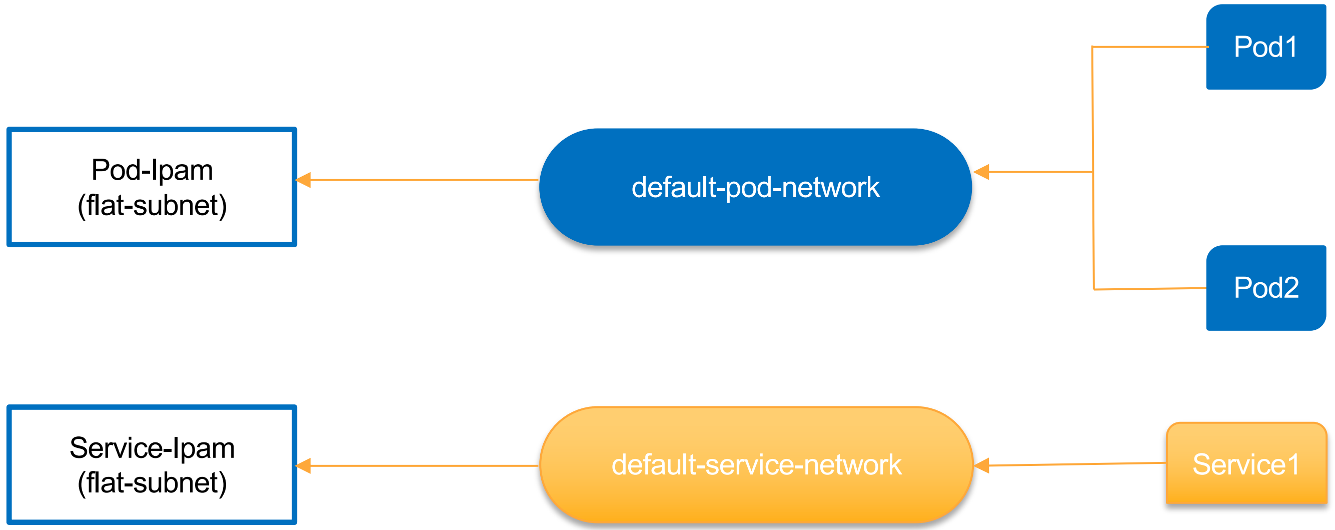


- Define Networks and Security Policies in Contrail
- Use Network/Security Policy per namespace and/or pods using annotations

- Virtual Network per namespace
- Isolation per namespace

- All Pods and Services are reachable

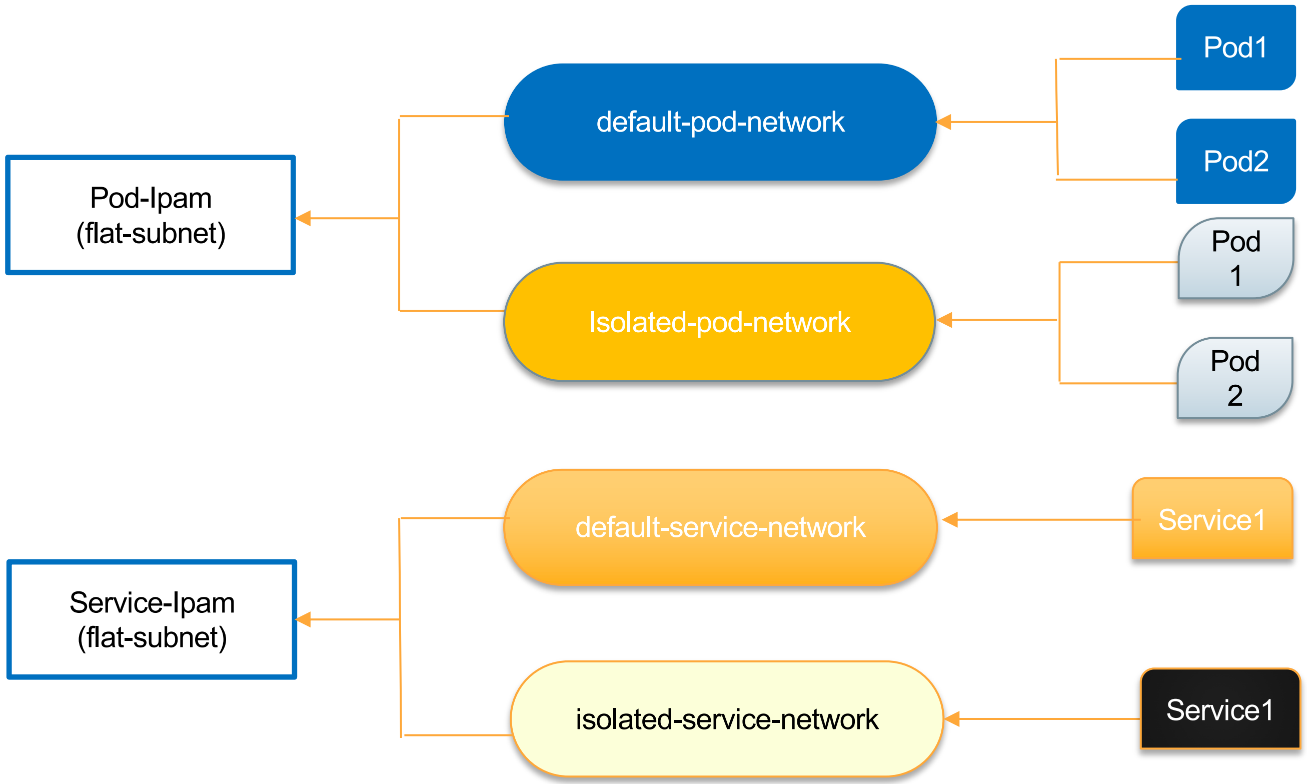
Default Virtual-Networks and Default IPAMs



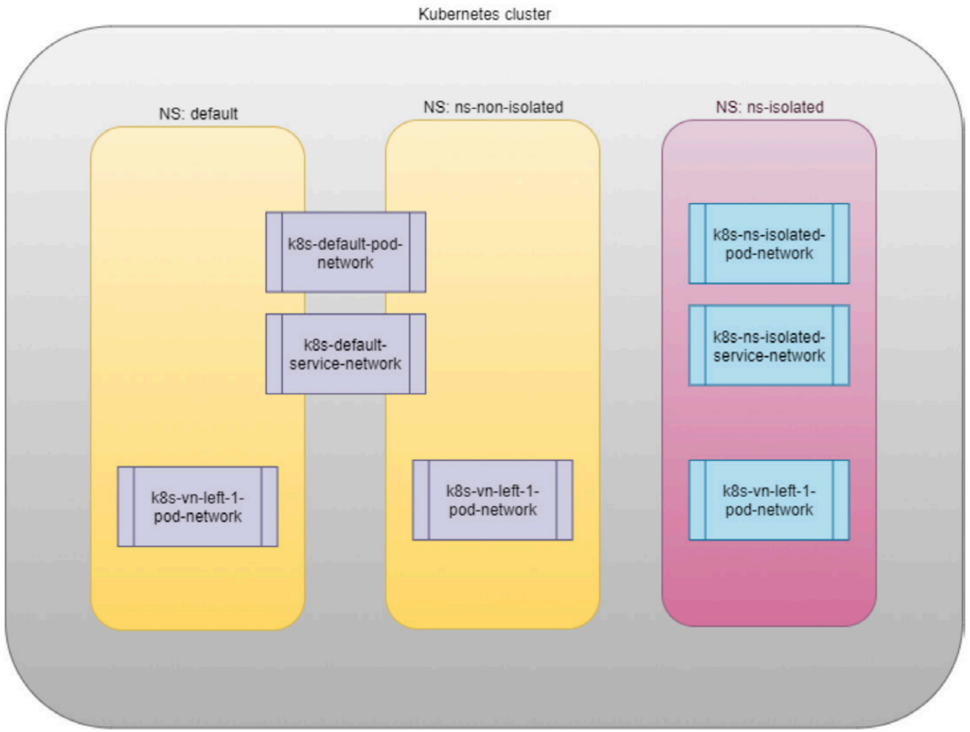
Namespaces, Projects and Virtual-Networks

- Each Namespace maps to a Project.
- Non-isolated Namespaces links to default-pod-network and default-service-network.
- Each Isolated Namespace has its own pod-network and service-network with default-ipams.

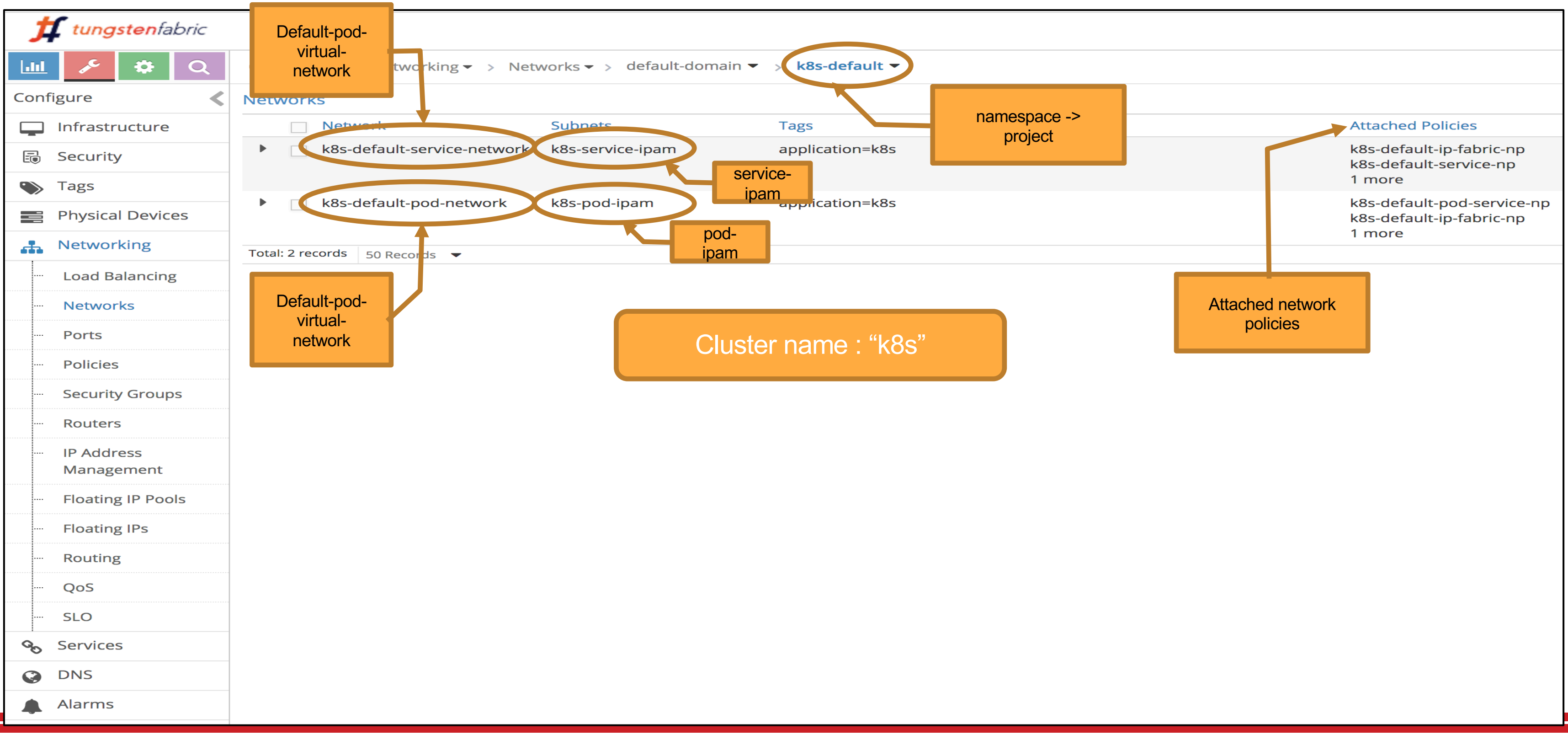
Default and Isolated Virtual-Networks



Default and Isolated Virtual-Networks


Kubernetes cluster
NS: default
NS: ns-non-isolated
NS: ns-isolated
k8s-default-pod-network
k8s-default-service-network
k8s-ns-isolated-pod-network
k8s-ns-isolated-service-network
k8s-vn-left-1-pod-network
k8s-vn-left-1-pod-network
k8s-vn-left-1-pod-network

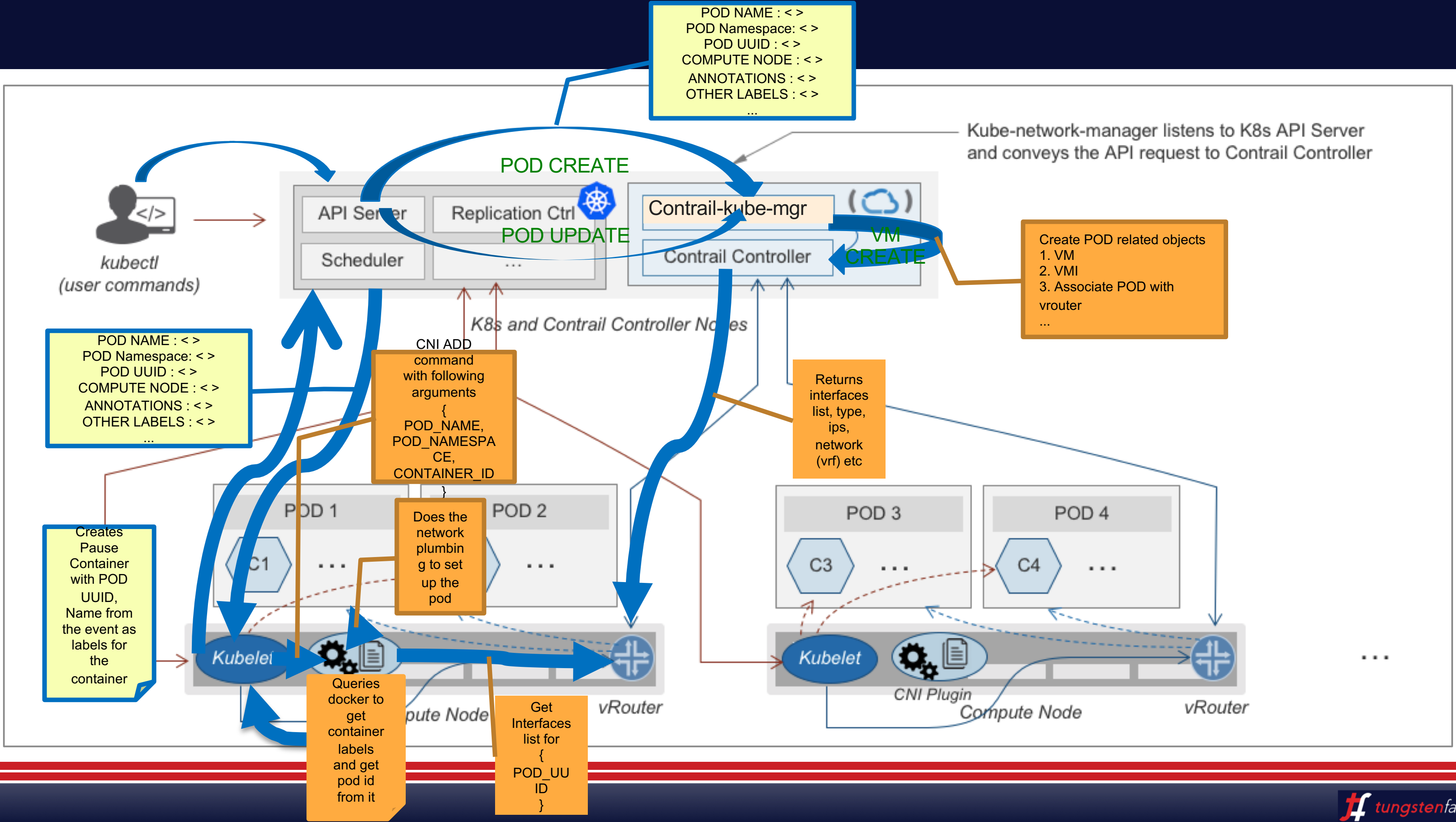
Tungsten Fabric UI



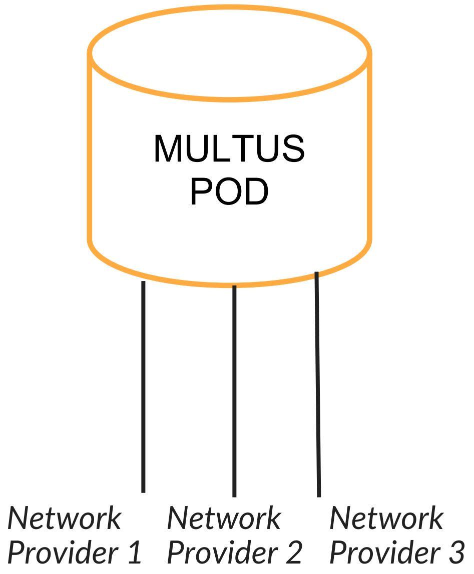
Network	Subnets	Tags	Attached Policies
k8s-default-service-network	k8s-service-ipam	application=k8s	k8s-default-ip-fabric-np k8s-default-service-np 1 more
k8s-default-pod-network	k8s-pod-ipam	application=k8s	k8s-default-pod-service-np k8s-default-ip-fabric-np 1 more

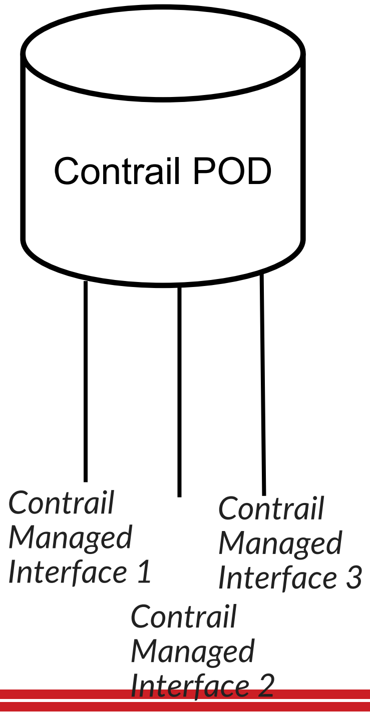
Total: 2 records 50 Records

Pod Creation Workflow


POD NAME : < >
POD Namespace: < >
POD UUID : < >
COMPUTE NODE : < >
ANNOTATIONS : < >
OTHER LABELS : < >
...
Kube-network-manager listens to K8s API Server and conveys the API request to Contrail Controller
POD CREATE
POD UPDATE
API Server
Replication Ctrl
Scheduler
...
Contrail-kube-mgr
Contrail Controller
VM CREATE
kubectl (user commands)
Create POD related objects
1. VM
2. VMI
3. Associate POD with vrouter
...
K8s and Contrail Controller Nodes
POD NAME : < >
POD Namespace: < >
POD UUID : < >
COMPUTE NODE : < >
ANNOTATIONS : < >
OTHER LABELS : < >
...
CNI ADD command with following arguments { POD_NAME, POD_NAMESPACE, CONTAINER_ID }
Returns interfaces list, type, ips, network (vrf) etc
POD 1
POD 2
POD 3
POD 4
C1
C3
C4
Creates Pause Container with POD UUID, Name from the event as labels for the container
Does the network plumbing to set up the pod
Kubelet
Queries docker to get container labels and get pod id from it
Compute Node
Get Interfaces list for { POD_UUID }
vRouter
CNI Plugin
...

MultI Interface for Pod




Contrail POD
Contrail Managed Interface 1
Contrail Managed Interface 3
Contrail Managed Interface 2

Network Attachment Definition - CRD

```
apiVersion: "k8s.cni.cncf.io/v1"
kind: NetworkAttachmentDefinition
metadata:
              name: <network-name>
              namespace: <namespace-name>
              annotations:
                            "opencontrail.org/cidr" : [<ip-subnet>]
                            "opencontrail.org/ip_fabric_snat" : <True/False>
                            "opencontrail.org/ip_fabric_forwarding" : <True/False>
spec:
              config: '{
                            "cniVersion": "0.3.0",
                            "type": "contrail-k8s-cni"
}'
```

name:	name of the network
namespace:	namespace that the network object belongs to.
opencontrail.org/cidr:	CIDR of the network
opencontrail.org/ip_fabric_snat:	Enable/Disable fabric SNAT
opencontrail.org/ip_fabric_forwarding:	Flag for ip fabric forwarding

Multi Network Pod Spec

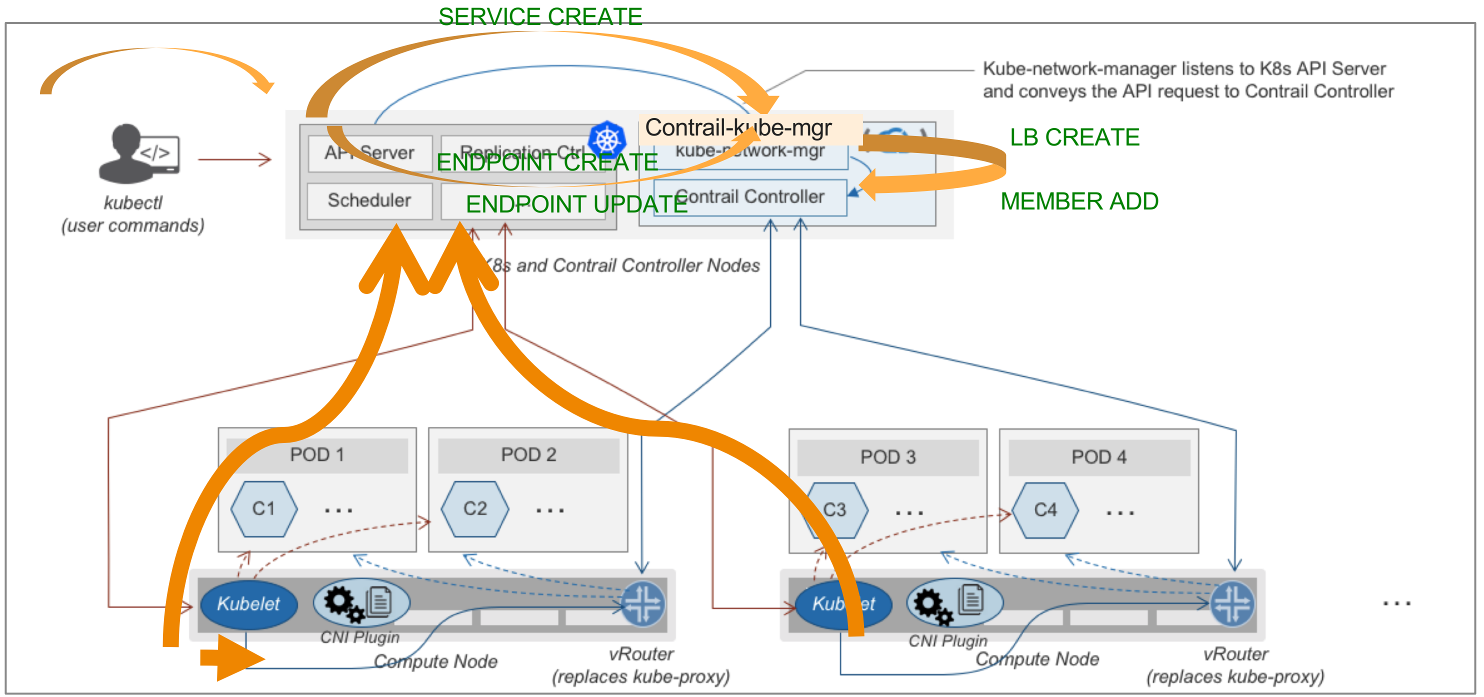
```
kind: Pod
metadata:
        name: my-pod
        namespace: my-namespace
        annotations:
                k8s.v1.cni.cncf.io/networks: ‘[
                        { "name": "net-a" },
                        { "name": "net-b" },
                        { "name": "net-c", "namespace": "other-ns” }
                ]’

spec:
        containers:
```

CNI and Kubemanager code

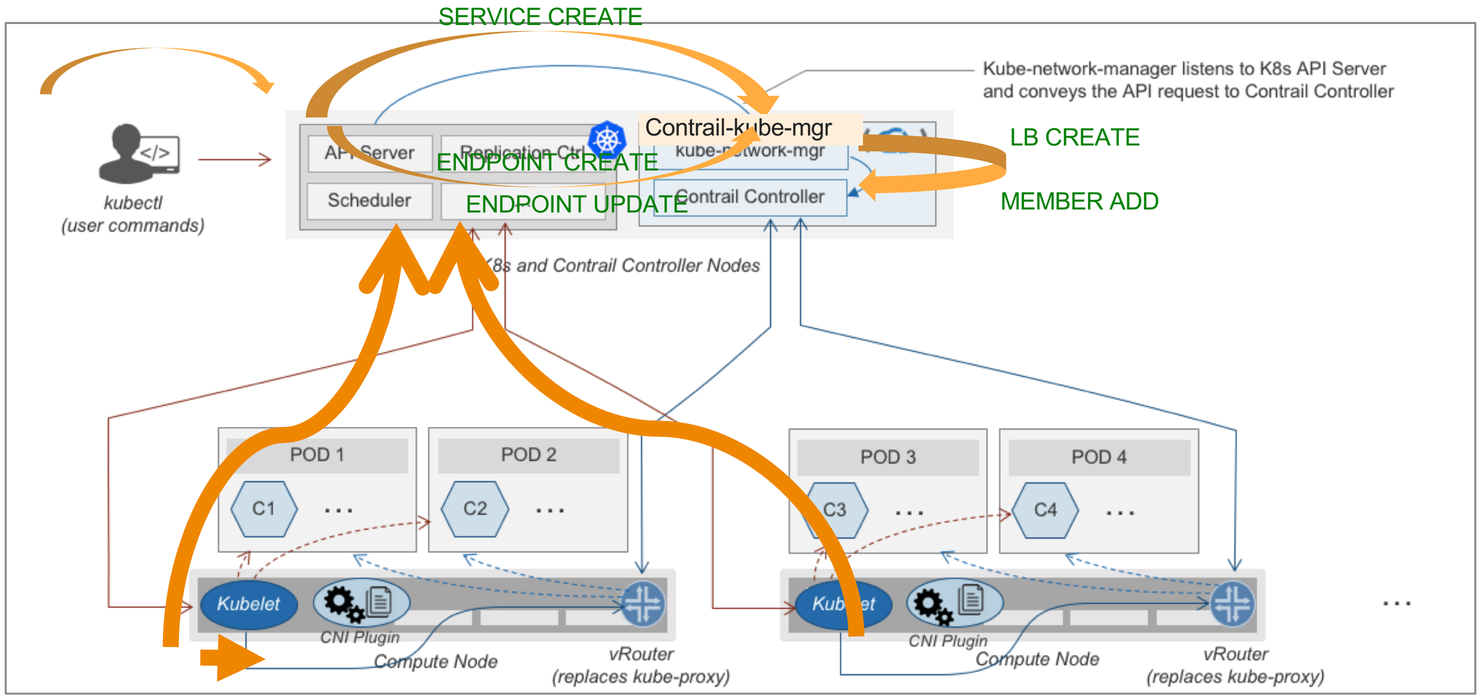
- https://github.com/Juniper/contrail-controller/blob/master/src/container/kube-manager/kube_manager/kube_manager.py → Kubemanager

- https://github.com/Juniper/contrail-controller/blob/master/src/container/cni/cni/kube_cni/contrail-kube-cni.go → CNI

Service Creation



Service Creation



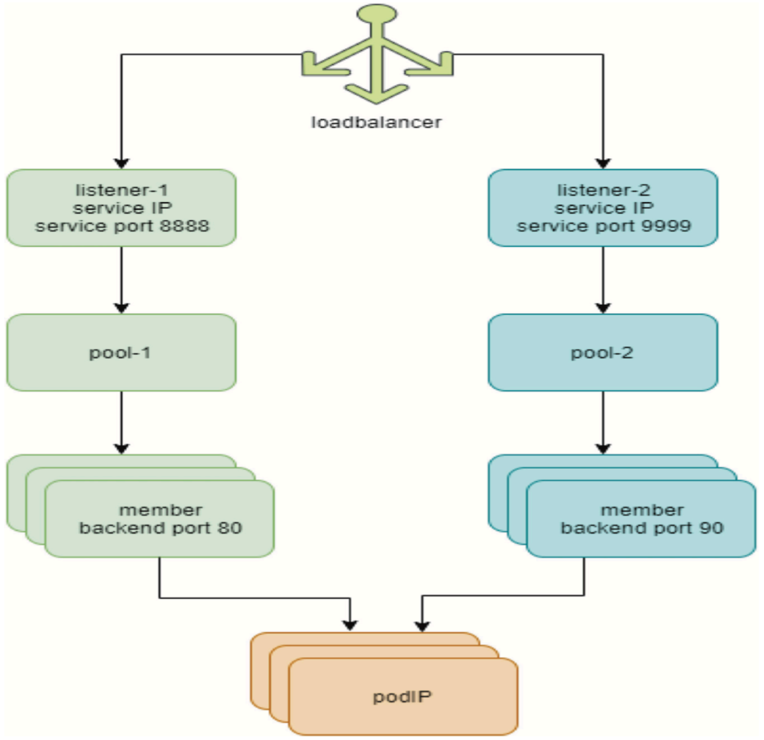
service-web-clusterip-mp yaml

service-web-clusterip-mp.yaml

```
apiVersion: v1
kind: Service
metadata:
  name: service-web-clusterip-mp
spec:
  selector:
    app: webserver
  ports:
    - name: port1
      port: 8888
      targetPort: 80
    - name: port2
      port: 9999
      targetPort: 90
```

service - loadbalancer objects

service-loadbalancer objects



Service Load Balancer

Kubernetes Network POLICY

tungstenfabric

Kubernetes Network Policy

A Network Policy is a specification of:

- how groups of Pods are allowed to communicate with each other
- how groups of Pods are allowed to communicate with other network endpoints.

Problem

Contrail supports Kubernetes Network Policy from R4.0.

So what is the problem ? Why are we talking about it in R5.0 ?

R4.0
Kubernetes Network Policy is implemented as Contrail Security Groups

R5.0:
Kubernetes Network Policy is implemented as Contrail FW Security Policy

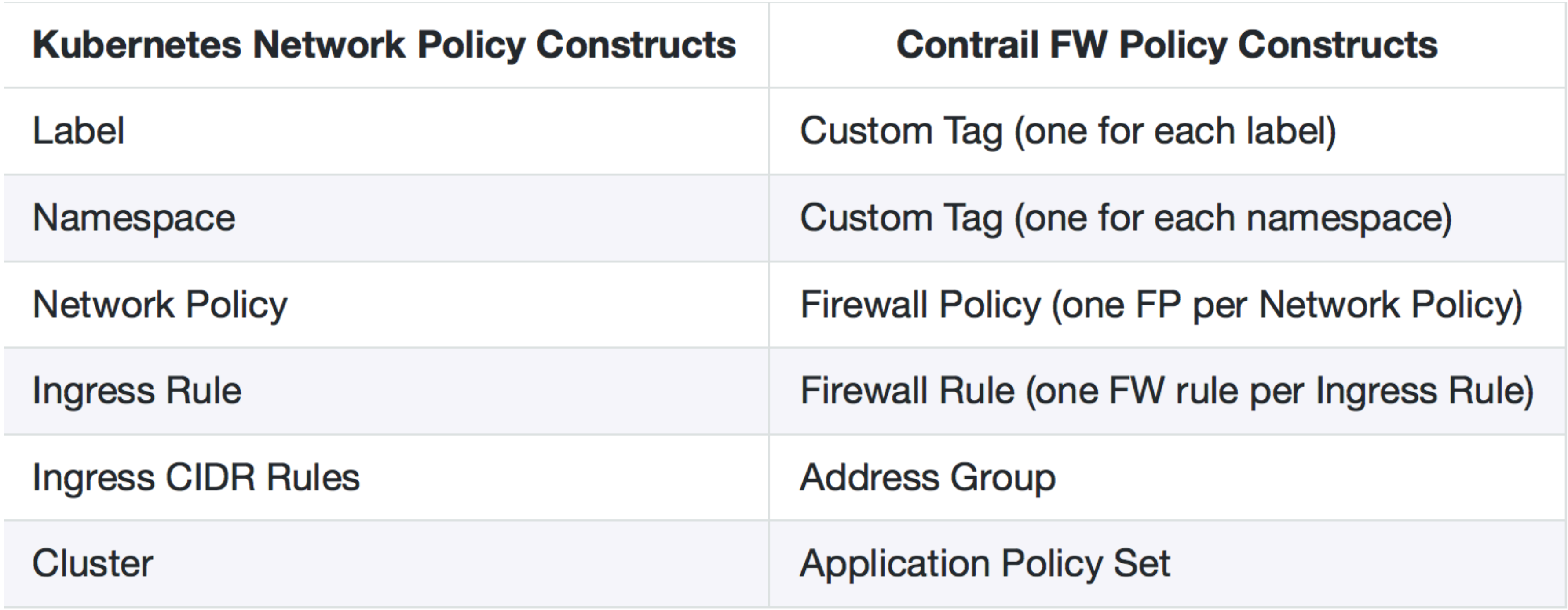
Why Contrail fw Security Policy

- Implements framework to enforce access specification across workloads ✓
- Workloads are represented and grouped by Contrail Tags ✓
- Ability to specify policy based on combinatorial tags ✓
- Ability to target untagged workloads ✓
- Ability to enforce default behavior ✓
- Ability to apply policies at various layers in forwarding stack ✓

Kubernetes Network Policy IS Contrail Security Policy

VISUALIZATION OF FLOWS by TAGS

constructs: kubernetes vs contrail



Kubernetes Network Policy Constructs	Contrail FW Policy Constructs
Label	Custom Tag (one for each label)
Namespace	Custom Tag (one for each namespace)
Network Policy	Firewall Policy (one FP per Network Policy)
Ingress Rule	Firewall Rule (one FW rule per Ingress Rule)
Ingress CIDR Rules	Address Group
Cluster	Application Policy Set

K8s Network policy - expectations

- By default, pods are non-isolated; they accept traffic from any source.
- Pods become isolated by having a NetworkPolicy that selects them.
- Once there is any NetworkPolicy in a namespace selecting a particular pod, that pod will reject any connections that are not allowed by any NetworkPolicy.
- Pods in the namespace that are not selected by any NetworkPolicy will continue to accept all traffic.

- A network policy may define traffic rules for a pod at the ingress, egress or both.
- Mutiple network policies can be applied on any pod.
- NetworkPolicy act on flows rather than individual packets.
- When network policy is applied to a pod, the policy should have explicit rules to specify a whitelist of permitted traffic in the ingress and egress. All traffic that does not match the whitelist rules are to be denied and dropped.

K8s Network policy spec

podSelector

The grouping of pods to which the policy applies.
An empty podSelector selects all pods in the namespace

policyTypes

Possible values: ***ingress*** and/or ***egress***
If no policyTypes are specified on a NetworkPolicy then by default Ingress will be assumed

ingress

List of whitelist ingress rules.
Identify source workloads by: ***namespaceSelector, podSelector, ipBlock***
Each rule allows traffic which matches both the ***from*** and ***ports*** sections.

egress

List of whitelist egress rules.
Identify destination workloads by: ***namespaceSelector, podSelector, ipBlock***
Each rule allows traffic which matches both the ***to*** and ***ports*** sections.

TEST POLICY

Filename: np.yml

Name

Spec

PODS, this Network Policy will be applied on

Type of Policies being created

Ingress ”from” spec

Ingress ”ports” spec

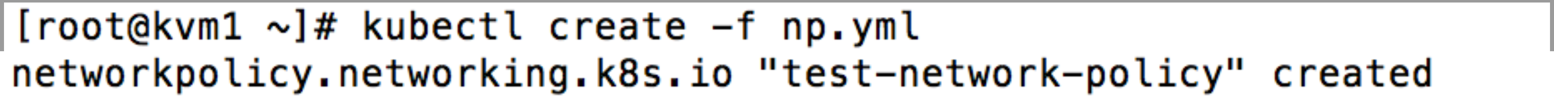
Egress ”to” spec

Egress ”ports” spec

```yaml
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: test-network-policy
  namespace: default
spec:
  podSelector:
    matchLabels:
      role: db
  policyTypes:
  - Ingress
  - Egress
  ingress:
  - from:
    - ipBlock:
        cidr: 172.17.0.0/16
        except:
        - 172.17.1.0/24
    - namespaceSelector:
        matchLabels:
          project: myproject
    - podSelector:
        matchLabels:
          role: frontend
    ports:
    - protocol: TCP
      port: 6379
  egress:
  - to:
    - ipBlock:
        cidr: 10.0.0.0/24
    ports:
    - protocol: TCP
      port: 5978
```

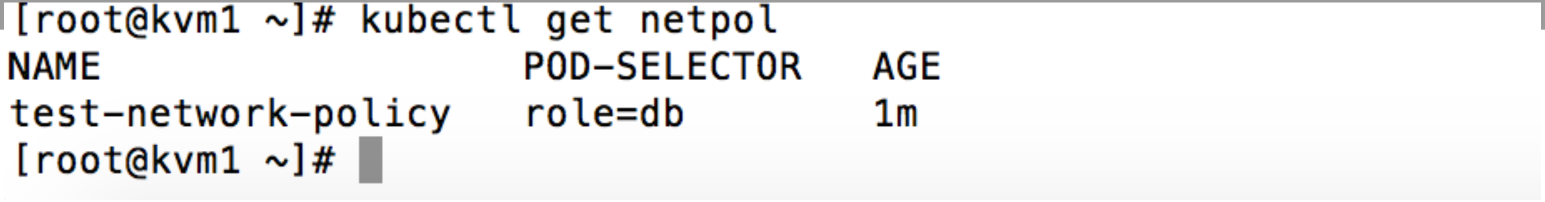
Policy in kubernetes

CREATE POLICY



```
[root@kvm1 ~]# kubectl create -f np.yml
networkpolicy.networking.k8s.io "test-network-policy" created
```

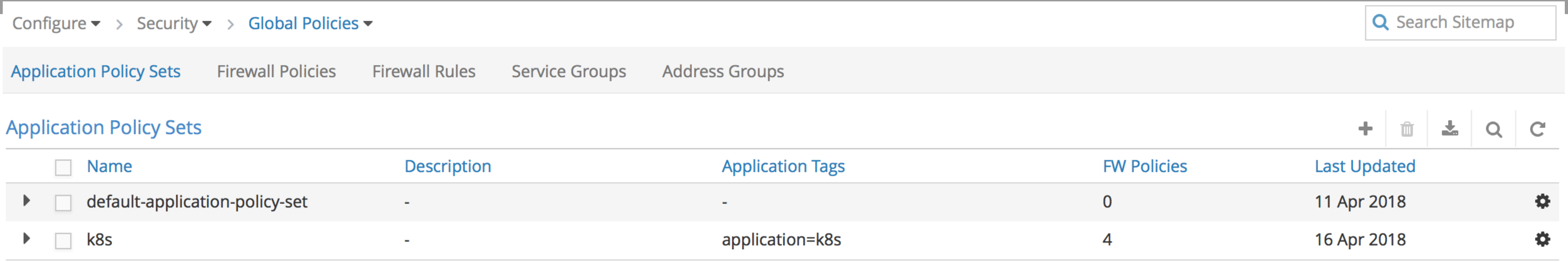
SHOW POLICY



```
[root@kvm1 ~]# kubectl get netpol
NAME                  POD-SELECTOR   AGE
test-network-policy   role=db        1m
[root@kvm1 ~]# 
```

Policy in Tungsten

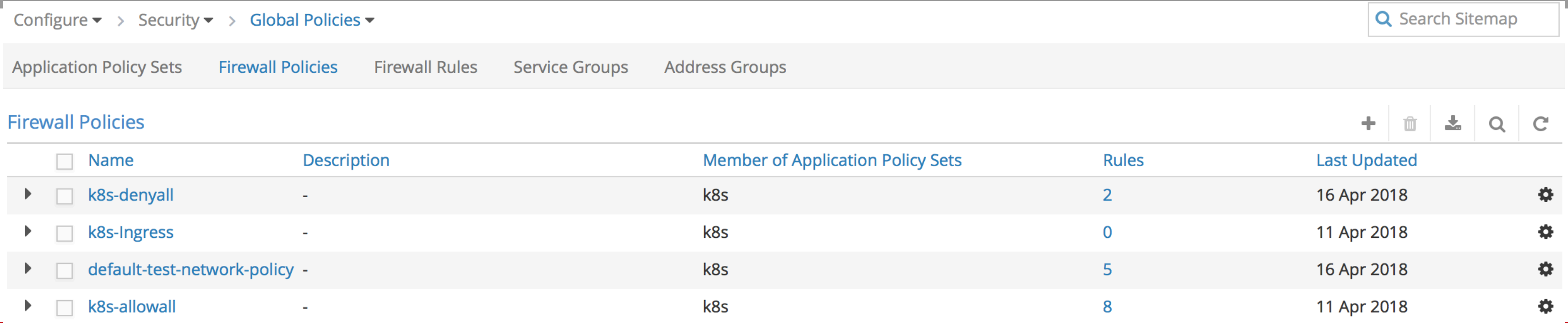
APPLICATION POLICY SET FOR THE KUBERNETES CLUSTER



Application Policy Sets

Name	Description	Application Tags	FW Policies	Last Updated
default-application-policy-set	-	-	0	11 Apr 2018
k8s	-	application=k8s	4	16 Apr 2018

SHOW POLICY

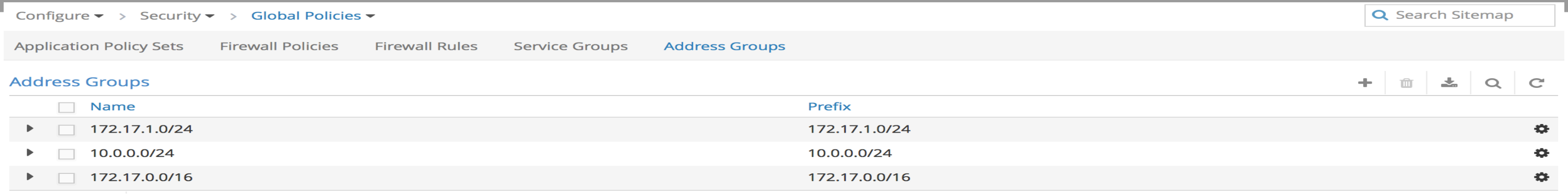


Firewall Policies

Name	Description	Member of Application Policy Sets	Rules	Last Updated
k8s-denyall	-	k8s	2	16 Apr 2018
k8s-Ingress	-	k8s	0	11 Apr 2018
default-test-network-policy	-	k8s	5	16 Apr 2018
k8s-allowall	-	k8s	8	11 Apr 2018

Policy in Tungsten … continued

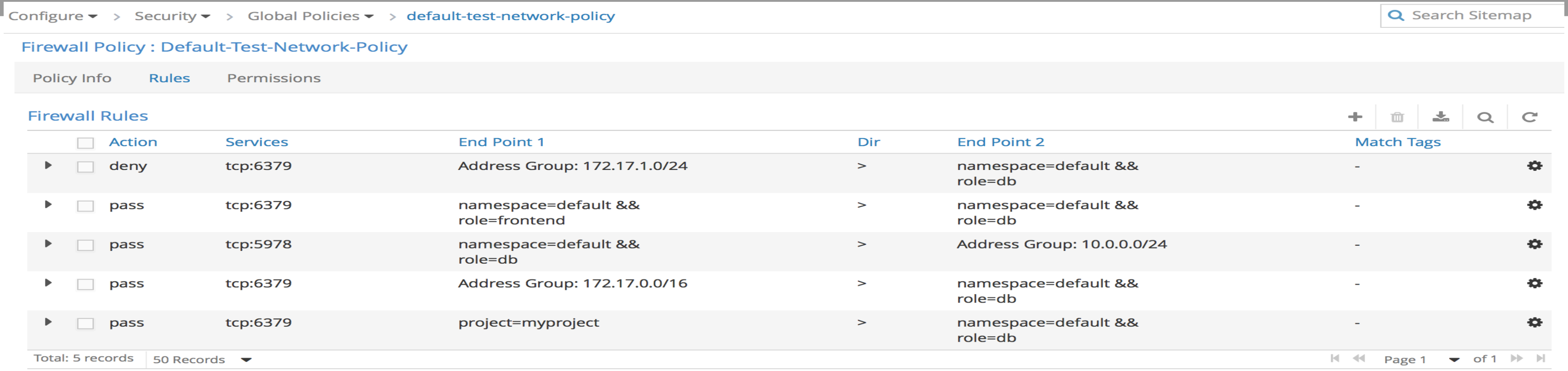
ADDRESS GROUPS CREATED FOR CIDR's



Address Groups

Name	Prefix
172.17.1.0/24	172.17.1.0/24
10.0.0.0/24	10.0.0.0/24
172.17.0.0/16	172.17.0.0/16

FIREWALL POLICY CREATED


Configure > Security > Global Policies > default-test-network-policy
Search Sitemap
Policy Info
Rules
Permissions

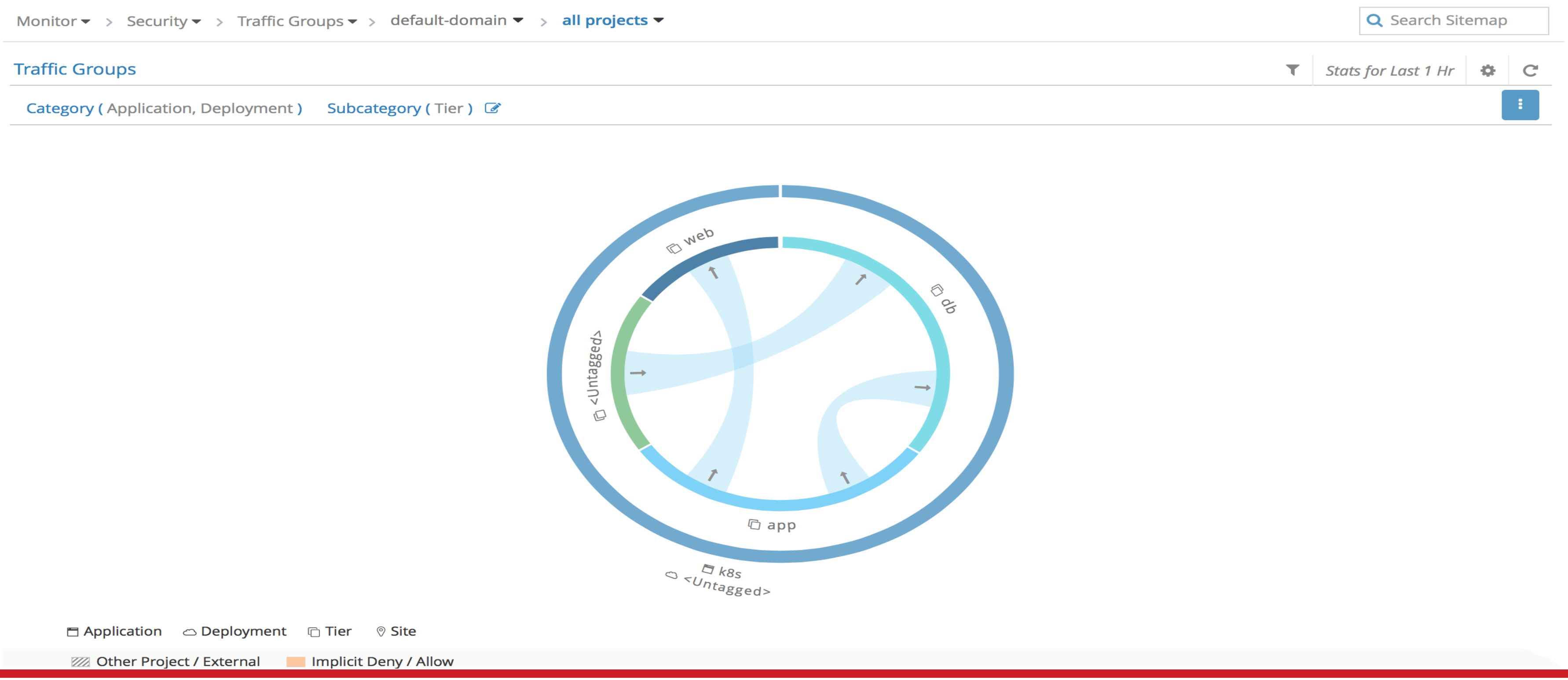
Firewall Policy : Default-Test-Network-Policy

Firewall Rules

Action	Services	End Point 1	Dir	End Point 2	Match Tags
deny	tcp:6379	Address Group: 172.17.1.0/24	>	namespace=default && role=db	-
pass	tcp:6379	namespace=default && role=frontend	>	namespace=default && role=db	-
pass	tcp:5978	namespace=default && role=db	>	Address Group: 10.0.0.0/24	-
pass	tcp:6379	Address Group: 172.17.0.0/16	>	namespace=default && role=db	-
pass	tcp:6379	project=myproject	>	namespace=default && role=db	-

Total: 5 records 50 Records Page 1 of 1

Visualization



Configuration

- ZERO TOUCH configuration.
- All required configuration created by Kube-Manager component at Bootup.

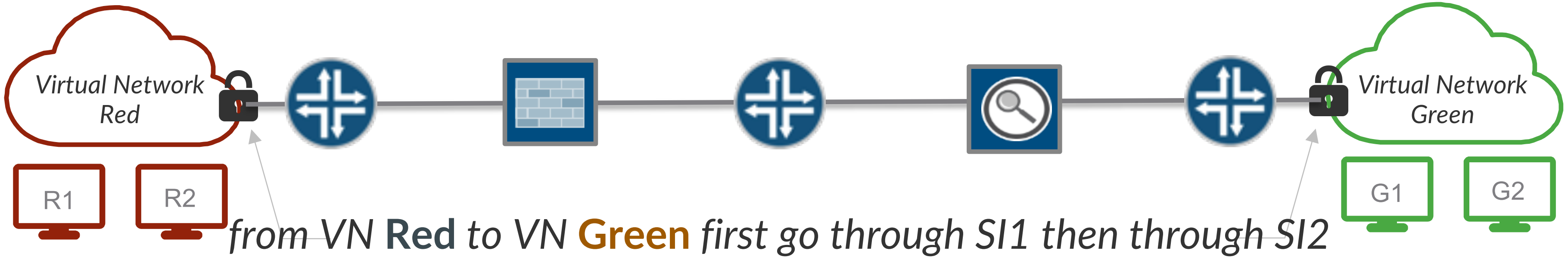
COMMANDS

- Kubernetes
  Kubectl
  Kubernetes GUI

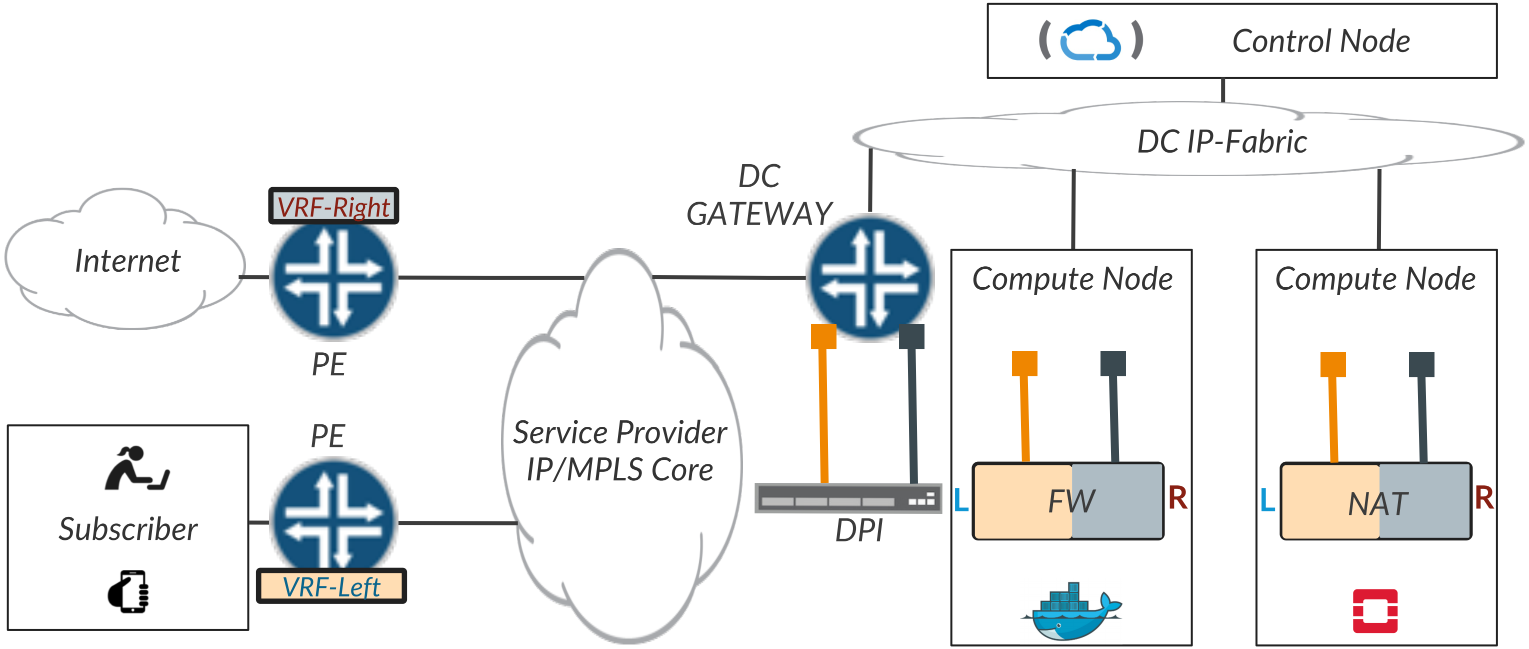
- Contrail
  Contrail GUI for Contrail FW Security Policy
  Kube-Manager introspect port for Network Policy configuration

What is a network function service chain?

- Routing (magic) in the data plane (enabled by vRouter) to steer traffic through a specified set of network functions, in that order.
- Magic = Route re-origination and filter-based forwarding
- Independent of the location and form factor of the network function
- Anchored to Virtual Networks

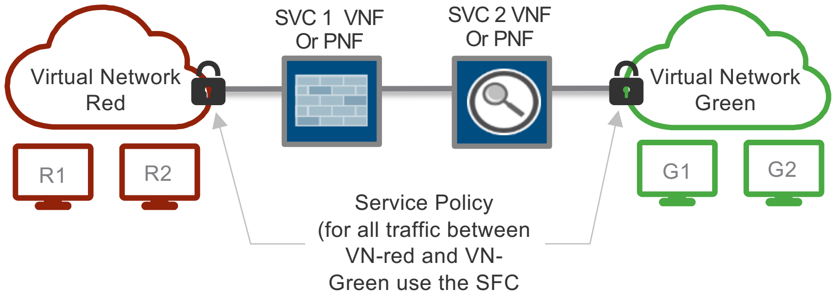

Virtual Network Red
R1
R2
Virtual Network Green
G1
G2
from VN **Red** to VN **Green** first go through SI1 then through SI2

Example Service Chaining Use case

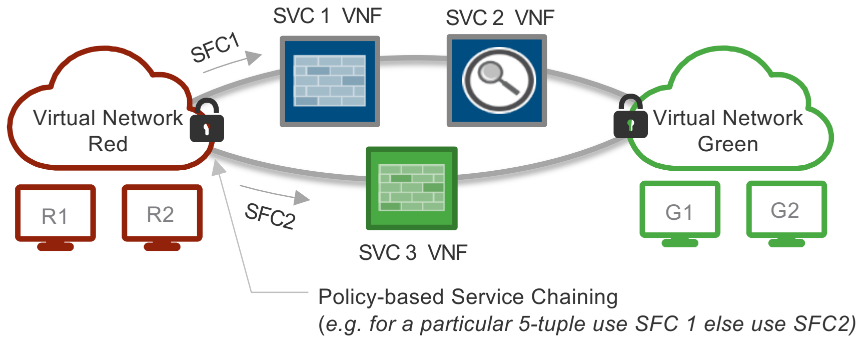


Service Chaining - Common Use Cases

Multiple Services in a Service Chain (includes VNF, PNF)

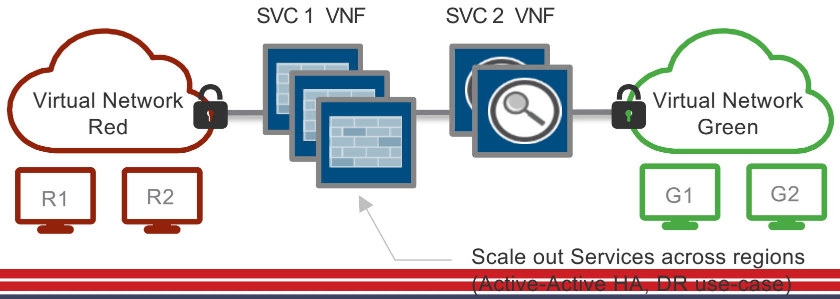


Multiple Service Chains between 2 networks

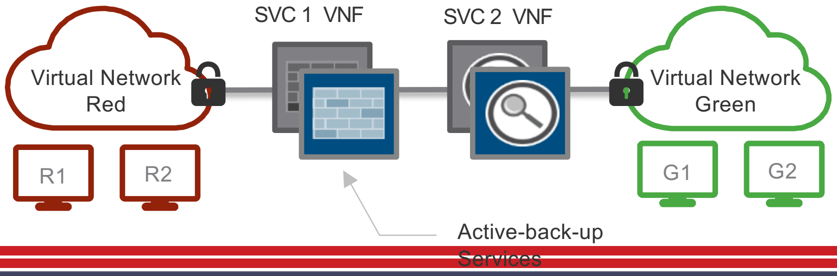


SVC 1 VNF
SVC 2 VNF
SFC1
Virtual Network Red
Virtual Network Green
SFC2
SVC 3 VNF
R1
R2
G1
G2
Policy-based Service Chaining
(e.g. for a particular 5-tuple use SFC 1 else use SFC2)

Multiple Service Instances (Scale-out aka active-active HA)



Service Instances Active-backup HA



Service Chaining – demo

Service Chaining – demo – left-vn and right-vn yaml files

left-vn.yaml

```
apiVersion: "k8s.cni.cncf.io/v1"
kind: NetworkAttachmentDefinition
metadata:
 name: left
 namespace: default
 annotations:
   "opencontrail.org/cidr" : "192.168.1.0/24"
spec:
 config: '{
   "cniVersion": "0.3.1",
   "type": "contrail-k8s-cni"
}'
```

right-vn.yaml

```
apiVersion: "k8s.cni.cncf.io/v1"
kind: NetworkAttachmentDefinition
metadata:
 name: right
 namespace: default
 annotations:
   "opencontrail.org/cidr" : "192.168.2.0/24"
spec:
 config: '{
   "cniVersion": "0.3.1",
   "type": "contrail-k8s-cni"
}'
```

Service Chaining – demo – left-ubuntu pod yaml file

left-ubuntu.yaml

```
apiVersion: v1
kind: Pod
metadata:
  name: left-intf-pod
  annotations:
    k8s.v1.cni.cncf.io/networks: '[
          { "name": ”left" }
    ]'
spec:
  containers:
  - name: ubuntuapp
    image: ubuntu-upstart
    securityContext:
      privileged: true
      capabilities:
        add:
        - NET_ADMIN
```

Service Chaining – demo – right-ubuntu pod yaml file

right-ubuntu.yaml

```
apiVersion: v1
kind: Pod
metadata:
  name: right-intf-pod
  annotations:
    k8s.v1.cni.cncf.io/networks: '[
          { "name": ”right" }
    ]'
spec:
  containers:
  - name: ubuntuapp
    image: ubuntu-upstart
    securityContext:
      privileged: true
      capabilities:
        add:
        - NET_ADMIN
```

Service Chaining – demo – service-ubuntu pod yaml file

service-ubuntu.yaml

```
apiVersion: v1
kind: Pod
metadata:
  name: service-pod
  annotations:
    k8s.v1.cni.cncf.io/networks: ‘[
          {“name”: left” },
          { "name": ”right" }
    ]'
spec:
  containers:
  - name: ubuntuapp
    image: ubuntu-upstart
    securityContext:
      privileged: true
      capabilities:
        add:
        - NET_ADMIN
```

Try Tungsten Fabric

https://tungstenfabric.github.io/website/Tungsten-Fabric-15-minute-deployment-with-k8s-on-AWS.html

DAYONE: BUILDING CONTAINERS WITH KUBERNETES AND CONTRAIL

JUNIPER NETWORKS | Engineering Simplicity

Day One Million

DAY ONE: BUILDING CONTAINERS WITH KUBERNETES AND CONTRAIL

Learn Kubernetes fundamentals and understand its integration with Juniper Contrail®

By Ping Song, Ayman Aborabh, and Yuvaraja Mariappan

https://www.juniper.net/assets/us/en/local/pdf/ebooks/day-one-containers-kubernetes-contrail.pdf

Thank You