

OPENSHIFT NETWORKING

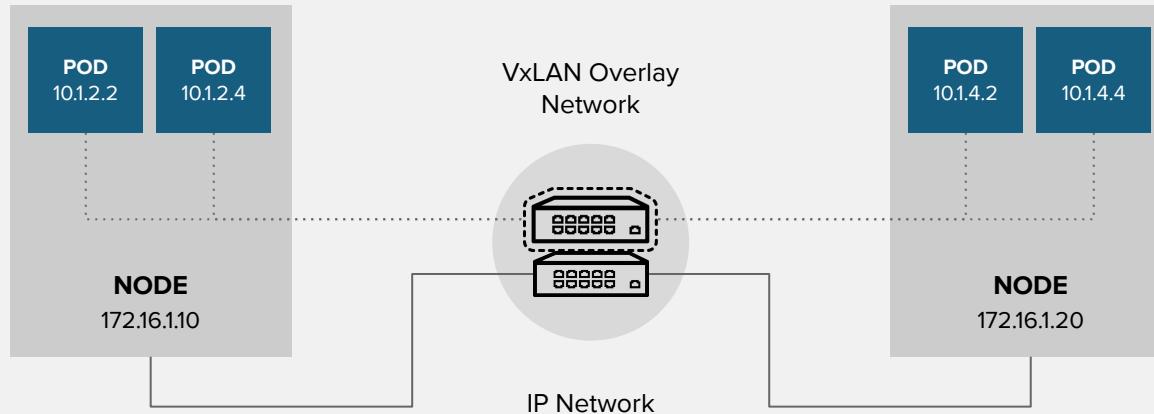
Understanding Kubernetes Networking

Explain networking in Kubernetes
OpenShift SDN
OVN Kubernetes
...

OPENShift NETWORKING

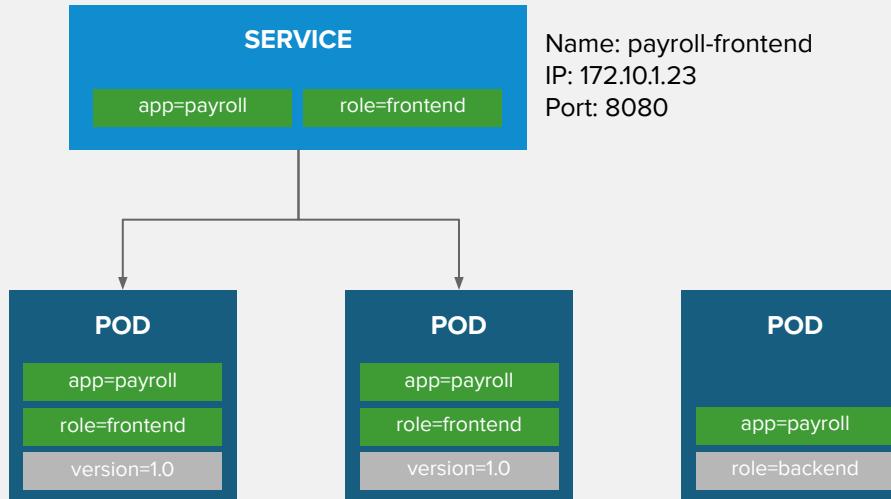
- DNS : Provide name resolution for each Pod
- Service : Virtual access point targeting several Pods
 - Access by IP or DNS, internally or externally
 - Load balance flows to target Pods
 - Dynamic selection of target Pods (dynamic discovery)
- Flow control :
 - Control flows between Pods/Namespaces (Multi-tenancy)
 - Control flows entering/leaving Kubernetes
- External Flows :
 - How to access Pods from outside Kubernetes
 - How to filter on firewalls, flows coming from Kubernetes
 - Give direct access to node network

POD to POD COMMUNICATION



Like a VPN built around Nodes and Pods

BUILT-IN SERVICE DISCOVERY INTERNAL LOAD-BALANCING



client



Ingress

www.app1.com

www.app2.com

www.app3.com

service

service

service

pod

pod

pod

pod

pod

pod

pod

pod

pod

replication controller

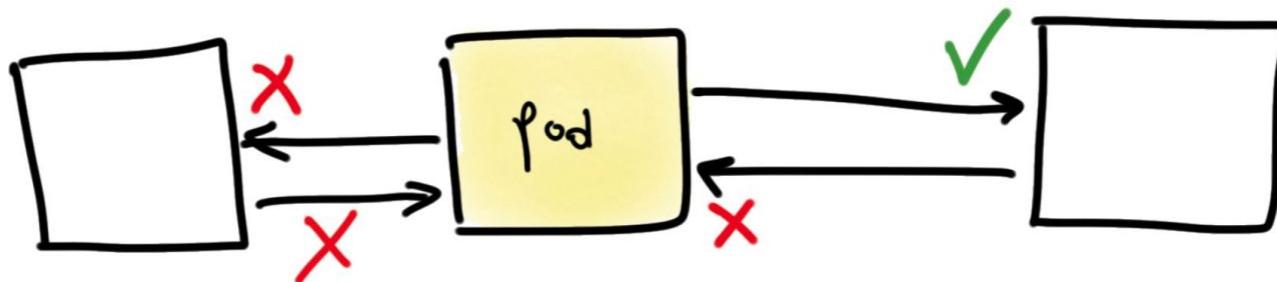
replication controller

replication controller

kubernetes cluster

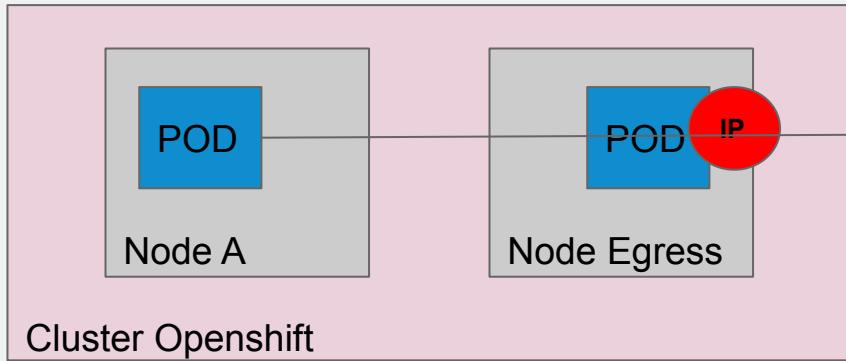
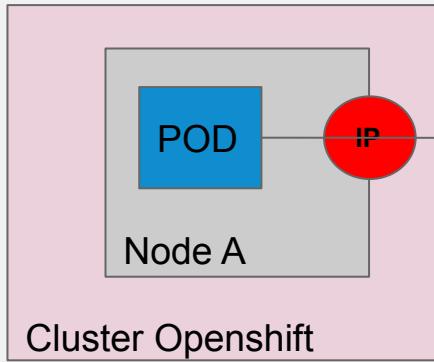
network policies control

traffic from/to pods

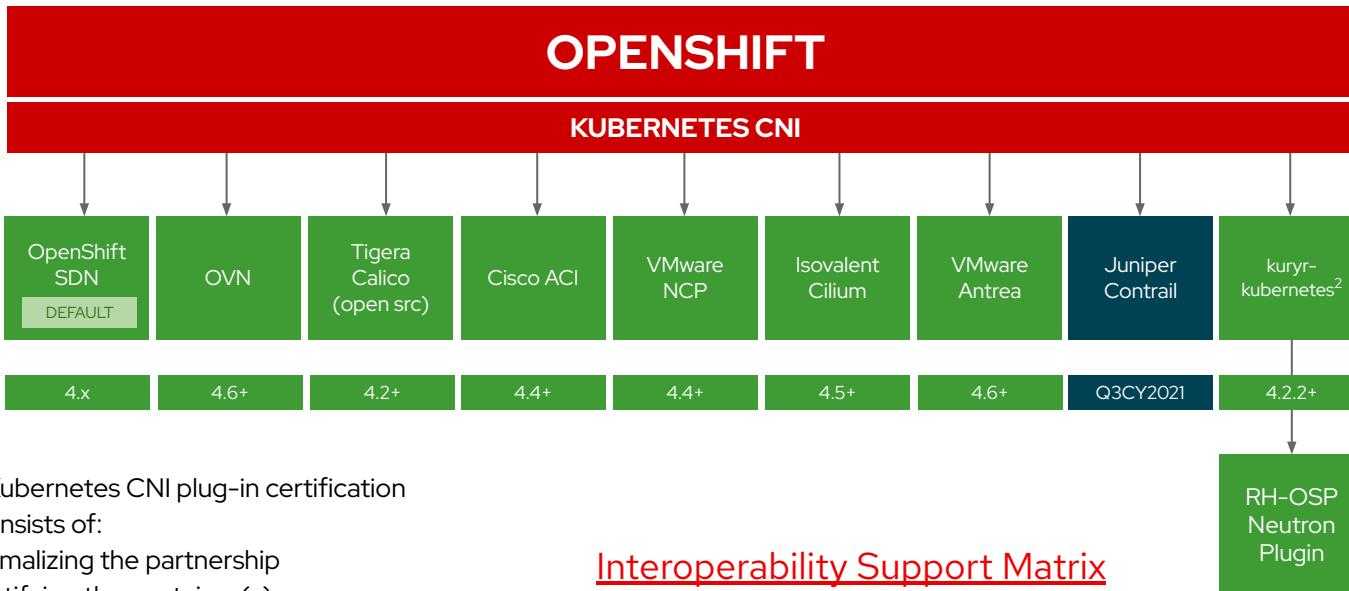




How to filter flows outside OpenShift



OpenShift Networking Plug-ins



3rd-party Kubernetes CNI plug-in certification primarily consists of:

1. Formalizing the partnership
2. Certifying the container(s)
3. Certifying the Operator
4. Successfully passing the same Kubernetes networking conformance tests that OpenShift uses to validate its own SDN

Interoperability Support Matrix

Fully Supported Tech Preview Cert In-Progress



DNS inside the Cluster

DNS: Pod Name Resolution

1. Container is configured with:

```
dnsPolicy: ClusterFirst
```

Resulting in containers with

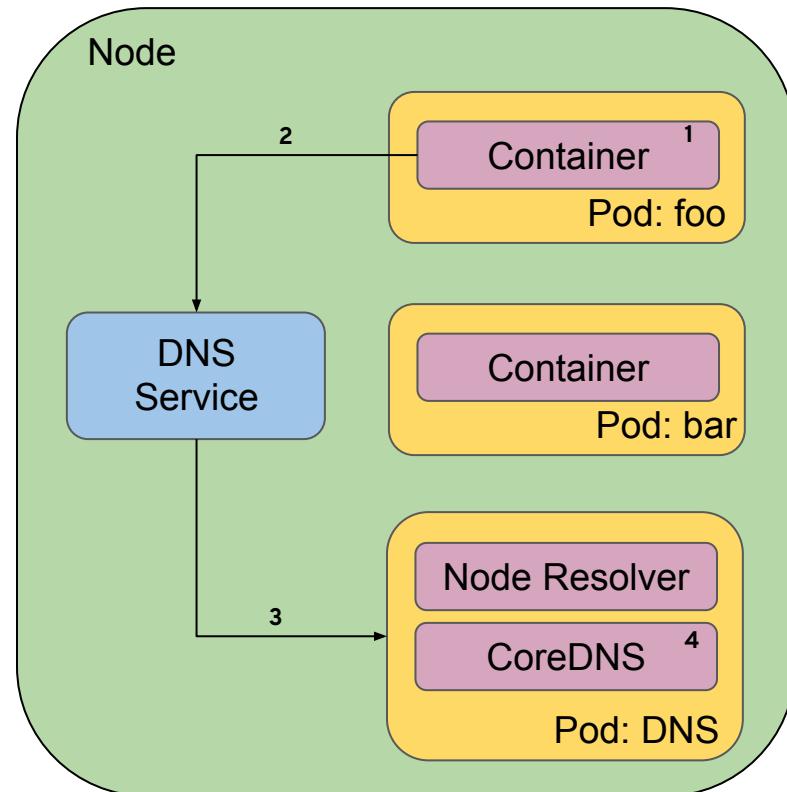
/etc/resolv.conf:

```
nameserver 172.30.0.10 *
```

Container “foo” looks-up “bar” in local cache. “bar” doesn’t exist, so “foo” uses /etc/resolv.conf to figure out how to resolve “bar”.

2. Container “foo” queries nameserver 172.30.0.10 for “bar”.
3. The query gets proxied to the DNS pod
4. CoreDNS resolves bar to “10.128.0.100”.

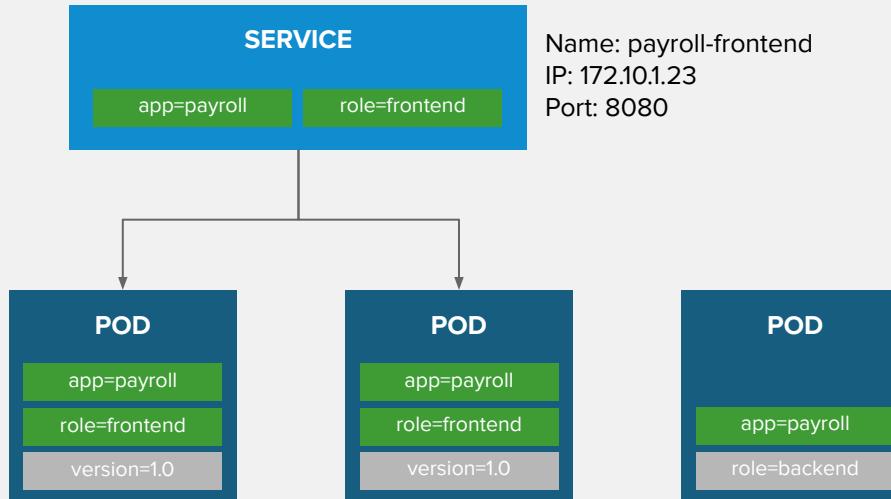
* Default serviceNetwork is 172.30.0.0/16



The background of the slide features a large, modern cable-stayed bridge at night. The bridge's towers and the cables supporting the roadway are brightly lit, creating a strong contrast with the dark, hazy sky above. The water below reflects the bridge's lights.

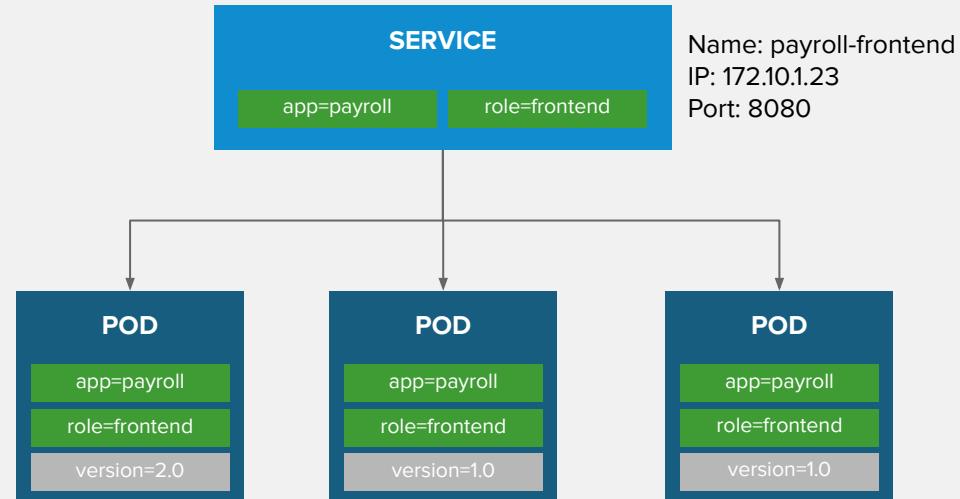
Load balancing inside the Cluster

BUILT-IN SERVICE DISCOVERY INTERNAL LOAD-BALANCING



BUILT-IN SERVICE DISCOVERY INTERNAL LOAD-BALANCING

- When a new pod is deployed on the cluster with the same labels as the service selector, it is automatically added to the list of endpoints load-balanced by the service. If one of the pod gets removed, the service immediately removes it from the list of endpoints and does not send traffic to it anymore.
- Service selector allows load-balancing between multiple versions of the application pods (version=1.0 and version=2.0) in order to achieve A/B testing and other deployment patterns out-of-the-box
- On the slide, the service is configured to discover pods and load-balance traffic between pods that have both app=payroll and role=frontend labels set. Since both v1 and v2 of the frontend pods are deployed and have those labels set, $\frac{1}{3}$ of the traffic would be sent to the frontend v2 to achieve canary release or A/B testing for example.





Network Micro-segmentation

OPENShift Network Multi-tenancy

FLAT NETWORK (Default) : OVS

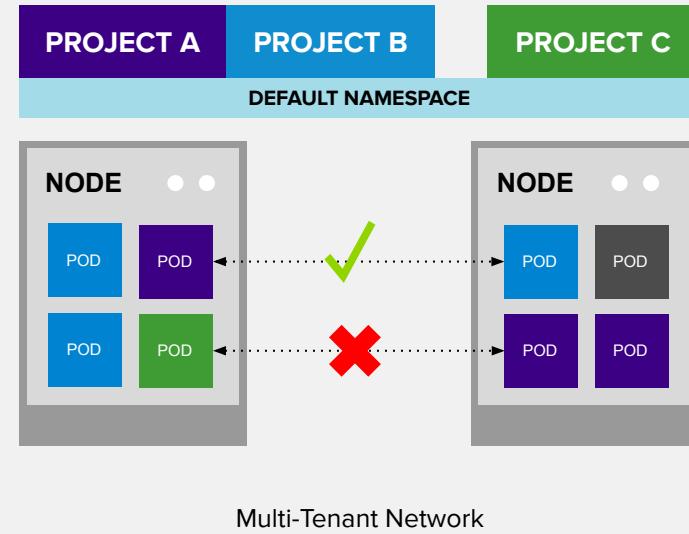
- All pods can communicate with each other across projects

MULTI-TENANT NETWORK: OVS

- Project-level network isolation
- Multicast support
- Egress network policies

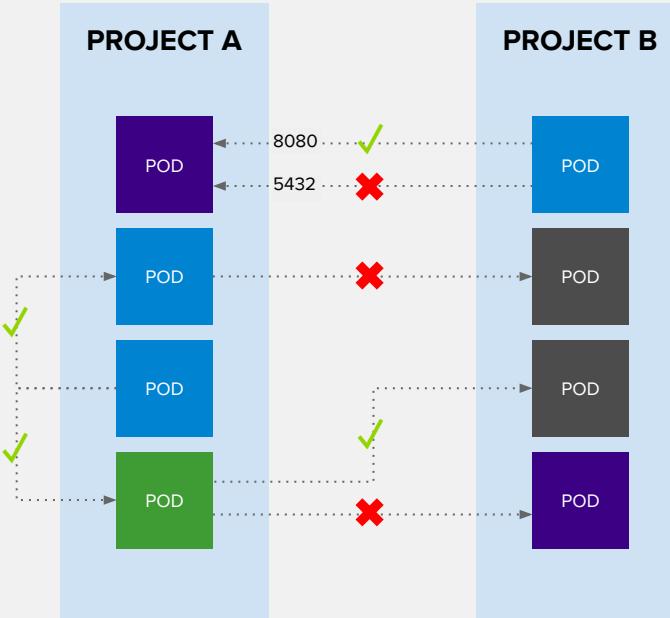
NETWORK POLICY : OVS+OVN

- Granular policy-based isolation



Multi-Tenant Network

OPENShift SDN - NETWORK POLICY



Example Policies

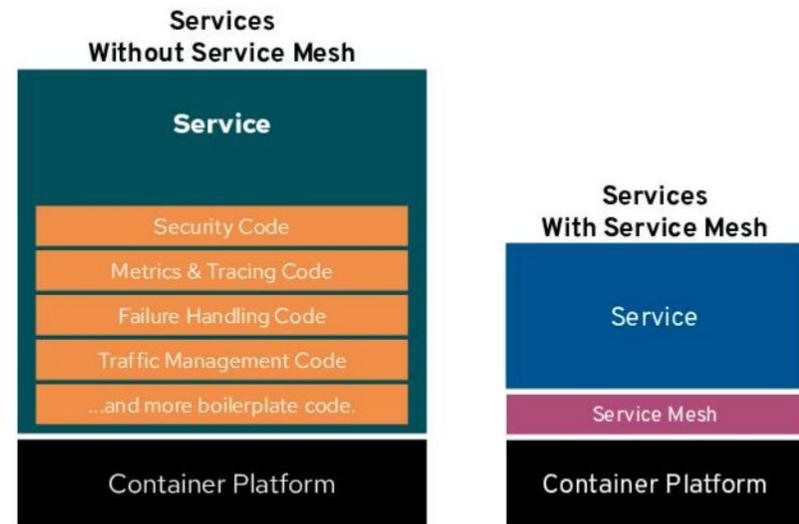
- Allow all traffic inside the project
- Allow traffic from green to gray
- Allow traffic to purple on 8080

```
apiVersion: extensions/v1beta1
kind: NetworkPolicy
metadata:
  name: allow-to-purple-on-8080
spec:
  podSelector:
    matchLabels:
      color: purple
  ingress:
    - ports:
        - protocol: tcp
          port: 8080
```

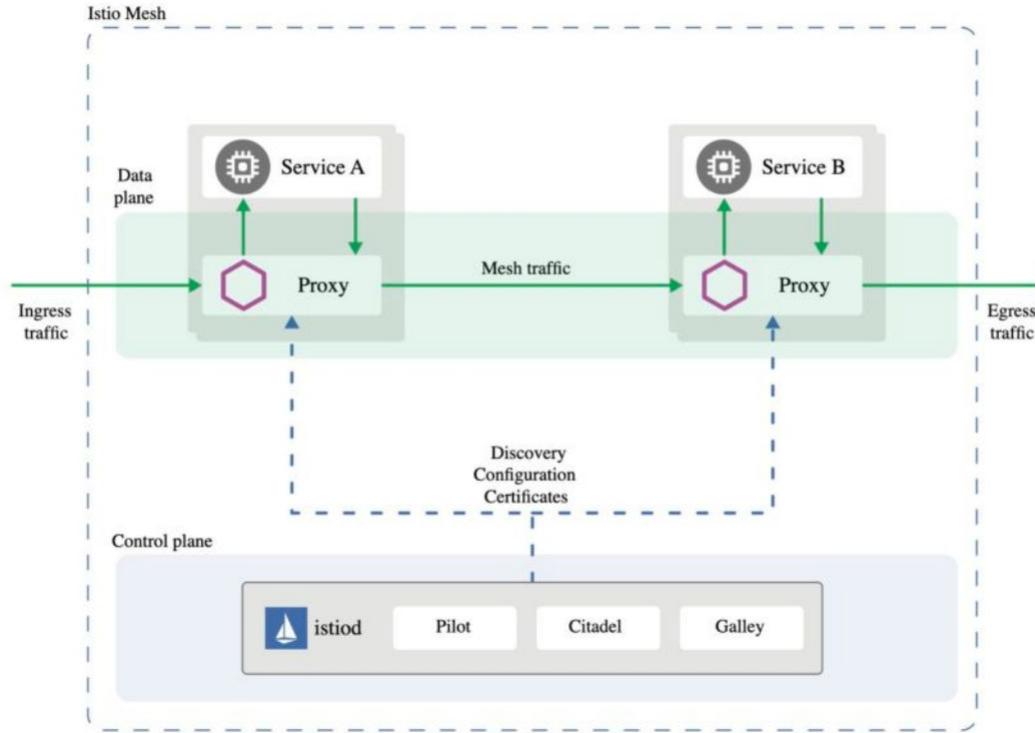
Why Service Mesh?

An Abstraction for Microservice Challenges

- Service Mesh solve distributed systems challenges at **a common infrastructure layer.**
- This reduces boilerplate code and copy/paste errors across services.
- Enforces common policies across all services.
- Removes the obligation to implement cross cutting concerns from developers.

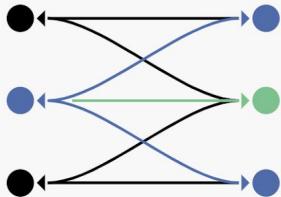


Istio Service Mesh - Architecture



Istio Service Mesh

A modern way to manage the complexity of microservice applications



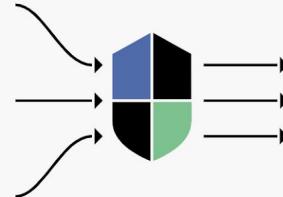
Connect

Intelligently control the flow of traffic and API calls between services, conduct a range of tests, and upgrade gradually with red/black deployments.



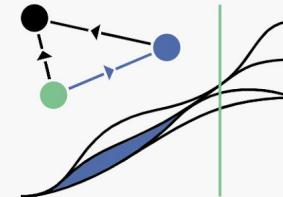
Secure

Automatically secure your services through managed authentication, authorization, and encryption of communication between services.



Control

Apply policies and ensure that they're enforced, and that resources are fairly distributed among consumers.



Observe

See what's happening with rich automatic tracing, monitoring, and logging of all your services.

Handling Partial Failures with the Service Mesh

The diagram illustrates a service mesh topology. On the left, a red-bordered box contains a triangle icon labeled 'app-ui' and a square icon labeled 'latest'. A green arrow labeled '100%' points from 'app-ui' to 'latest'. From 'latest', a red arrow labeled '100%' points to a purple lightning bolt icon labeled 'userprofile'. This lightning bolt is connected to a green arrow labeled 'http 99.9%' pointing to a blue circle labeled 'userprofile'. Another green arrow labeled '0.1%' points from the lightning bolt to a red circle labeled 'userprofile-3'. Above the 'userprofile' node, a grey arrow points to a white square labeled 'userprofile-postgresql'.

Traffic			Response Codes		
HTTP requests per second:			Total	%Success	%Error
35.58	100.00	0.00			

HTTP Request Traffic min / max:
RPS: 27.73 / 31.07 , %Error 0.00 / 0.00

A horizontal bar chart with four categories: OK (green), 3xx (blue), 4xx (orange), and 5xx (red). The 'OK' bar reaches 100% on the x-axis, which ranges from 0 to 100 in increments of 25. Below the chart is a legend: green for OK, blue for 3xx, orange for 4xx, and red for 5xx.

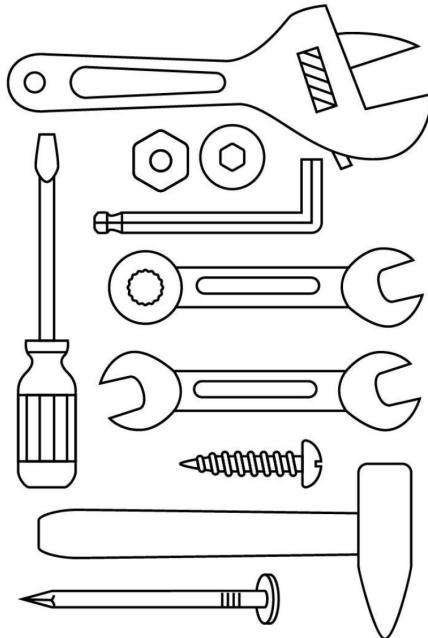
```
26 |     trafficPolicy:
27 |       connectionPool:
28 |         http:
29 |           http1MaxPendingRequests: 1
30 |           maxRequestsPerConnection: 1
31 |     outlierDetection:
32 |       consecutiveErrors: 1
33 |       interval: 1s
34 |       baseEjectionTime: 10m
35 |       maxEjectionPercent: 100
```

YAML configurable,
(Kubernetes native)

Red Hat

21

Features needed to meet the challenges of microservices



SERVICE DISCOVERY

CIRCUIT BREAKING AND BULKHEADS

RATE LIMITING

MIRRORING / TRAFFIC SHIFTING

VERSION BASED ROUTING

AUTOSCALING

STAGED ROLLOUTS

CANARY DEPLOYMENTS

BLUE/GREEN DEPLOYMENTS

DISTRIBUTED TRACING

VISUAL SERVICE HEALTH

DISTRIBUTED LOGGING

COLLECTING AND VISUALIZING METRICS

CHAOS ENGINEERING

SERVICE SECURITY

CONTAINER BUILD AUTOMATION

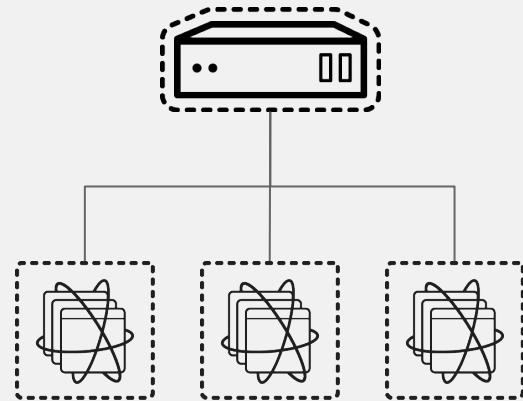
Table Stakes for
doing microservices

The background of the slide features a large, modern cable-stayed bridge at night. The bridge's towers are brightly lit, casting a reflection on the water below. The cables are visible against the dark sky. The overall atmosphere is industrial and architectural.

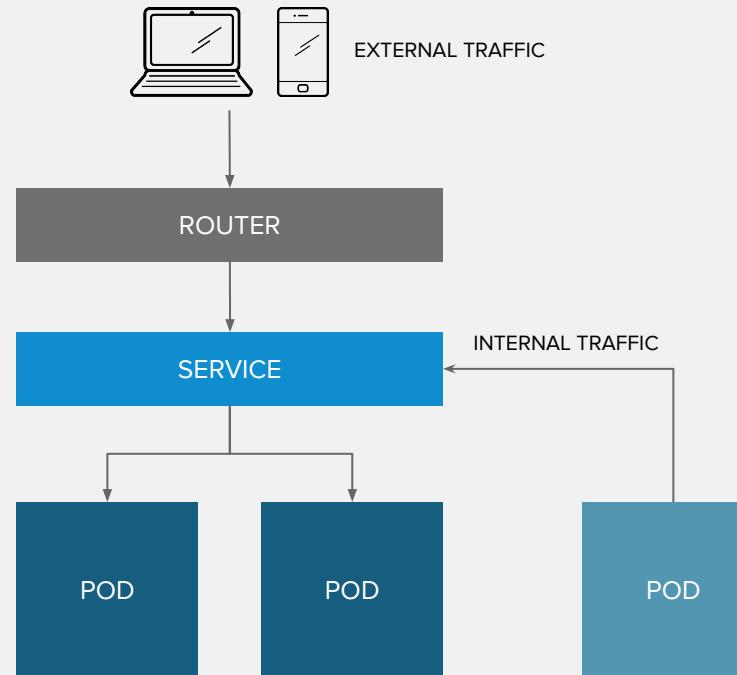
Getting Traffic In the Cluster

ROUTING AND EXTERNAL LOAD-BALANCING

- Pluggable routing architecture
 - HAProxy Router
 - F5 Router
 - NGINX
- Multiple-routers with traffic sharding
- Router supported protocols
 - HTTP/HTTPS
 - WebSockets
 - TLS with SNI
- Non-standard ports via cloud load-balancers, external IP, and NodePort



ROUTE EXPOSES SERVICES EXTERNALLY

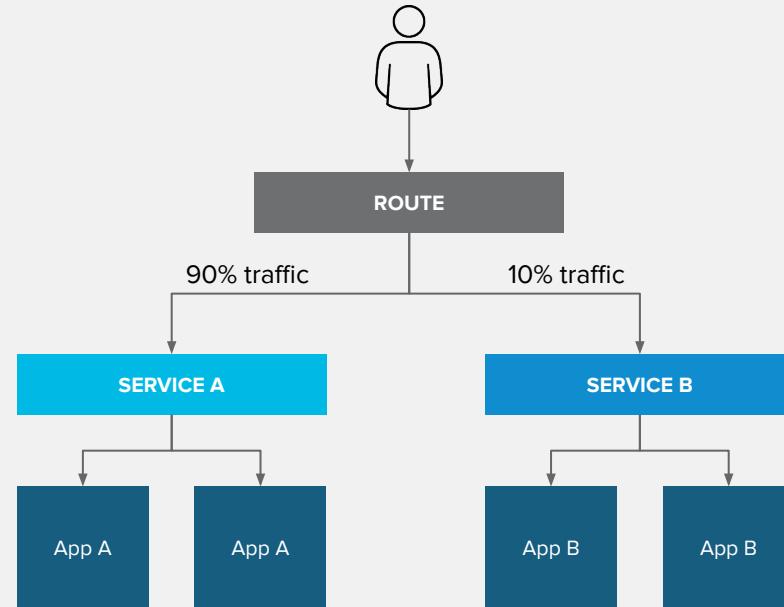


Routes vs Ingress

Feature	Ingress	Route
Standard Kubernetes object	X	
External access to services	X	X
Persistent (sticky) sessions	X	X
Load-balancing strategies (e.g. round robin)	X	X
Rate-limit and throttling	X	X
IP whitelisting	X	X
TLS edge termination	X	X
TLS re-encryption	X	X
TLS passthrough	X	X
Multiple weighted backends (split traffic)		X
Generated pattern-based hostnames		X
Wildcard domains		X

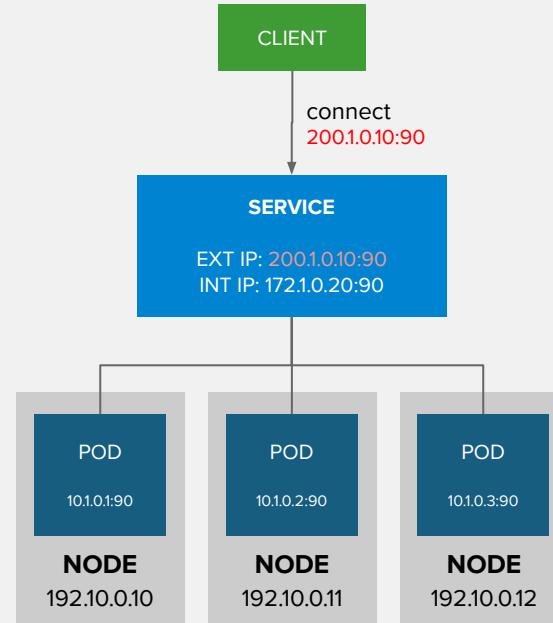
ROUTE SPLIT TRAFFIC

Split Traffic Between
Multiple Services For A/B
Testing, Blue/Green and
Canary Deployments



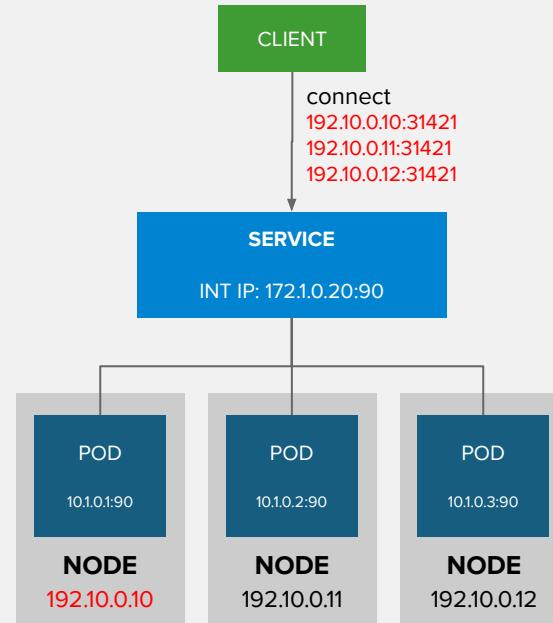
EXTERNAL TRAFFIC TO A SERVICE ON ANY PORT WITH INGRESS

- Access a service with an external IP on any TCP/UDP port, such as
 - Databases
 - Message Brokers
- Automatic IP allocation from a predefined pool using Ingress IP Self-Service
- IP failover pods provide high availability for the IP pool



EXTERNAL TRAFFIC TO A SERVICE ON A RANDOM PORT WITH NODEPORT

- NodePort binds a service to a unique port on all the nodes
- Traffic received on any node redirects to a node with the running service
- Ports in 30K-60K range which usually differs from the service
- Firewall rules must allow traffic to all nodes on the specific port



The background of the slide features a large, modern cable-stayed bridge at night. The bridge's towers are brightly lit, casting a reflection on the water below. The cables are visible against the dark sky. The overall atmosphere is industrial and architectural.

Getting Traffic Out of the Cluster

Getting Traffic Out of the Cluster

- Default: Traffic is NAT'd to the host IP of the Node running the Pod

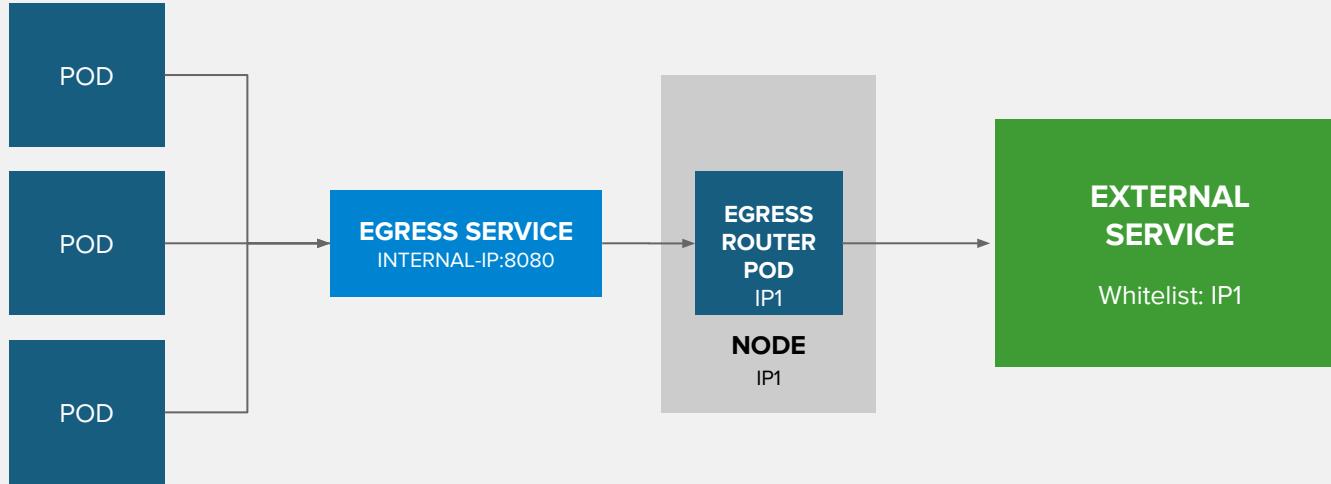


Getting Traffic Out of the Cluster

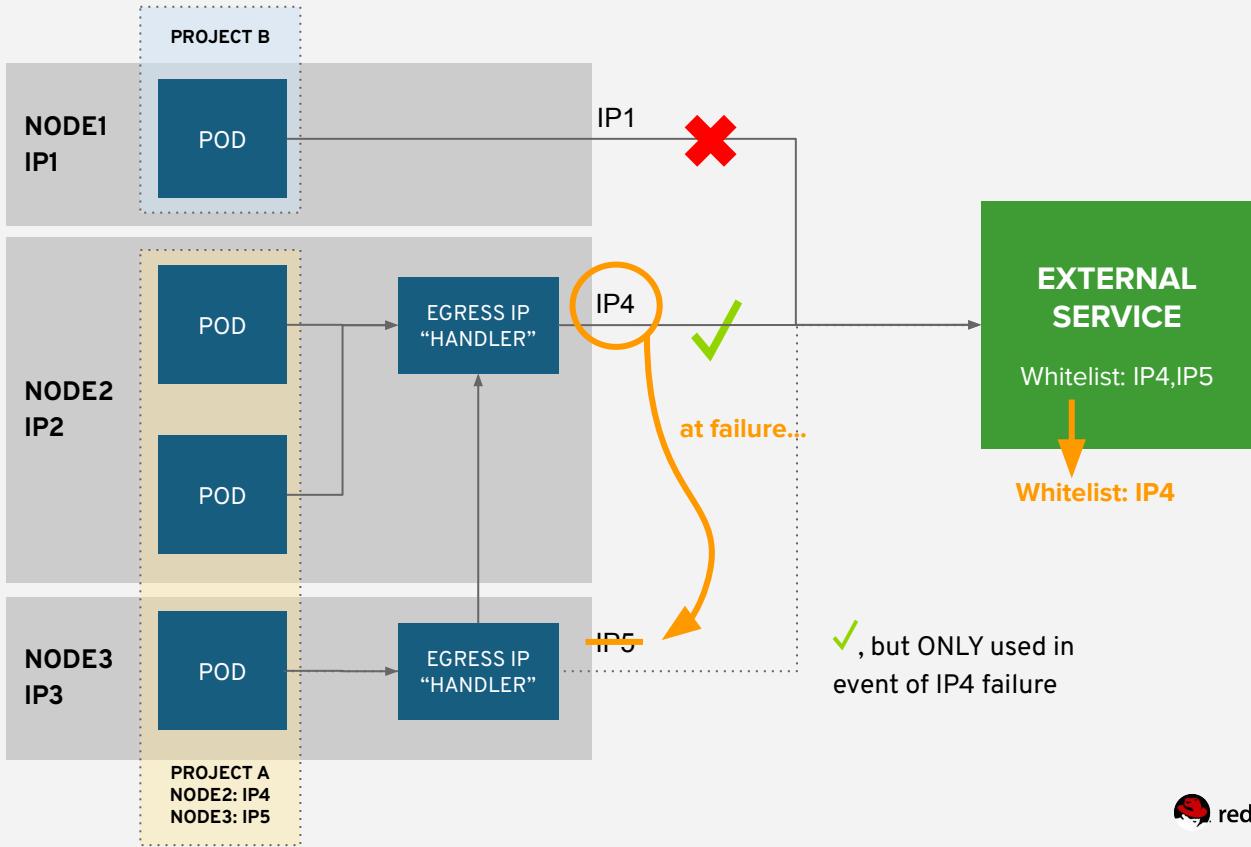
- Can control egress traffic in two ways
 - **Firewall** - using egress firewall allows you to enforce the acceptable outbound traffic - Still NAT'd
 - **Router** - An egress router allows you to create identifiable services to send traffic to a specific destination ensuring the destination treat as if originating from a known source



CONTROL OUTGOING TRAFFIC SOURCE IP WITH EGRESS ROUTER



Fully Automatic, HA Namespace-Wide Egress IP

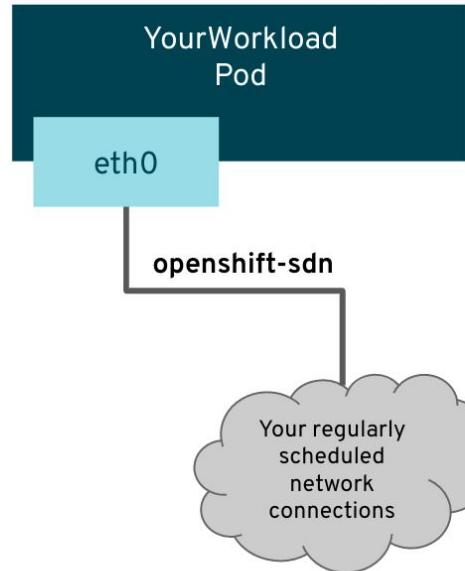


The background of the slide features a large, modern cable-stayed bridge at night. The bridge's towers are brightly lit, casting a reflection on the water below. The cables are visible against the dark sky. The overall atmosphere is industrial and architectural.

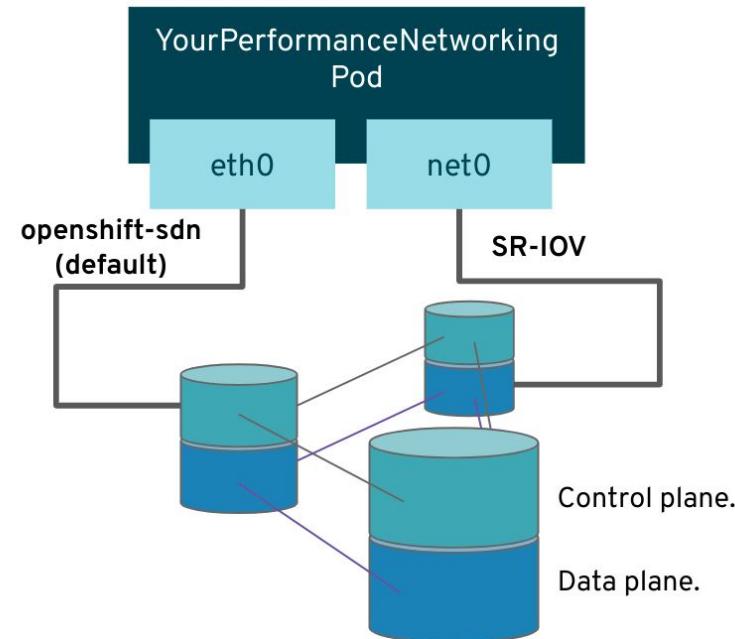
Using direct access to a host network

WHAT MULTUS DOES

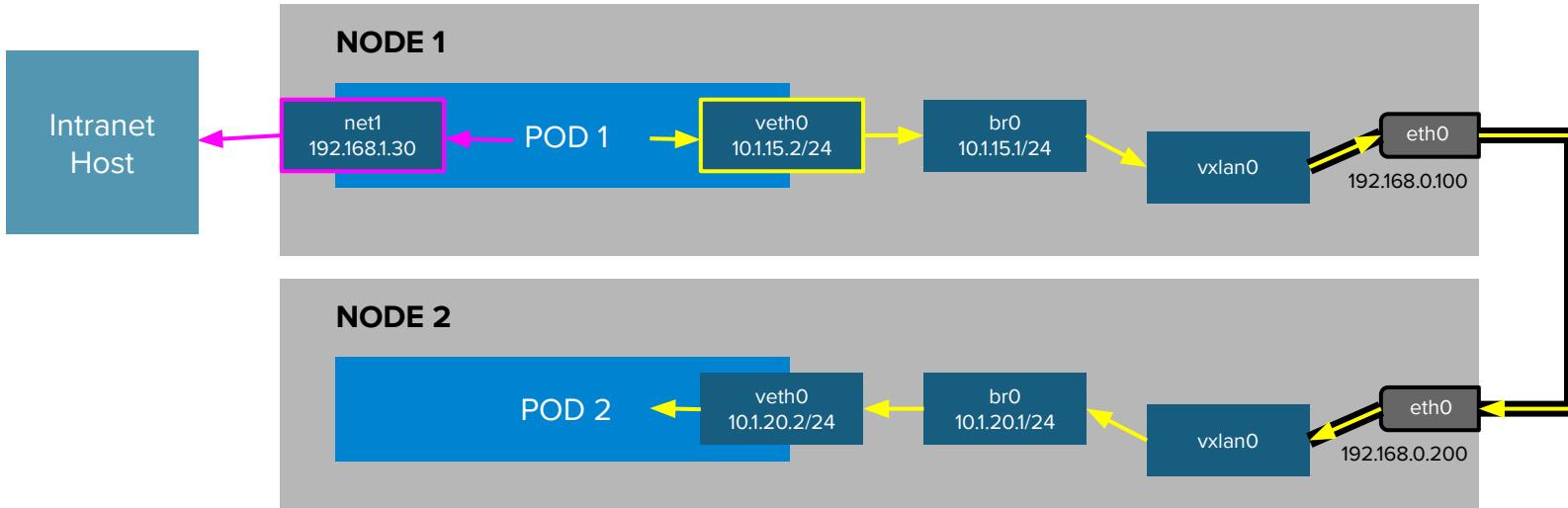
Pod without Multus



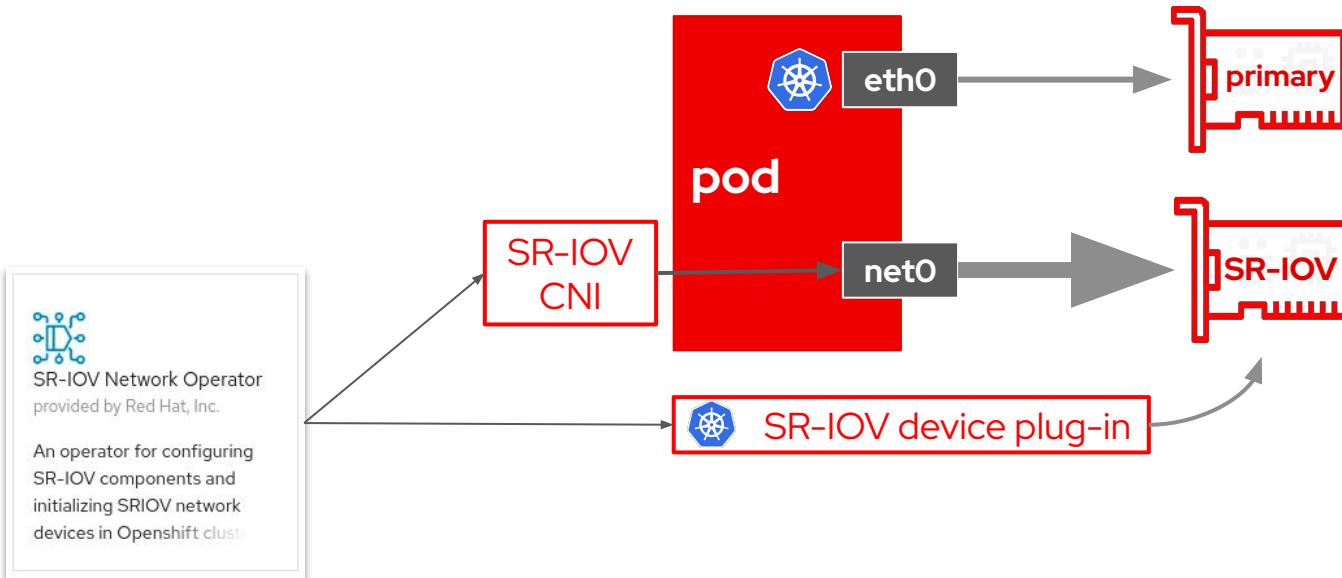
Pod with Multus



Multus Packet Flow



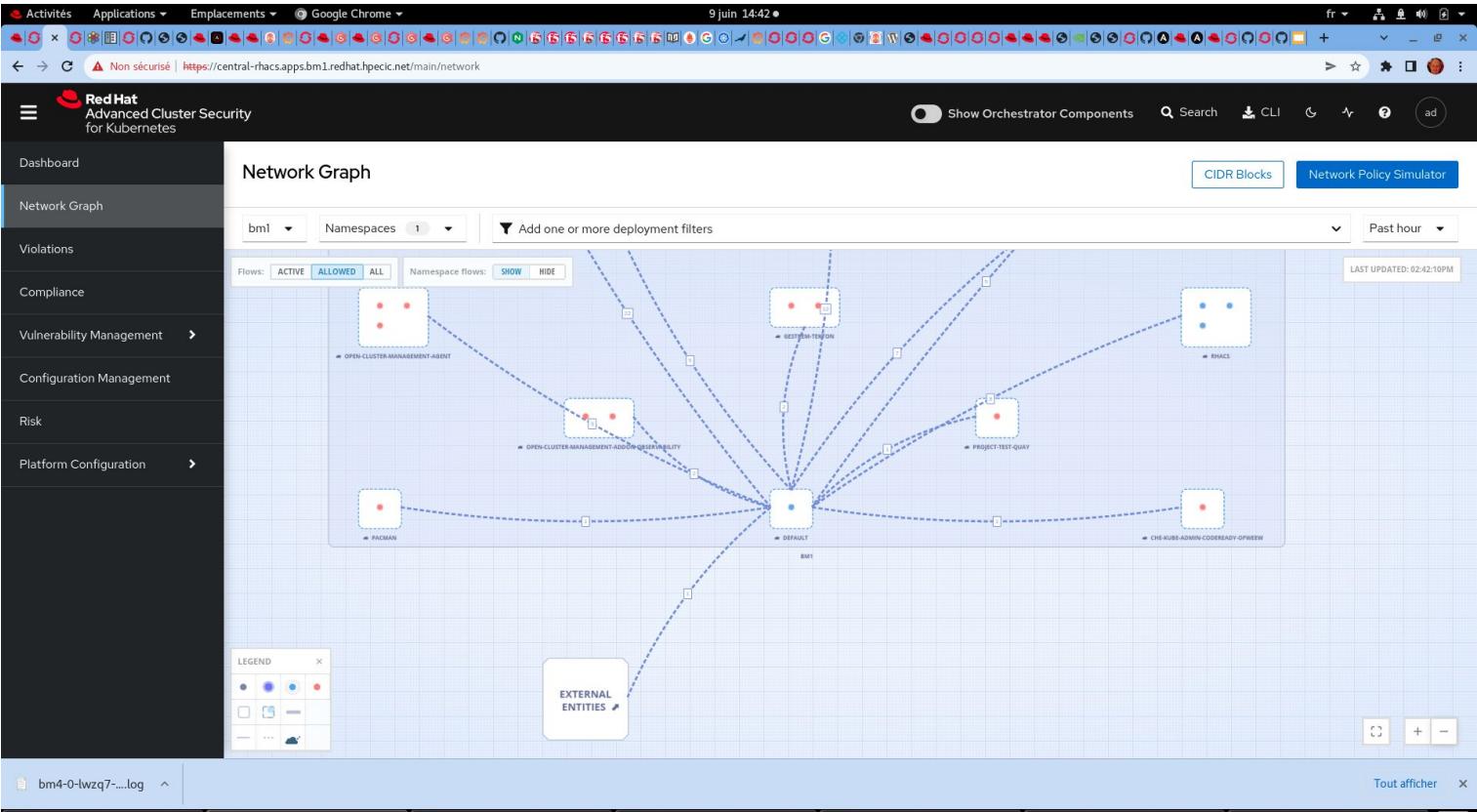
SR-IOV





Advanced Cluster Security Network Policy Management

ACS Network Graph



ACS Network Graph

Activités Applications Emplacements Google Chrome

Non sécurisé | https://central-rhacs.apps.bm1.redhat.hpeic.net/main/network

Red Hat Advanced Cluster Security for Kubernetes

Dashboard Network Graph Violations Compliance Vulnerability Management Configuration Management Risk Platform Configuration

Network Graph

bm1 Namespaces 1 Add one or more deployment filters

Flows: ACTIVE ALLOWED ALL Namespace flows: SHOW HIDE LAST UPDATED: 02:38:36PM

OPENSOURCE-OPERATORS OPEN-CLUSTER-MANAGEMENT-AGENT-ADDON OPEN-CLUSTER-MANAGEMENT-AGENT RHACS GESTRE-TEKTON LEGEND

ANSIBLE 12 13 14

PIPLINES-TUTORIAL

41

Red Hat

bm4-0-lwzq7...log Tout afficher

Red Hat @ HPE CIC - Google She... Network Graph | Red Hat Advanc... root@fdaval0:~ hit@bm1:~ hit@bm1:~ *Document 1 sans titre - gedit Keepass2Android Password Datab... 2 / 7

ACS

Generate Network Policies / Upload and simulate Network Policy

The screenshot shows the Red Hat Advanced Cluster Security (ACS) web interface. The left sidebar contains navigation links: Activities, Applications, Emplacements, Google Chrome, Dashboard, Network Graph (selected), Violations, Compliance, Vulnerability Management, Configuration Management, Risk, and Platform Configuration.

The main area features a "Network Graph" with a grid of nodes representing different components like OpenShift Operators, Metrics, Log View, OpenShift Management Addon, External Entities, and Metrics. A legend at the bottom left defines node types: blue dots for Nodes, red dots for Services, green squares for Namespaces, grey lines for External Entities, and a cloud icon for External Services.

A top navigation bar includes tabs for "bml" and "Namespaces", a filter for "Add one or more deployment filters", and time controls for "Past hour". The status bar shows "LAST UPDATED: 02:38:36PM".

A modal window titled "SELECT AN OPTION" is open, offering two paths:

- Generate network policies:** Describes generating recommended policies based on environment configuration. It includes a checkbox for "Exclude Ports & Protocols" and a button labeled "Generate and simulate network policies".
- Upload a network policy YAML:** Describes uploading existing policies to preview the environment. It includes a "UPLOAD AND SIMULATE NETWORK POLICY YAML" button with an upload icon.

The bottom of the interface shows a file list with "bm4-0-lwzq7-....log" and a "Tout afficher" (Show all) link. The bottom right corner features the Red Hat logo.

Network Policy Simulator

Activités Applications Emplacements Google Chrome 9 juin 14:38

Non sécurisé | https://central-rhacs.apps.bm1.redhat.hpeic.net/main/network

Red Hat Advanced Cluster Security for Kubernetes

Dashboard Network Graph Violations Compliance Vulnerability Management Configuration Management Risk Platform Configuration

Network Graph

SIMULATED VIEW STOP LAST UPDATED: 02:38:14PM

Add one or more deployment filters

CIDR Blocks Network Policy Simulator

Policies generated from network activity in the past hour

STACKROX GENERATED

```
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  creationTimestamp: "2022-06-09T12:38:14Z"
  labels:
    network-policy-generator.stackrox.io/generated: "true"
    name: stackrox-generated-tekton-operator-webhook
    namespace: openshift-operators
spec:
  ingress:
    - ports:
        - port: 8443
          protocol: TCP
    podSelector:
      matchLabels:
        name: tekton-operator-webhook
    policyTypes:
      - Ingress
  
```

apiVersion: networking.k8s.io/v1
kind: NetworkPolicy

Apply Network Policies Share YAML

Legend

EXTERNAL ENTITIES

bm4-0-lwzq7....log Tout afficher

Network Graph | Red Hat Advanced Cluster Security for Kubernetes

root@fdavalos:~ hit@bm1:~ hit@bm1:~ *Document 1 sans titre - gedit

Keepass2Android Password Database

Network policy enforcement on namespaces

The screenshot shows the Red Hat Advanced Cluster Security for Kubernetes interface. The left sidebar has a dark theme with white text and icons. The 'Violations' section is selected. The main area displays a table of violations:

Violation Description	Resource	Type	Status	Severity	Category	Action	Last Update	More
Deployments should have at least one ingress Network Policy	pacman-app in "bm1/pacman"	deployment	No	Medium	Security Best Practices	Deploy	06/10/2022 11:44:43AM	...
Deployments should have at least one ingress Network Policy	klusterlet-registration-agent in "bm1/open-cluster-management-agent"	deployment	No	Medium	Security Best Practices	Deploy	06/10/2022 11:44:43AM	...
Deployments should have at least one ingress Network Policy	virt-launcher-test-bdd-n95k9 in "bm1/lamp-virt"	deployment	No	Medium	Security Best Practices	Deploy	06/10/2022 11:44:45AM	...
Deployments should have at least one ingress Network Policy	virt-launcher-rhel7-beneficial-donkey-s7kqg in "bm1/lamp-virt"	deployment	No	Medium	Security Best Practices	Deploy	06/10/2022 11:44:45AM	...
Deployments should have at least one ingress Network Policy	workspace56e8f337381a4542 in "bm1/kube-admin-codeready-ofweew"	deployment	No	Medium	Security Best Practices	Deploy	06/10/2022 11:44:44AM	...
Deployments should have at least one ingress Network Policy	aap-controller-postgres in "bm1/ansible"	deployment	No	Medium	Security Best Practices	Deploy	06/10/2022 11:44:44AM	...
Deployments should have at least one ingress Network Policy	endpoint-observability-operator in "bm1/open-cluster-management-addon-observability"	deployment	No	Medium	Security Best Practices	Deploy	06/10/2022 11:44:43AM	...
Deployments should have at least one ingress Network Policy	governance-policy-framework in "bm1/open-cluster-management-agent-addon"	deployment	No	Medium	Security Best Practices	Deploy	06/10/2022 11:44:45AM	...
Deployments should have at least one ingress Network Policy	aap-hub-worker in "bm1/ansible"	deployment	No	Medium	Security Best Practices	Deploy	06/10/2022 11:44:45AM	...
Deployments should have at least one ingress Network Policy	cert-policy-controller in "bm1/open-cluster-management-agent-addon"	deployment	No	Medium	Security Best Practices	Deploy	06/10/2022 11:44:43AM	...
Deployments should have at least one ingress Network Policy	aap-hub-postgres in "bm1/ansible"	deployment	No	Medium	Security Best Practices	Deploy	06/10/2022 11:44:46AM	...
Deployments should have at least one ingress Network Policy	nodejs-basic in "bm1/project-test-quay"	deployment	No	Medium	Security Best Practices	Deploy	06/10/2022 11:44:46AM	...
Deployments should have at least one ingress Network Policy	aap-hub-api	deployment	No	Medium	Security Best	Deploy	06/10/2022	...

At the bottom, there are several terminal windows and a file viewer. The terminal windows show command-line interactions like 'root@fdaval0:~\$' and 'fdaval0@fdaval0:~\$'. The file viewer shows 'untitled-policy.yaml' and 'Document 1 sans titre - gedit'. A status bar at the bottom right shows 'Tout afficher' and '2 / 5'.



Monitoring network flows

Observability Networking



**Unified
Experience**

Network Traffic Metrics and Tracing

- Whether one cluster or one hundred, developers and cluster administrators require seamless connectivity across applications.



**Security
Everywhere**

Network Policy and Governance

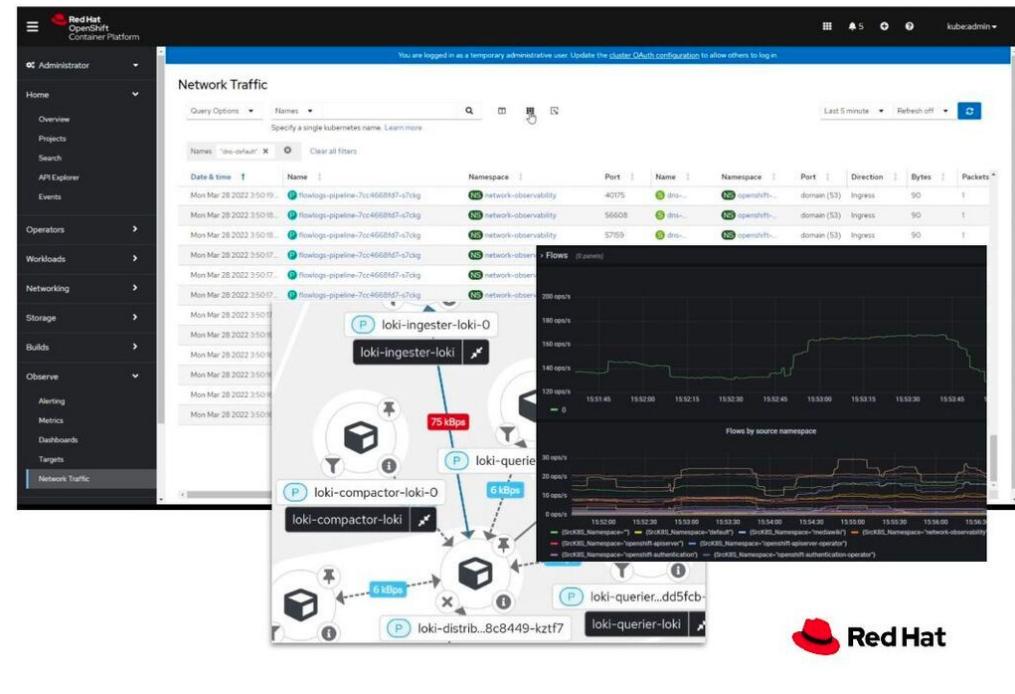
- Security and regulatory compliance requires governance of traffic in, around, and out of networks.



**Platform
Consistency**

Network Traffic Flow and Topology

- Developers and administrators require a common understanding of their traffic within and across cluster boundaries.



What is Network Observability?

Network Observability provides **real-time visualization** and **insight** into your network.

It goes beyond traditional monitoring which includes **metrics**, **logs**, and **tracing**.

Monitoring tells you **what** happened. Observability tells you **why** it happened.

Six Functional Areas of Network Observability



Network flow visibility

What traffic is flowing in, around, and out of my network?



Network topology

What physical and virtual resources are connected to my network?



Network metrics

High-level performance metrics to indicate the overall health of my network



Network tracing

What path does specific traffic take through various interfaces to its final destination?



Network policy

Rules governing the flow of traffic in, around, and out of my network



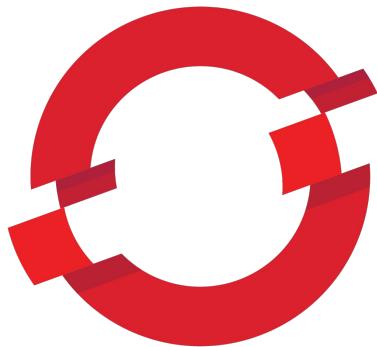
Network traffic capture

Capture + mirror to an external location for debugging, audit, or lawful capture



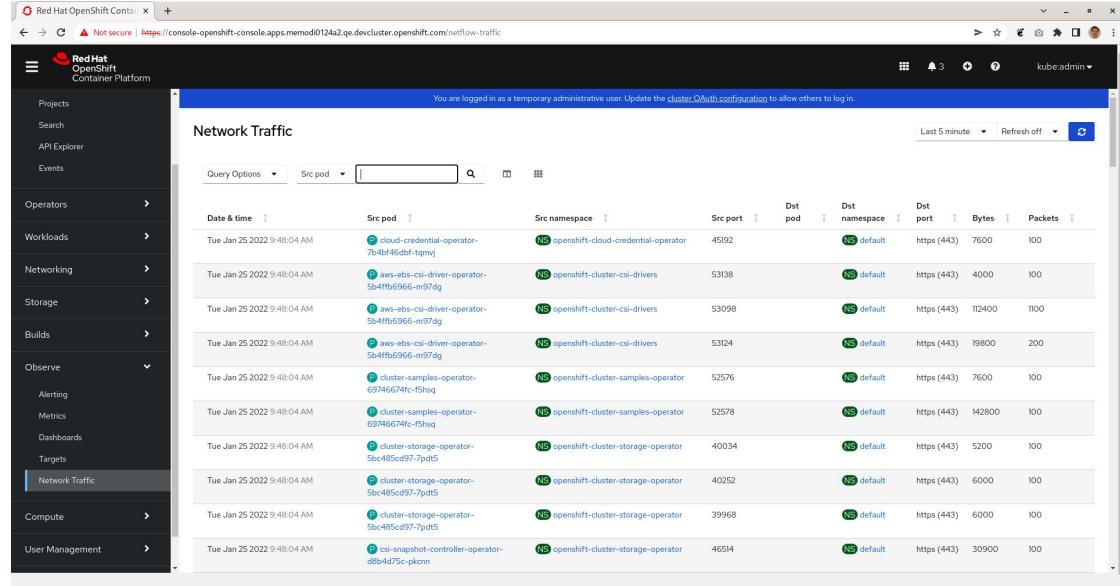
Network Observability in OCP 4.10

(Dev Preview)



- Collect and store NetFlows
 - GoFlow2-Kube Enricher
 - NetFlow table and graphs
 - Network Observability UI Plugin
 - Network Observability Operator
-
- Single cluster
 - OVN-Kubernetes only
 - kubeadmin (not developer)

NetFlow Table in OCP 4.10



The screenshot shows the Red Hat OpenShift Container Platform web console with the URL <https://console-openshift-console.apps.memdb0124a2.qe.devcluster.openshift.com/netflow-traffic>. The user is logged in as a temporary administrative user. The left sidebar shows various navigation options like Projects, Search, API Explorer, Events, Operators, Workloads, Networking, Storage, Builds, Observe, Alerting, Metrics, Dashboards, Targets, Compute, and User Management. The 'Network Traffic' option under the 'Observe' section is selected. The main content area displays a table titled 'Network Traffic' with the following columns: Date & time, Src pod, Src namespace, Src port, Dst pod, Dst namespace, Dst port, Bytes, and Packets. The table lists network traffic entries from Tuesday, Jan 25, 2022, at 9:48:04 AM. The first entry is from 'cloud-credential-operator-7b4bf46dbf-tqmsq' to 'openshift-cloud-credential-operator' on port 45192. Other entries include traffic between various CSI drivers and their respective namespaces.

Date & time	Src pod	Src namespace	Src port	Dst pod	Dst namespace	Dst port	Bytes	Packets
Tue Jan 25 2022 9:48:04 AM	cloud-credential-operator-7b4bf46dbf-tqmsq	openshift-cloud-credential-operator	45192		default	https (443)	7600	100
Tue Jan 25 2022 9:48:04 AM	aws-efs-csi-driver-operator-5b4ff6b966-m97dg	openshift-cluster-csi-drivers	53138		default	https (443)	4000	100
Tue Jan 25 2022 9:48:04 AM	aws-efs-csi-driver-operator-5b4ff6b966-m97dg	openshift-cluster-csi-drivers	53098		default	https (443)	112400	1100
Tue Jan 25 2022 9:48:04 AM	aws-efs-csi-driver-operator-5b4ff6b966-m97dg	openshift-cluster-csi-drivers	53124		default	https (443)	19800	200
Tue Jan 25 2022 9:48:04 AM	cluster-samples-operator-69746674fc-f5hsq	openshift-cluster-samples-operator	52576		default	https (443)	7600	100
Tue Jan 25 2022 9:48:04 AM	cluster-samples-operator-69746674fc-f5hsq	openshift-cluster-samples-operator	52578		default	https (443)	142800	100
Tue Jan 25 2022 9:48:04 AM	cluster-storage-operator-5bc485cd97-7p9t5	openshift-cluster-storage-operator	40034		default	https (443)	5200	100
Tue Jan 25 2022 9:48:04 AM	cluster-storage-operator-5bc485cd97-7p9t5	openshift-cluster-storage-operator	40252		default	https (443)	6000	100
Tue Jan 25 2022 9:48:04 AM	cluster-storage-operator-5bc485cd97-7p9t5	openshift-cluster-storage-operator	39968		default	https (443)	6000	100
Tue Jan 25 2022 9:48:04 AM	csi-snapshot-controller-operator-dtb4fd7sc-pcmn	openshift-cluster-storage-operator	46514		default	https (443)	30900	100

Features

- Pod-to-pod traffic
- Namespaces / projects
- Map port to service names
- Column selection
- Filtering
- Sorting
- Time range
- Refresh interval

Network observability operator

The screenshot shows the Red Hat OpenShift Container Platform interface. The left sidebar navigation includes:

- Activités
- Applications
- Emplacements
- Google Chrome
- Red Hat OpenShift Container Platform
- StorageClasses
- VolumeSnapshots
- VolumeSnapshotClasses
- VolumeSnapshotContents
- Object Buckets
- Object Bucket Claims
- Builds
- Pipelines
- Observe
 - Alerting
 - Metrics
 - Dashboards
 - Targets
 - Network Traffic
- Compute
- User Management
- Administration
 - Cluster Settings
 - Namespaces

The "Network Traffic" option is selected in the "Observe" menu.

The main content area displays the "Network Traffic" dashboard. It features a "Network Traffic" title bar with "Query Options", "Common Namespace" dropdown (set to "ansible"), a search bar, and filter buttons for "Common Namespace" (selected) and "Clear all filters". Below this is a "Find in view" search bar.

The central part of the dashboard is a network topology diagram showing nodes and their connections. Nodes include "aap-hub-work-54c87c-gbqcj", "aap-hub-postgres-0", "aap-hub-cont-4f6ccb-n5ijk", "aap-hub-api-79b997b9fc-xqn2k", "aap-hub-redu-8cf057-t5", "automation-c-48d9cc7fb", and "kubernetes" (with namespaces "default" and "ansible"). Arrows indicate traffic flow between nodes, with labels such as "791 kB", "790 kB", "141 kB", and "151 kB".

At the bottom of the dashboard are several small icons for filtering and searching.

The browser status bar at the bottom shows multiple tabs and windows, including "root@fdavalov:~" and "Red Hat OpenShift Container Plat...".

Network observability operator

Activités Applications Emplacements Google Chrome 10 juin 15:06

console-openshift-console.apps.bm1.redhat.hpeic.net/netflow-traffic?filters=namespace%3Dansible&timeRange=300&limit=100&match=all&reporter=destination&function=sum&type=bytes

Red Hat OpenShift Container Platform

You are logged in as a temporary administrative user. Update the cluster OAuth configuration to allow others to log in.

Network Traffic

Query Options Common Namespace Manage columns Display size Export Last 5 minutes Refresh off

Specify a single Kubernetes name. [Learn more](#)

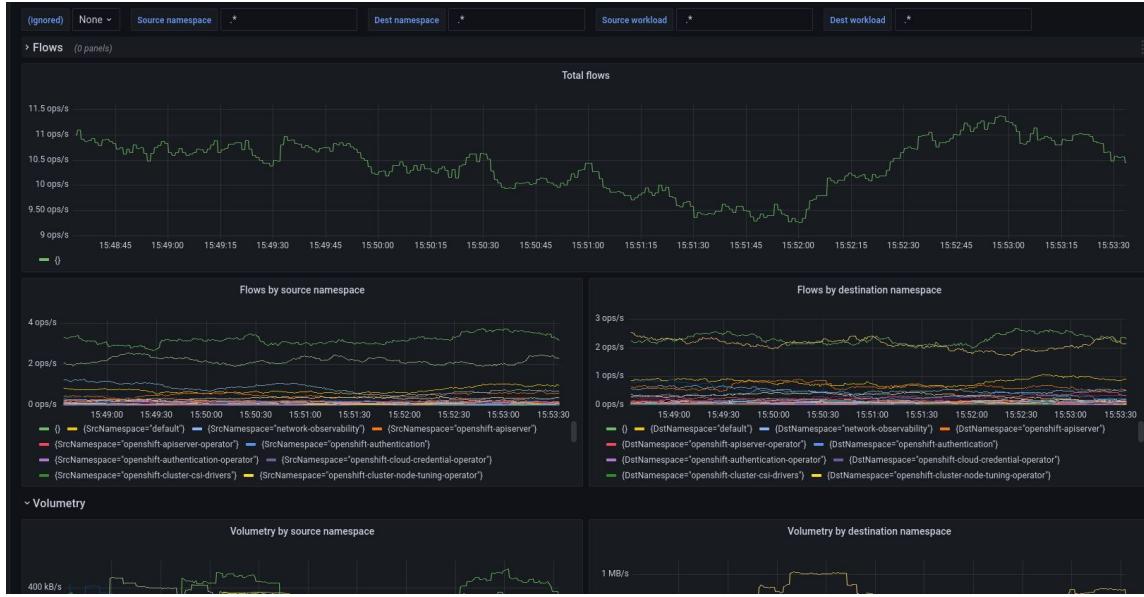
Common Namespace ansible Clear all filters

Date & time	Source	Destination	Bytes	Packets			
Name	Namespace	Name	Namespace	Port			
Fri Jun 10 2022 15:05:23	aap-controller-postgres-0	aap-controller-77d8cbf8cd-cz6tj	ansible	47906	24000	400	
Fri Jun 10 2022 15:05:03	aap-controller-postgres-0	aap-controller-77d8cbf8cd-cz6tj	ansible	46746	372800	400	
Fri Jun 10 2022 15:05:03	aap-controller-postgres-0	aap-controller-77d8cbf8cd-cz6tj	ansible	5432	46740	179600	400
Fri Jun 10 2022 15:04:23	aap-controller-postgres-0	aap-controller-77d8cbf8cd-cz6tj	ansible	5432	44540	24000	400
Fri Jun 10 2022 15:03:23	aap-controller-postgres-0	aap-controller-77d8cbf8cd-cz6tj	ansible	5432	41192	20800	400

44 flows 3 MB 18800 packets 9 Kbps

root@fdavalos:~ fdavalos@fdavalos:~ hit@bm1:~/install-dir *Document 1 sans titre - gedit keepass.kdbx [verrouillé] - KeePass... ILO: bay2.gva.synergy.hybridit.h... Red Hat OpenShift Container Plat... 2 / 5

NetFlow Graphs in OCP 4.10

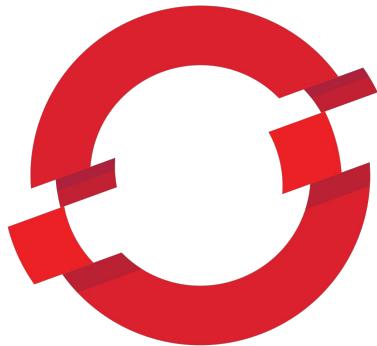


Features

- Leverages Grafana
- Create your own graphs
- Provides dashboards

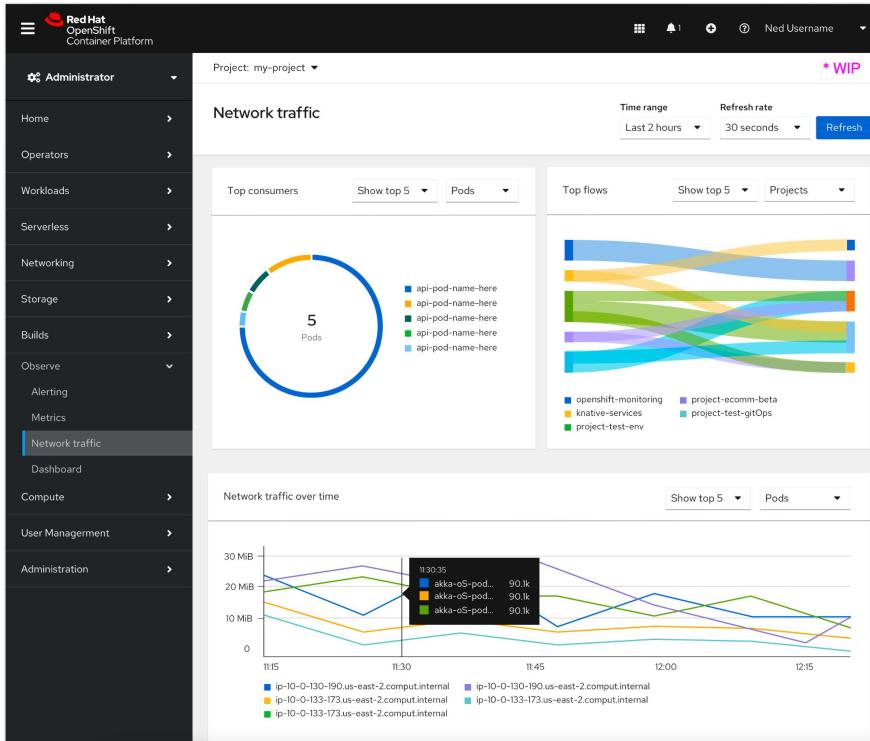
Network Observability in OCP 4.11 and Beyond

(Tech Preview)



- Retrieve NetFlows using eBPF
- Topology
- Dashboards on top pod talkers, layer 4 services, ...
- Apache Kafka for scalability
- Productization
- Persistent storage

OCP 4.11 Dashboard



Features

- Top pod talkers
- Graphical pod-to-pod communication
- Network traffic over time



OVN Kubernetes

Open Virtual Network (OVN)

Next-Gen Default OpenShift SDN

- An implementation of virtual networking via Open vSwitch project
- Developer Community
 - SDN portfolio consolidation / common network tech (RH-OCP, RH-OSP, RHV)
- Acceleration and enablement of customer-driven feature requirements
 - Egress IP per pod
 - Distributed Ingress/Egress firewall
 - Distributed services LB
 - Multi-Network/Interface
 - Heterogeneous clusters w/ Windows nodes
 - Capability to span on-prem & cloud nodes
 - Traffic isolation / Multi-tenancy
 - DPDK support
 - Encrypted tunnels
 - IPv6 / DHCPv6
 - QoS, Control/Data plane separation
 - ...

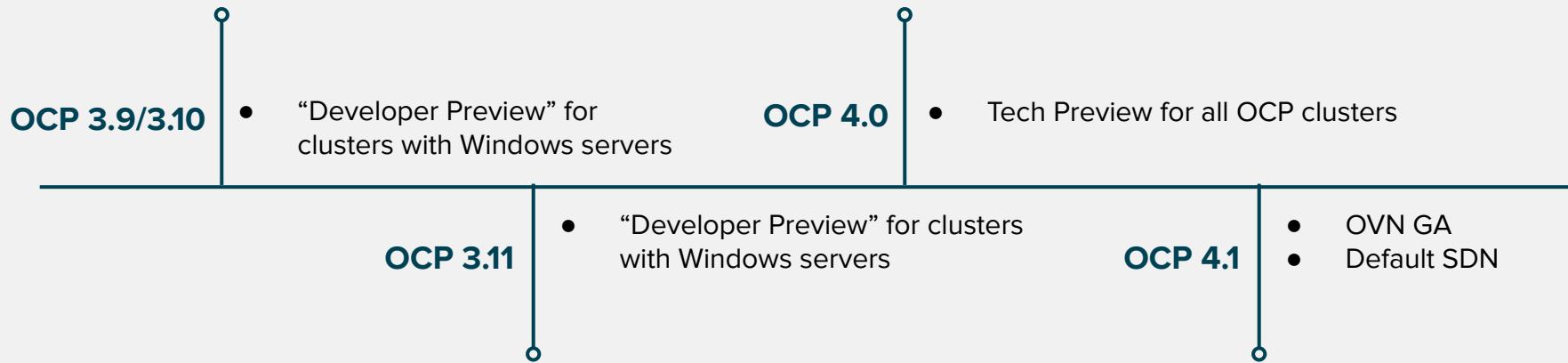
Advantages of OVN Kubernetes

- The core OVN project is also used by OpenStack and RHEV
 - We get their bugfixes, performance improvements, and features
- Red Hat has a team working on OVN who understands it deeply
- Other companies are also using (or planning to use) OVN-Kubernetes
 - The project was started by NVIDIA and VMware; Red Hat joined later
- Effectively, our team is much larger now, giving us more ability to implement new features like: IPv6, hybrid Linux/Windows clusters, more powerful NetworkPolicy support, etc

Hey... Wait a minute...

<u>OpenShift SDN</u>	<u>OVN Kubernetes</u>
veth pairs	veth pairs
OVS bridge	OVS bridge
Central controller / host-ipam	Central controller / host-ipam
VXLAN tunnels	Geneve tunnels
OVS flows for NetworkPolicy	OVS flows for NetworkPolicy
IPTables for Services	OVN LBs for Services
IPTables for NAT	OVS for NAT

OVN Enablement Timeline

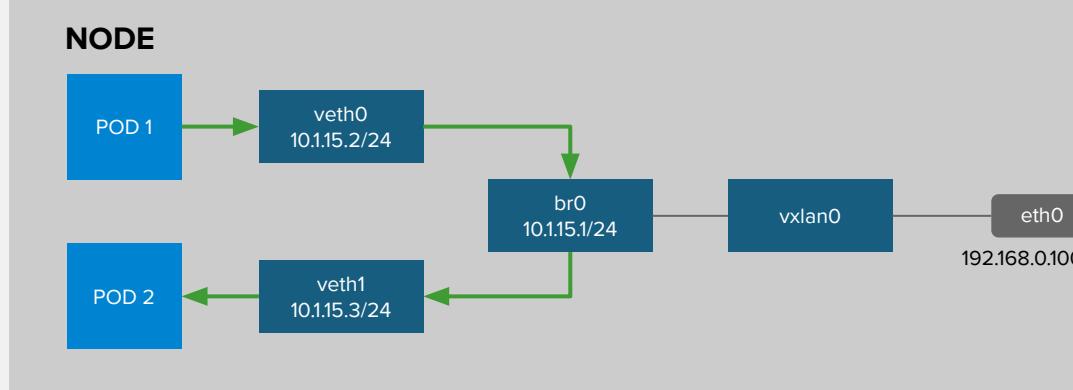




Openshift SDN Flows

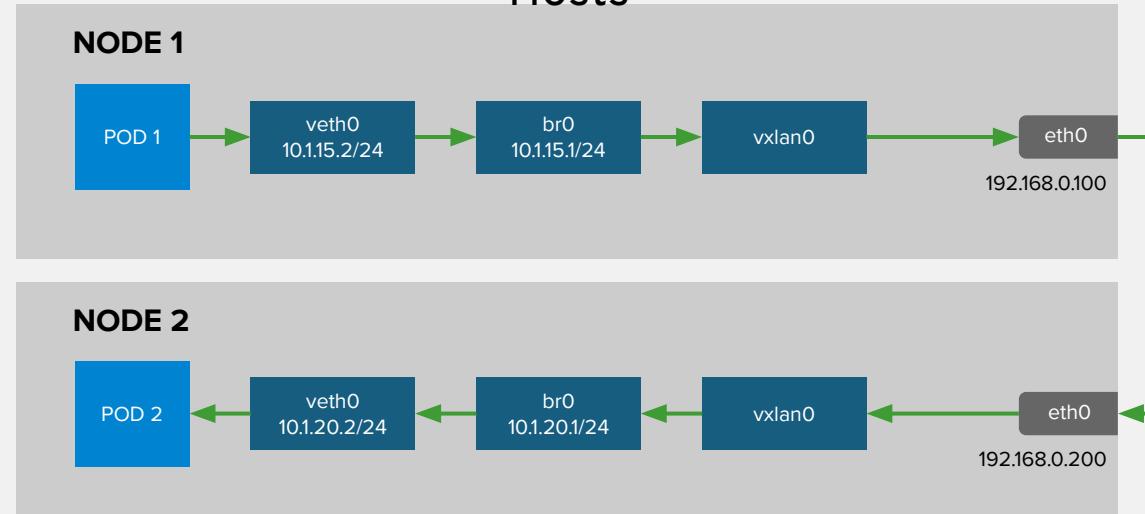
OPENShift SDN - OVS PACKET FLOW

Container to Container on the Same Host



OPENShift SDN - OVS PACKET FLOW

Container to Container on the Different Hosts



OPENShift SDN - OVS PACKET FLOW

Container Connects to External Host

