

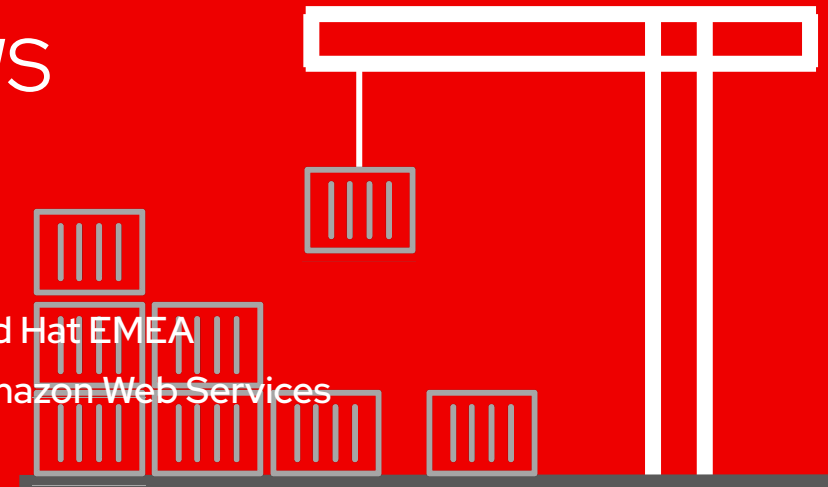
Deploying a Modern Application Platform

with Red Hat OpenShift Service on AWS

Robert Åkerblom, Senior Black Belt, Managed OpenShift, Red Hat EMEA

John Johansson, Senior Cloud Success Architect, Managed OpenShift, Red Hat EMEA

Andreas Skarmutsos Lindh, Specialist Solutions Architect, Containers at Amazon Web Services



Who this Workshop is For.



- ▶ Platform engineers looking to build an application platform.
- ▶ Developers looking to understand foundations of an application platform.
- ▶ DevOps looking to Ops with their Dev.

Knowledge Prerequisites

Skills/Knowledge required to be successful in this workshop.



- ▶ Basic understanding of OpenShift or Kubernetes concepts.
- ▶ Knowledge of running workloads in a Cloud Provider environment.
- ▶ Basic CLI/Linux experience.

What you will learn today!

Day Two ROSA Operations:

- ▶ Configuring Cluster Authentication
- ▶ Managing Cluster Upgrades
- ▶ Managing Worker Nodes
- ▶ Cluster Autoscaling
- ▶ Labeling Nodes
- ▶ Logging with AWS CloudWatch

Deploy and Expose an APP:

- ▶ Deploy the App
- ▶ Make an App Resilient
- ▶ Restrict Network Access
- ▶ Using OpenShift GitOps
- ▶ Automate Deploying the App with Openshift Pipelines

What you will learn today!

Service Mesh:

- ▶ Introduction to OpenShift Service Mesh
- ▶ Deploy Service Mesh Operator
- ▶ Deploy Control Plane
- ▶ Deploy Workloads
- ▶ Configure and Observe Traffic

Accessing the Workshop

Access the Workshop

Registration Page:

- ▶ <https://red.ht/rosa-sto>

Password:

- ▶ rosa

WIFI:

- ▶ Guest
- ▶ BrokenWires@@2019

Notes: <https://red.ht/rosa-sto-notes>

Building a Modern Application Platform Workshop

Access to Building a Modern Application Platform Workshop

Email * ?

Workshop Password * ?

Access this workshop →

Building a Modern Application Platform Workshop

Instructions for Building a Modern Application Platform Workshop

Lab User Interface <https://bookbag-cpvn6-bookbag.apps.shared-410.openshift.redhatworkshops.io/> 

Messages Lab instructions: <https://bookbag-cpvn6-bookbag.apps.shared-410.openshift.redhatworkshops.io/>

Data

- aws_access_key_id: AKIA52VPS74UF5NFKPNB
- aws_default_region: us-east-2
- aws_route53_domain: .sandbox2424.opentlc.com
- aws_secret_access_key: Ajjf3WE8dpR8eHXbTcZ41+UMnEMG7WB+mQ7kAv5b
- aws_web_console_password: EI-sJh3nfaOo3
- aws_web_console_url: <https://950629760808.signin.aws.amazon.com/console>

WORKSHOP MODULES

Home

Environment Setup
Deploy a ROSA cluster
Create an Admin User
Configure Cluster Authentication Using Amazon Cognito
Managing Cluster Upgrades
Managing Worker Nodes
Cluster Autoscaling
Labeling Nodes
Log Forwarding to AWS Cloudwatch
Deploy an Application
Making your Application Resilient
Securing the application using NetworkPolicies
Deploy the Application using OpenShift Gitops
Automate Deploying the App using OpenShift Pipelines
Introduction to OpenShift Service Mesh
Deploy Service Mesh Operator
Deploy Service Mesh Control Plane
Deploy Microservices

Home



Welcome to the Building a Modern Application Platform workshop. In this workshop you will learn the building blocks of modern application platform and leverage Amazon Web Services (AWS) and Red Hat OpenShift Service on AWS (ROSA) to build a modern application platform.

Who this workshop is for: This workshop is aimed at Platform Engineers, DevOps Engineers, CCloud Operations, Architects, and Developers that want to learn what makes a modern application platform, and how they can leverage cloud services to streamline the delivery and operations of their application platforms.

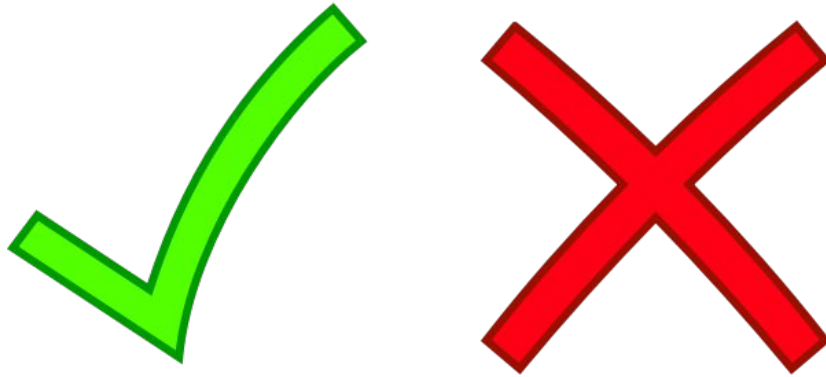
What to expect: During the workshop, we will take you through a series of hands on exercises to help you understand some of the concepts of modern application platforms. Attendees will learn:

- How to deploy a Red Hat OpenShift Service on AWS (ROSA) cluster
- Complete Day 2 operations tasks including: configuring node and cluster scaling policies, configuring managed upgrades, configuring single-sign-on for the cluster using Amazon Cognito, and forwarding logs to Amazon CloudWatch.
- Deploy an application that uses AWS IAM Roles for Service Accounts and AWS STS to connect to an Amazon DynamoDB table.
- Make an application on OpenShift scalable and resistant to node failures and upgrades
- Deploy an application using CI/CD tooling, including OpenShift GitOps and Source-to-Image, and use labels for deterministic app placement on nodes

Terminal

[~] \$

Workshop Guidelines



- ▶ Be respectful of facilitators, participants, and the compute environments provided.
- ▶ Raise your hand or find a facilitator if you need help, have a question, or get stuck.
- ▶ Let us know how we did, positive or constructive criticism is welcome!

Red Hat OpenShift Cloud Services

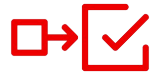
Red Hat OpenShift cloud services

A turnkey application platform with management and support from Red Hat and leading cloud providers



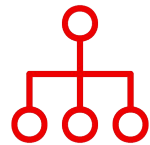
Accelerate time to value

Quickly build, deploy, and run applications that scale as needed.



Operational efficiency

Enhance operational consistency, efficiency and security with proactive management and support.



Focus on innovation

Simplify operations so your teams can refocus on innovation, not managing infrastructure.



Hybrid cloud flexibility

Deliver a consistent experience on premises and in the cloud.

Build business value, not a technology platform

DIY Kubernetes-powered platform



Time and resources to integrate
10-20+ individual services

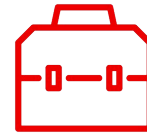


In house or multiple vendors for
ongoing maintenance and support

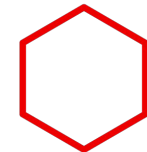


Reliance on one cloud or DIY
integration of multi cloud apps

Red Hat OpenShift cloud services



Turnkey application platform with
integrated services and tools



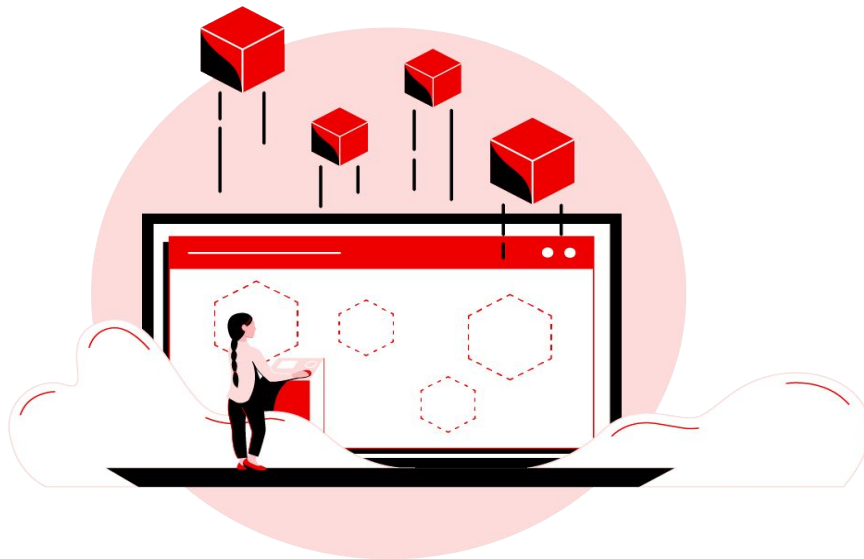
Managed Kubernetes AND
components to reduce complexity



Consistent hybrid
cloud experience and cloud choice

What is an Application Platform?

Used for building, deploying, and running applications through a simple, but flexible experience. An application platform includes the components: runtimes, build tools, CI/CD, and observability (including application logging) and abstracts away technical details such as containers and Kubernetes.



Build, Test, Deploy

Apply the heart of DevSecOps policy & procedure on a consistent infrastructure foundation.

Run and Manage

with consistency and unified security.

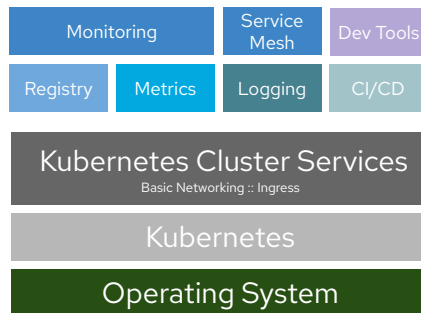
Design & Code

using cloud-native dev tools & application technology while benefiting from DevSecOps right at the start.

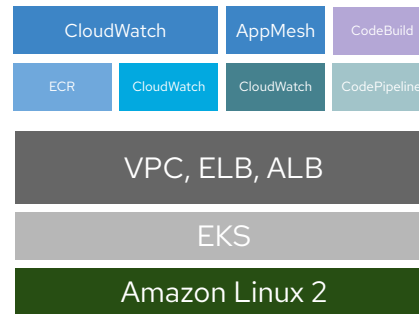
Building an Application Platform on Public Clouds



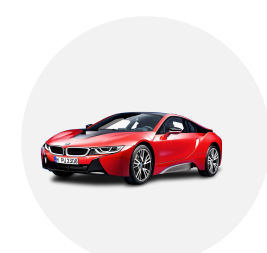
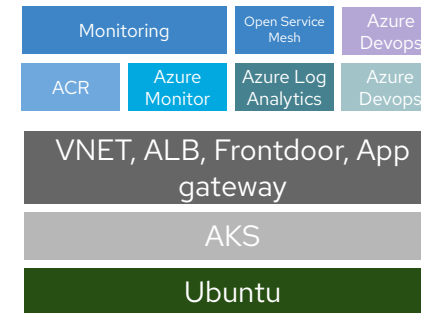
The required Parts



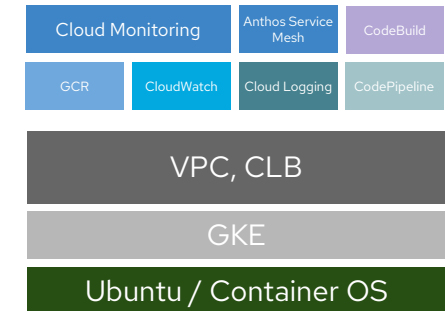
The AWS Car



The Azure Car



The GCP Car



3 Different Cars

- Different component versions
- Different life cycles
- Different support models
- Different developer and ops tooling

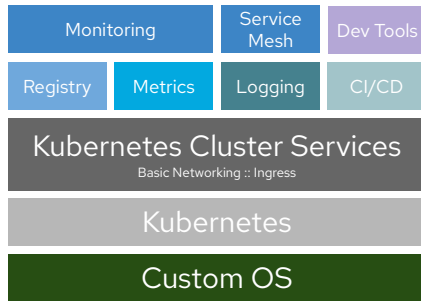


3 different drivers and pit crews needed

Build and run a platform *versus* using a turnkey cloud service



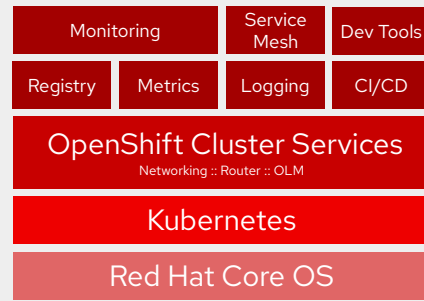
The Parts



xKS + 'native'
services



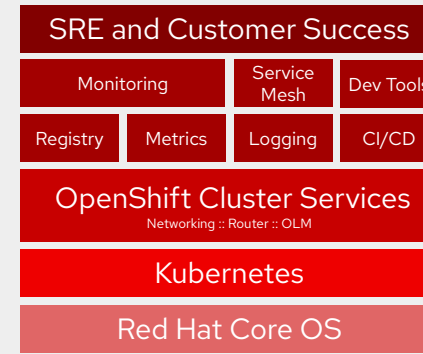
The Assembled Car



- Application Platform -
Self-managed Red Hat OpenShift



The Car & Pit Crew



- Turnkey Application Platform -
Red Hat OpenShift cloud services

"Batteries Included"

... but swappable

Individual components can be swapped out

Eg.

- Using AWS CloudWatch for logging on AWS
- Use specific cloud services or ISV offerings

An opinionated platform for building, deploying and running applications



Service Mesh

App-Services

DB-Services

CI/CD

DNS

Authentication

Monitoring

Kubernetes

Automation

Logging

Registry

Security

Compute

Storage

Network

- ▶ Fully integrated and supported components
- ▶ Expert SRE and Customer Success support
- ▶ Abstracts away technical details
- ▶ Consistent experience across clouds

Move from 24x7 operations to 9-5 innovation

End-to-End support for your entire application platform



- ▶ OpenShift cloud services includes full support for worker nodes
 - Zero downtime upgrades,
 - proactive monitoring
 - automated patching
 - Compliance and certifications extend to worker nodes
- ▶ 99.95% financially backed SLA
- ▶ 24x7 joint support from Red Hat and cloud provider
- ▶ Automation and Day 2 Operations by global SREs

Full Stack management from a global Site Reliability Engineering (SRE) team



Product



Systems

Who are our SREs?

- ▶ Developers and systems engineers who know how to handle volume and a diversity of clusters.
- ▶ Uniquely offer both an engineering and development mindset.

SRE responsibilities

- ▶ Build solutions and features
- ▶ Automate at scale
- ▶ Day 2 operations: monitor, patch, and upgrade the platform.
- ▶ Work hand in hand with cloud providers and open source community.

Customer benefits

- ▶ Accelerate application development and delivery through automation.
- ▶ Improve operational efficiency, security and resiliency.
- ▶ Refocus on innovation and core competencies.

Complexity of running your own Kubernetes Cluster

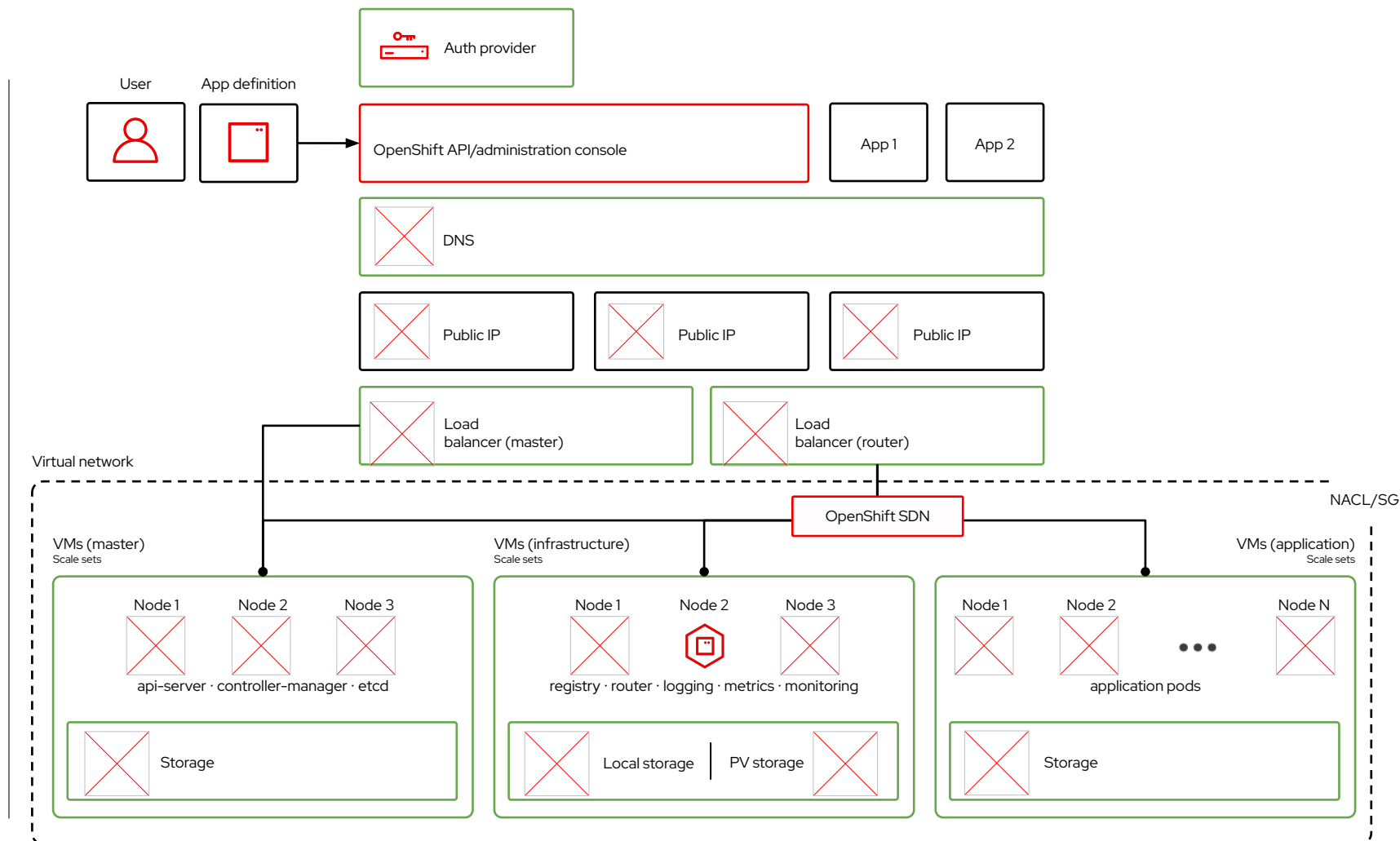
Responsibilities	
User management	■
Project and quota management	■
Application life cycle	■
Cluster creation	■
Cluster management	■
Monitoring and logging	■
Network configuration	■
Software and security updates	■
Platform support	■



Customer



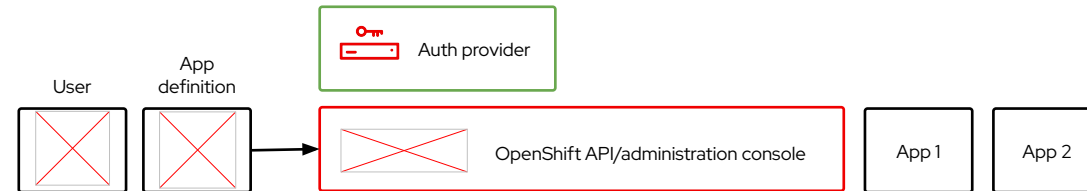
Cloud provider & Red Hat



Simplify with fully managed clusters

Red Hat OpenShift cloud services

Responsibilities	
User management	Customer
Project and quota management	Customer
Application life cycle	Customer
Cluster creation	Cloud provider and Red Hat
Cluster management	Cloud provider and Red Hat
Monitoring and logging	Cloud provider and Red Hat
Network configuration	Cloud provider and Red Hat
Software and security updates	Cloud provider and Red Hat
Platform support	Cloud provider and Red Hat



Let Red Hat & your cloud provider...

Manage all your clusters.

Secure your nodes.

Monitor and operate your VMs.

Manage environment patches.

You...

Improve focus, efficiency and productivity

Forrester Research: The Total Economic Impact™ of OpenShift cloud services



50%

50% improvement in operational efficiency¹

35%

35% increase in developer productivity¹

65%

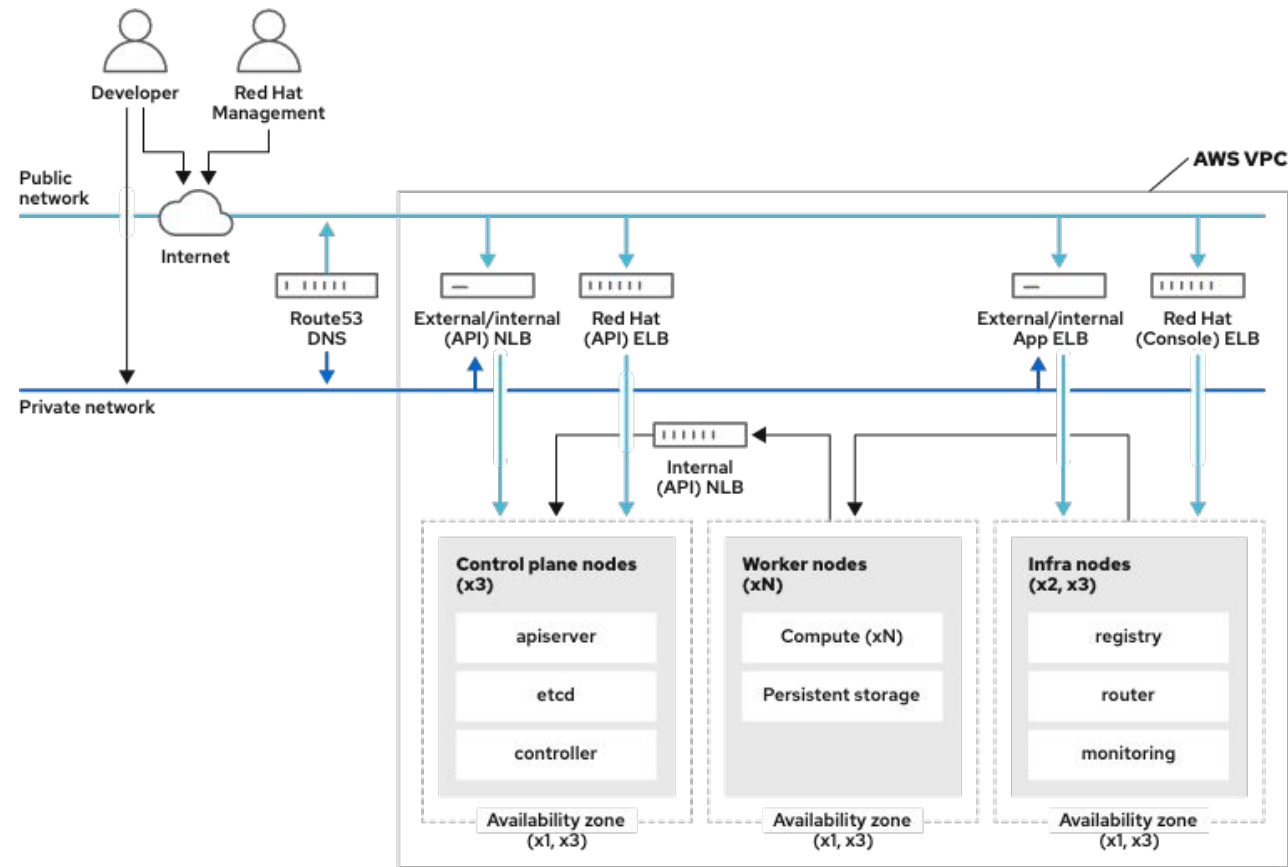
Shortened development cycle by 65%¹

“One of our pain points is we don’t want to do infrastructure. We just want to **focus on building great experiences**. We wanted to find somebody who could **manage this for us**, so we didn’t have to.”

Director for operations and infrastructure,
Telecom company

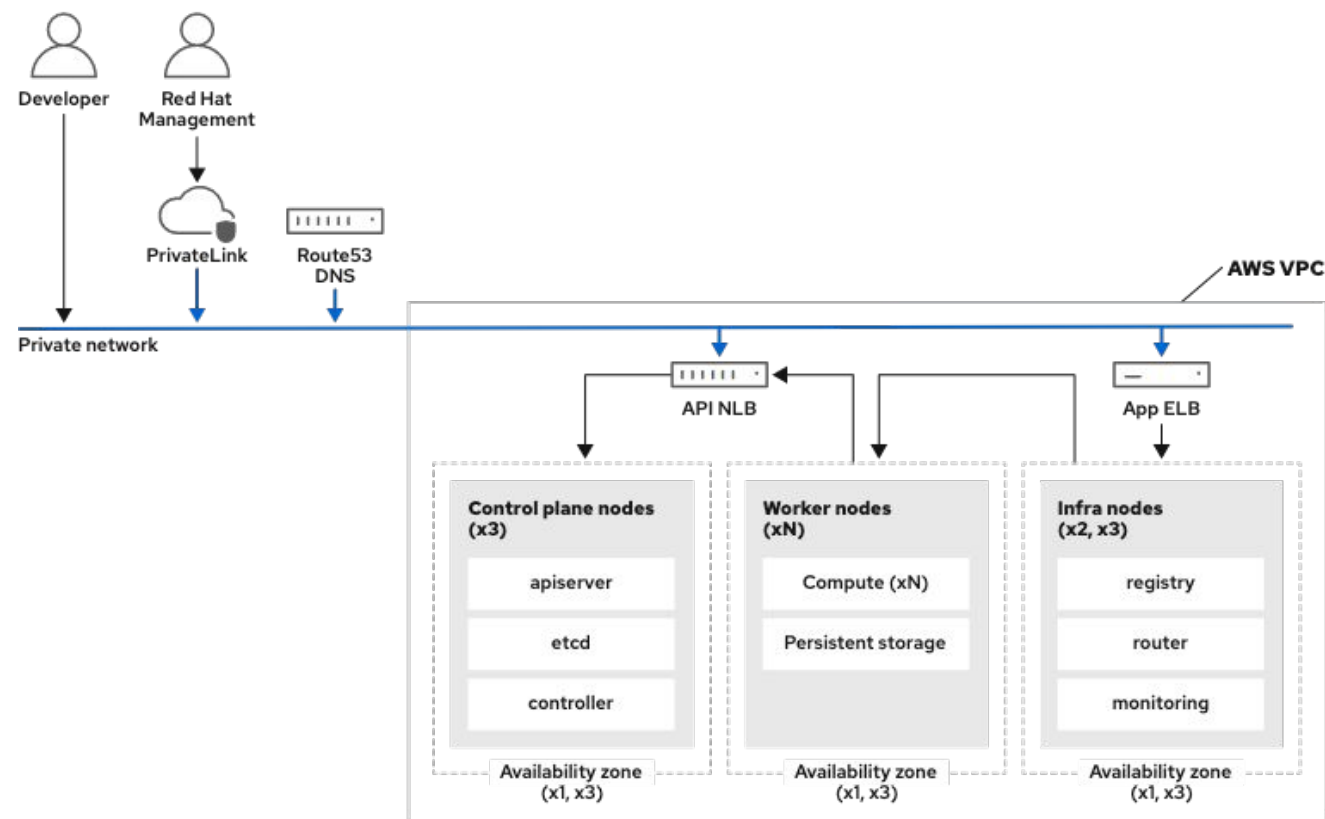
Red Hat OpenShift on AWS Architecture

Public / Private Networking



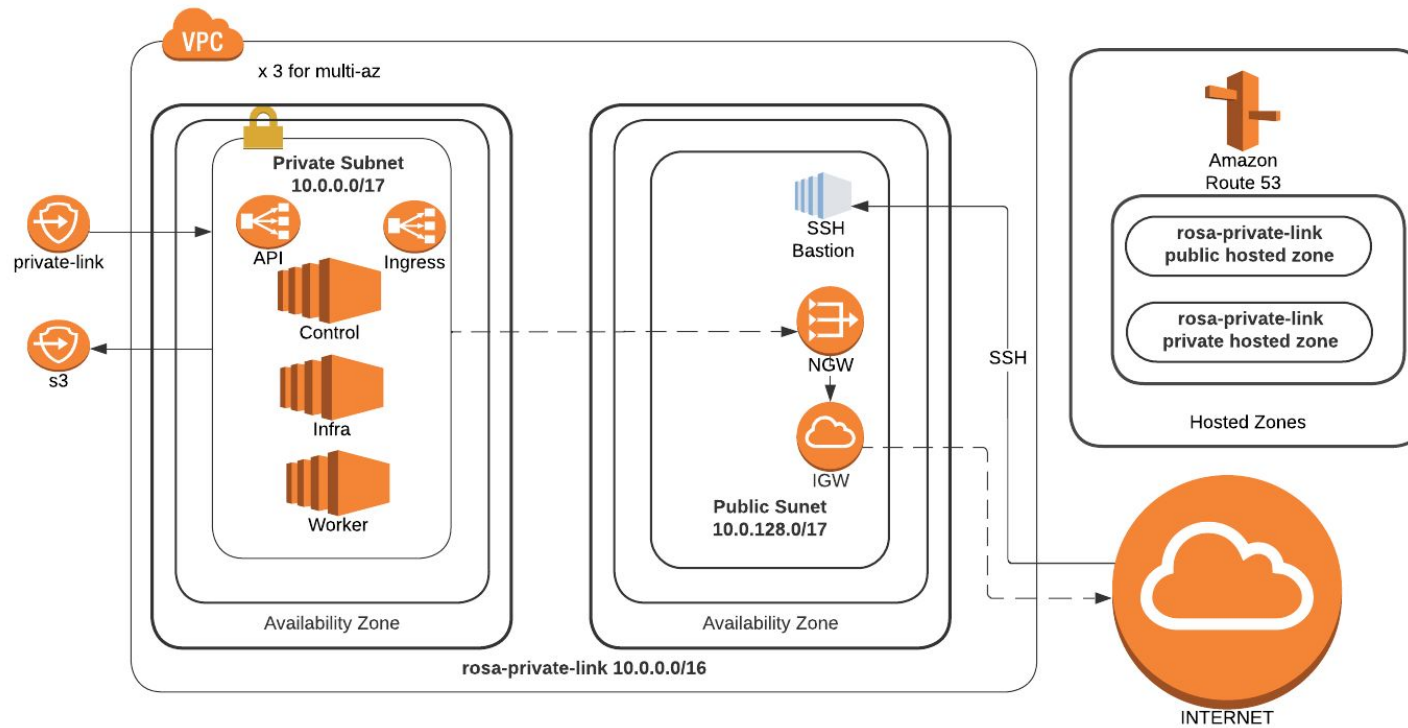
156_OpenShift_0521

Private Link Networking

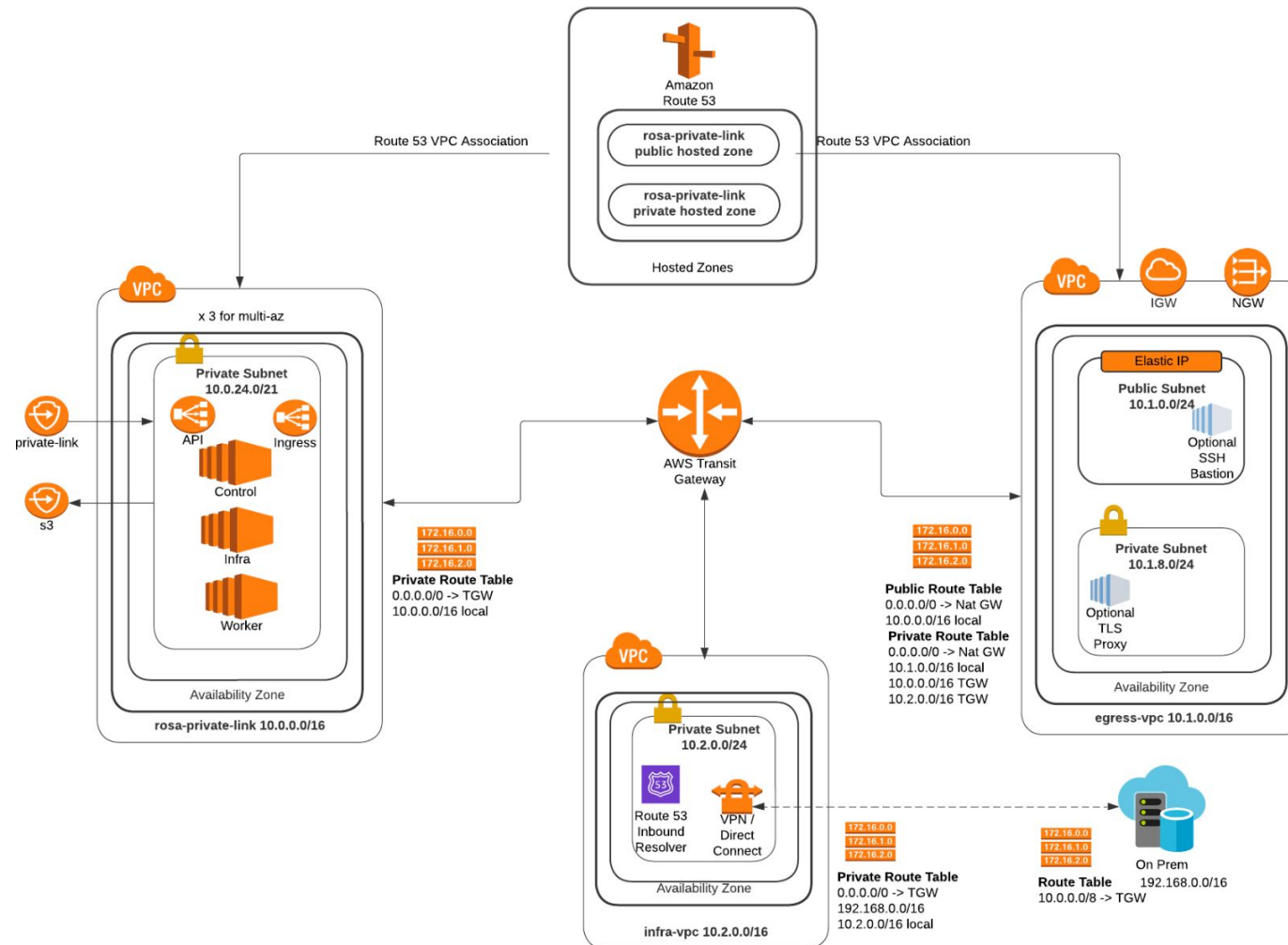


ROSA_OpenShift_122

Private Link Networking



Private Link Networking (Transit Gateway)



ROSA Private-Link - TGW

pczarkow | May 2, 2022

Extra: ROSA Hosted Control Planes GA as of yesterday!

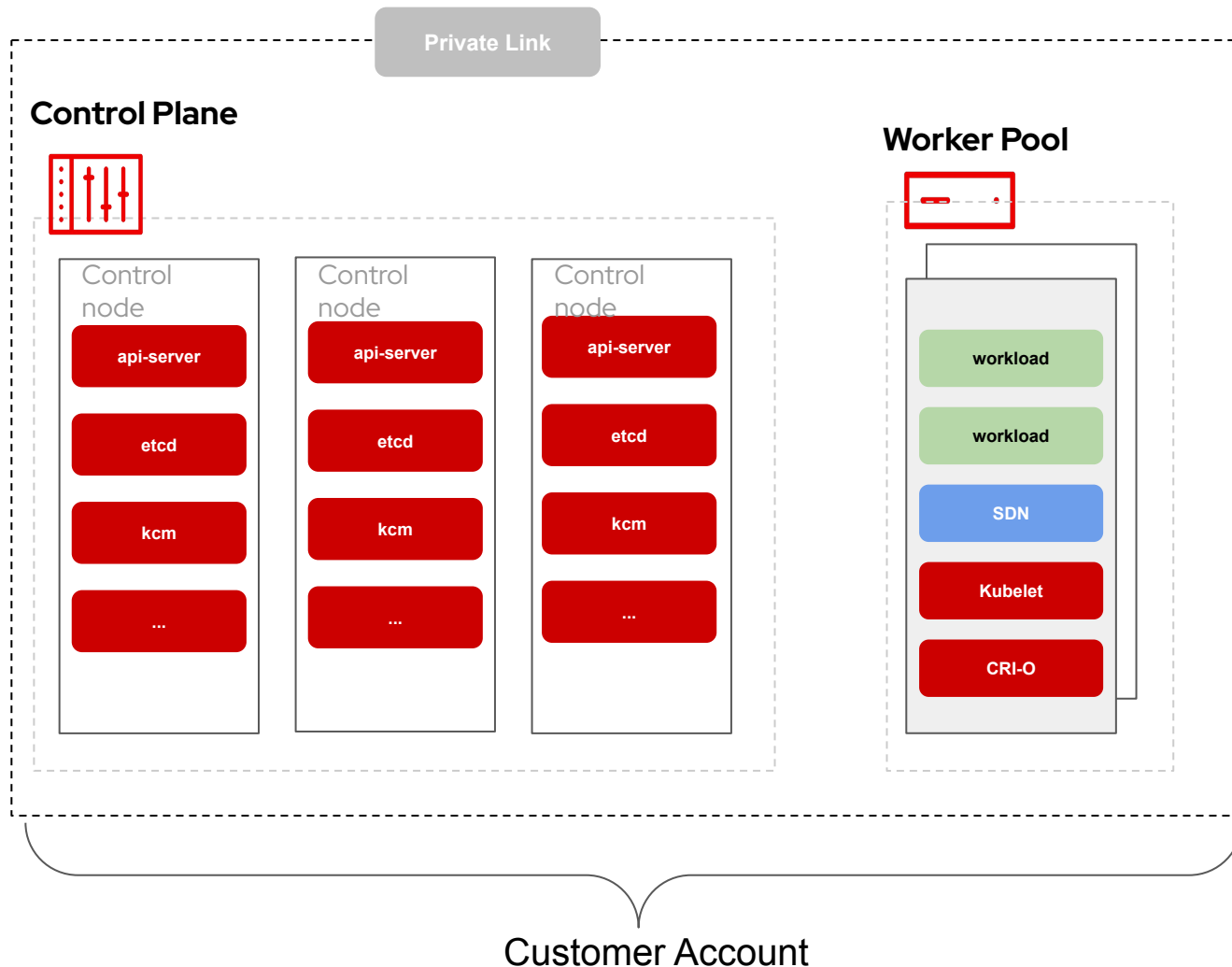
What is ROSA with hosted control planes?

- ▶ New **deployment model** for ROSA in which the control plane is hosted in a Red Hat owned AWS account.
 - Control plane no longer hosted in customer's AWS account
 - The control plane is dedicated to a single cluster
 - Provides a highly available control plane that is isolated within Red Hat's AWS account
- ▶ Why? Reduce costs & improve efficiency and reliability.
 - OpenShift with Hosted Control Planes is *designed* to be "managed"
 - Reduced chance of accidental misconfiguration or deletion of resources

ROSA with hosted control planes Benefits

- **Cost Savings**
 - Customers reduce costs by 5x on average vs hosting the control plane in their own account
 - Significantly reduced AWS infrastructure costs (typically \$8k / cluster / year)
 - Quickly and easily spin up or tear down clusters when needed for efficiency and cost savings
 - More flexibility and portability for annual billing allowing customers to easily change between node types
 - Smaller overall footprint (2 nodes vs 7)
 - Scale worker nodes to 0 (post GA)
- **Operational efficiency**
 - Provisioning time ~ 10 minutes for a new cluster – get started and build/deploy apps faster
 - Seamless autoscaling of control plane at no additional cost
 - Installer runs in ROSA Service account reducing required permissions
 - Designed to be managed; taking what we learned from operating OpenShift at scale, making improvements and putting it into the core product out of the box for a better experience
- **Increased reliability**
 - Control plane is always HA over multiple availability zones
 - Selectively upgrade control plane and worker nodes separately, giving increased control and flexibility for customers
 - Increased resiliency from offloading control plane infra management, reducing the chance of accidental misconfiguration or deletion of resources

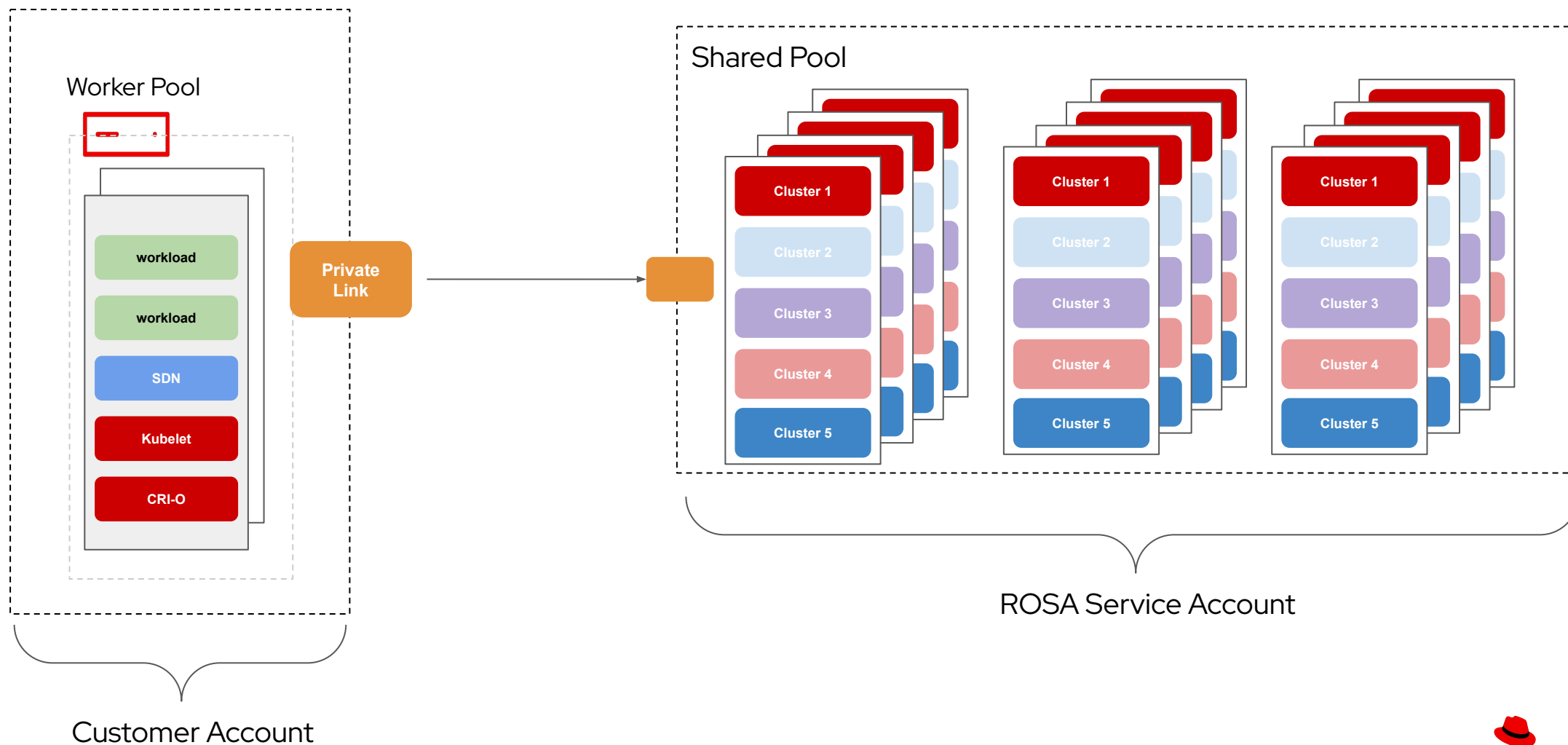
Classic OpenShift Architecture



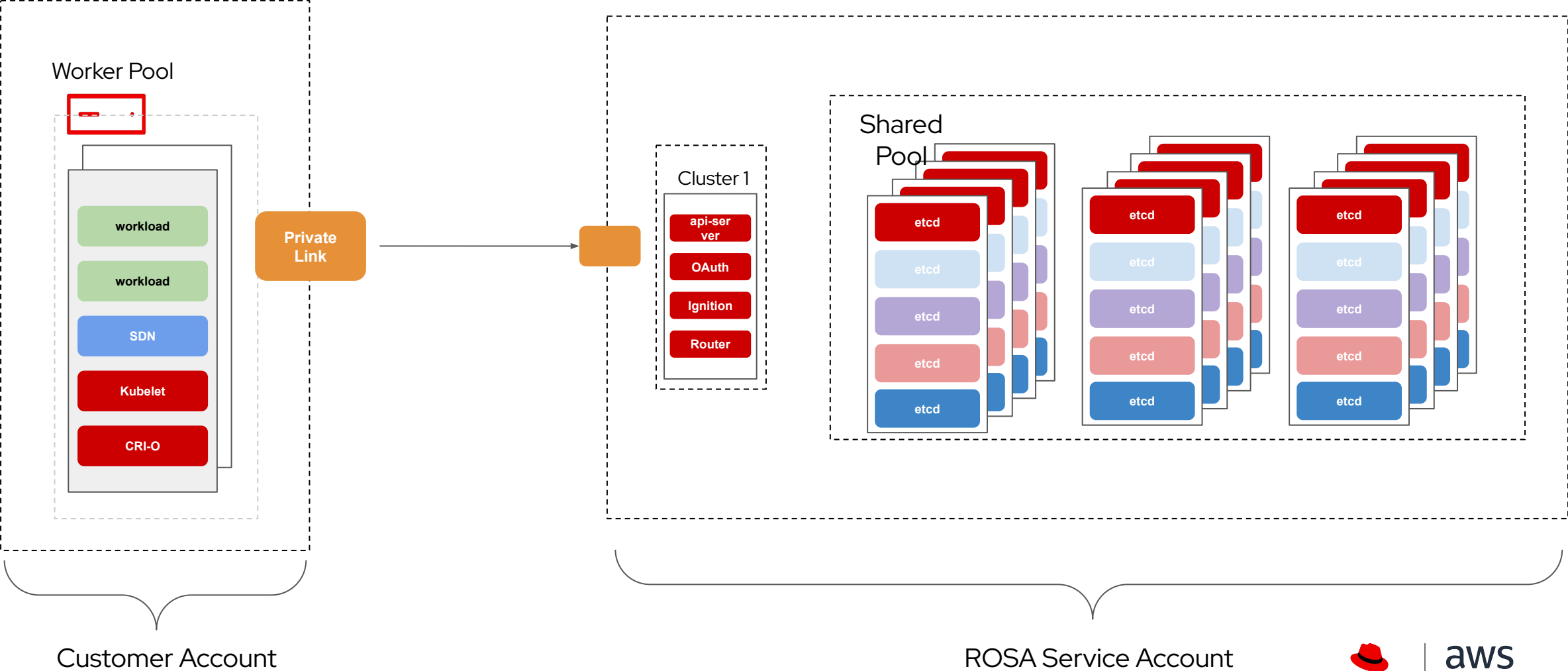
Private Link

- ▶ Cluster API Server published using AWS Private Link
- ▶ API Server is accessed by Red Hat management system (HIVE) to allow programmatic access to API server
- ▶ Private traffic passes through private network not through the Internet

Hosted Control Plane Architecture (Shared Pool)



Hosted Control Plane Architecture



ROSA with HCP

- ▶ Control plane components run in Red Hat's AWS account
- ▶ Control plane components are exposed to worker nodes through AWS PrivateLink
- ▶ Worker nodes communicate with control plane over PrivateLink connection
- ▶ Red Hat SRE management traffic takes place within Red Hat's AWS account
- ▶ Red Hat network access to customer VPC is minimized

vs

ROSA Classic

- ▶ Control plane components run in customer's AWS account
- ▶ Control plane components are exposed to Red Hat management traffic through AWS PrivateLink
- ▶ Worker nodes communicate directly with control plane nodes within same VPC

ROSA with hosted control planes vs ROSA "Classic"

	Hosted Control Plane	Classic
What is it?	Control plane components (e.g., etcd, API server, oauth) are hosted on AWS in a Red Hat owned and managed OpenShift cluster	Control plane, infra & worker nodes all live in customer's AWS account
Provisioning Time	~10 minutes	~40 minutes
Architecture	<ul style="list-style-type: none">Underlying control plane infrastructure is fully managed and directly unavailable to end customers except through dedicated and explicitly exposed endpoints	<ul style="list-style-type: none">Customers are responsible for control plane, infra and networkingAll-in-one OpenShift on AWS infrastructure architecture
Footprint	1 cluster = minimum 2 worker nodes	1 cluster = minimum 7 nodes (3 control plane, 2 infra, 2 worker nodes)
Upgrades	Selectively upgrade control plane and worker nodes separately	Entire cluster is upgraded at one time

ROSA with hosted control planes vs ROSA "Classic"

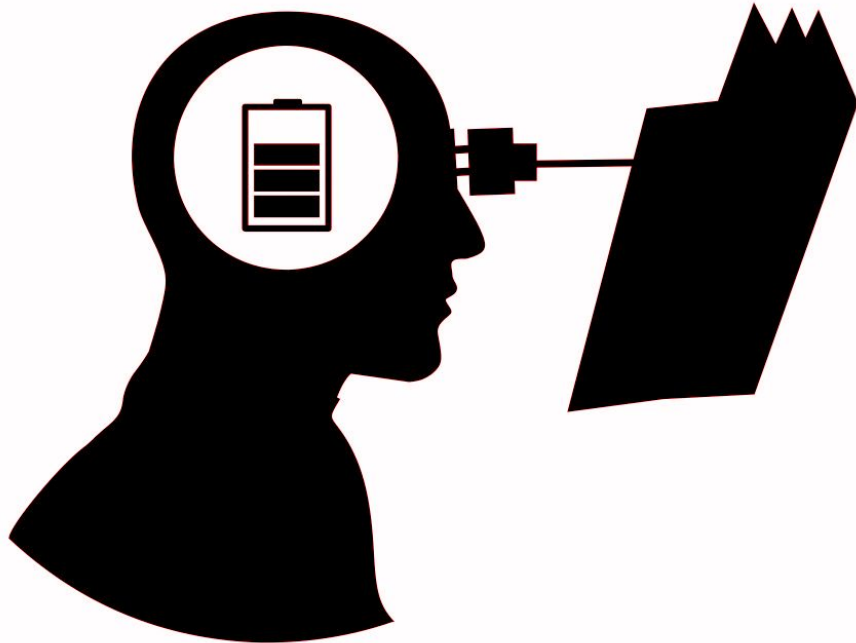
	Hosted Control Plane	Classic
Deployment	<ul style="list-style-type: none">• Deploy using ROSA CLI (web UI coming soon)• Customers provision "Hosted Clusters" that deploy the control plane components into Red Hat's Management clusters• Customers request "Machine Pools" that deploy worker nodes into the customer's AWS account	<ul style="list-style-type: none">• Deploy using ROSA CLI or web UI• Full cluster provisioning occurs in customer's AWS account
Regional Availability	Initially 6 regions available us-east-1, us-east-2, us-west-2, eu-west-1, eu-central-1, ap-southeast-3	Available for purchase in all countries where AWS is commercially available
Compliance	No compliance certifications or FIPS at GA	ISO 27001, 17, 18; SOC 2 Type 2, SOC 3, PCI-DSS, HIPAA
Add-ons	No add-ons support at GA	RHOAM, RHODS

Section 1:

Day Two Operations

Day Two Operations

What you'll learn today.



- ▶ Integrating with Amazon Cognito for IDP
- ▶ Managing Cluster Upgrades
- ▶ Managing Worker Nodes
- ▶ Cluster Autoscaling
- ▶ Labeling Nodes
- ▶ Logging with AWS CloudWatch

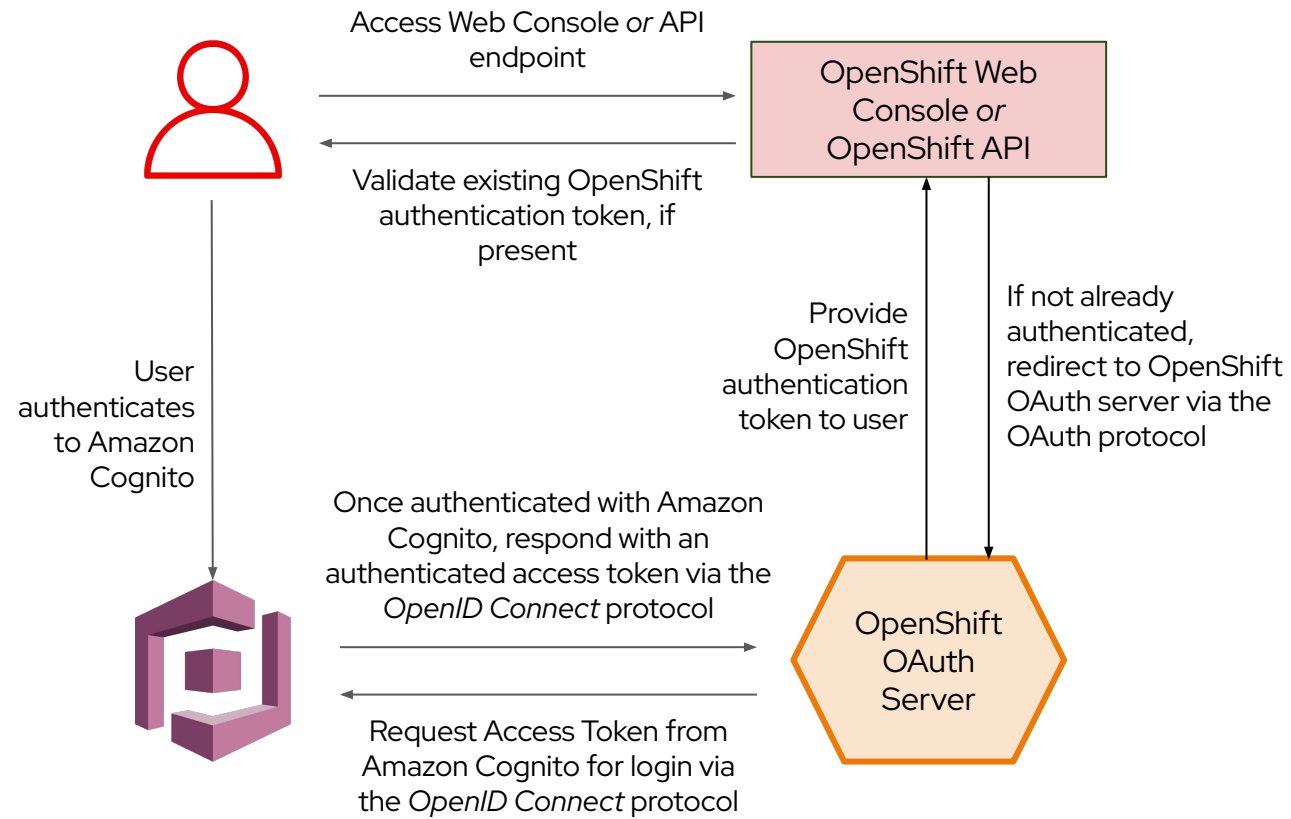
Day Two Operations

- ▶ **Integrating with Amazon Cognito for IDP** - In this module you will learn how to configure ROSA to authenticate against an OpenID Identity Provider such as Amazon Cognito.
- ▶ **Managing Cluster Upgrades** - ROSA makes cluster upgrades easy. Manage cluster upgrades - automatically or manually - for major, minor, or patch updates.
- ▶ **Managing Worker Nodes** - In this module you'll learn about MachinePools, and how to use them to manage the desired state of worker nodes.
- ▶ **Cluster Autoscaling** - In this module we'll cover how to configure a cluster to automatically scale based on the requirements of running (or requested) pods.
- ▶ **Labeling Nodes** - Labeling nodes allows for a number of use cases. In this module we will show how to schedule workloads on specific nodes which can be useful to match the application to required hardware (CPU, Memory, GPU).
- ▶ **Logging with AWS CloudWatch** - By default cluster logs are stored within the ROSA cluster. This module will show you how to ship logs off the cluster and into your preferred logging destination such as AWS CloudWatch.

Integrating with IDPs

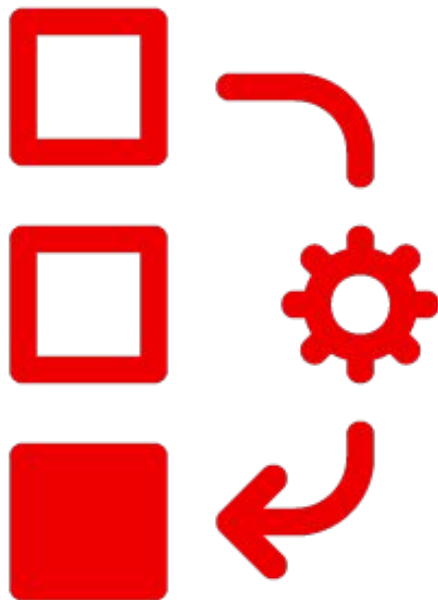


- ▶ ROSA supports a number of Identity Providers:
 - GitHub, GitHub Enterprise, GitLab, Google, LDAP, OpenID Connect.
- ▶ In this workshop, we'll use Amazon Cognito via the OpenID Connect integration.



Cluster Upgrades

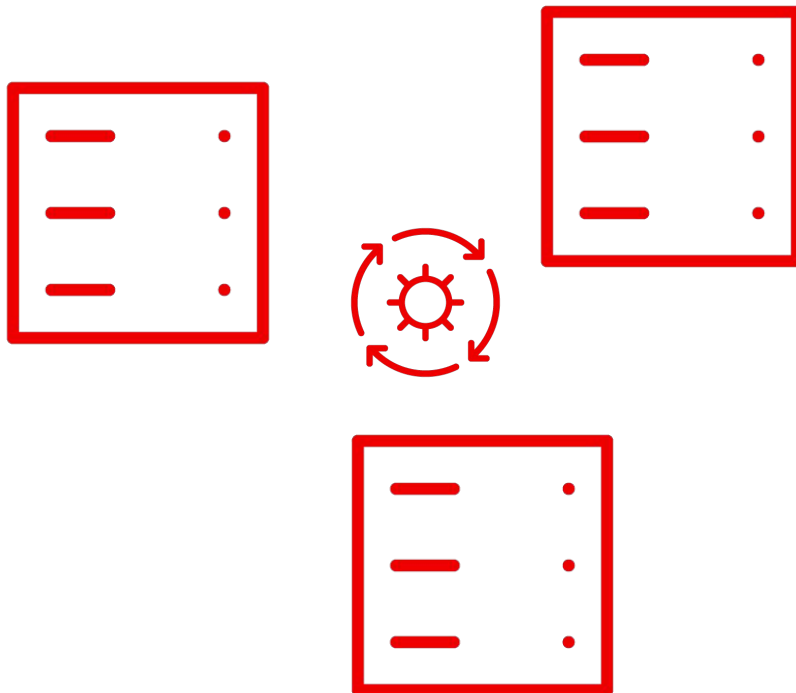
Major.Minor.Patch



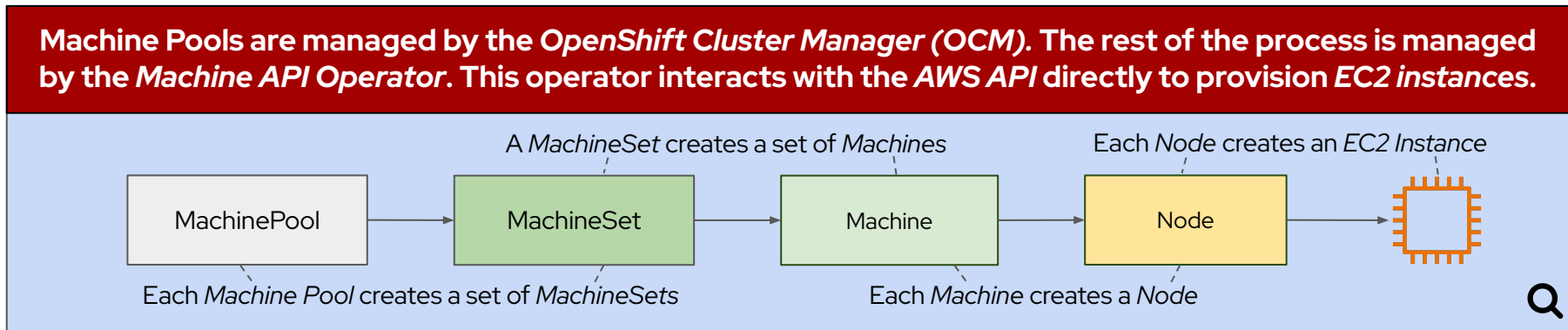
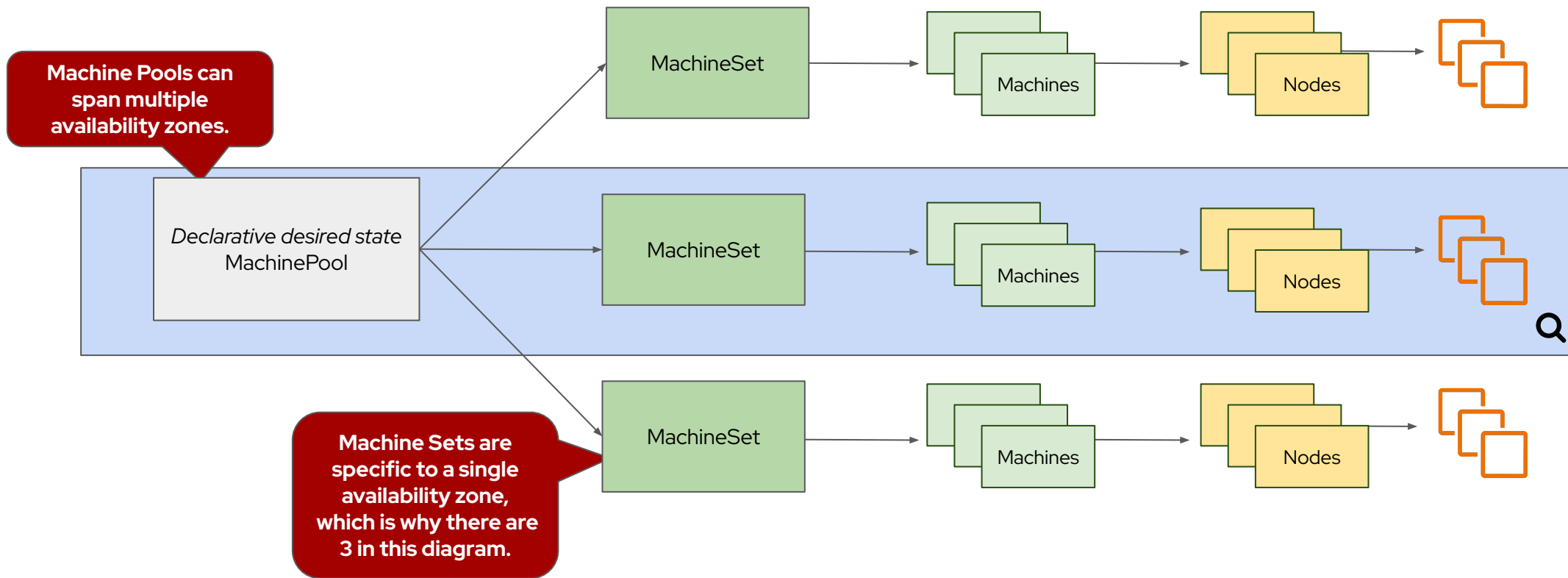
- ▶ Cluster upgrades can be **manually initiated** or **automatically scheduled**.
- ▶ **Critical CVEs** are **automatically patched** within **48 hours** of a Patch release.
 - Impacted Patch releases are deprecated and not supported.
- ▶ **Minor versions** are supported for **14 months**.
- ▶ **Major versions** are supported for **12 months following** the release of a **new major version**.

Managing Worker Nodes

Providing highly available compute.

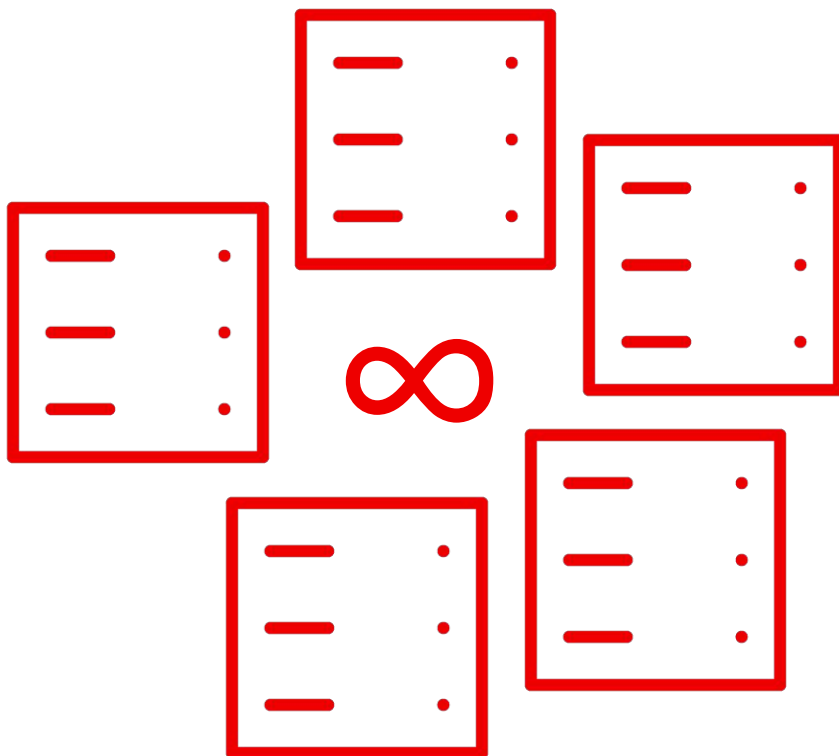


- ▶ MachinePools allows for worker nodes that span multiple availability zones (AZs).
- ▶ MachinePools provide a declarative desired state for worker nodes to ensure consistency across AZs.
- ▶ MachinePools can be scaled up or down manually or automatically.

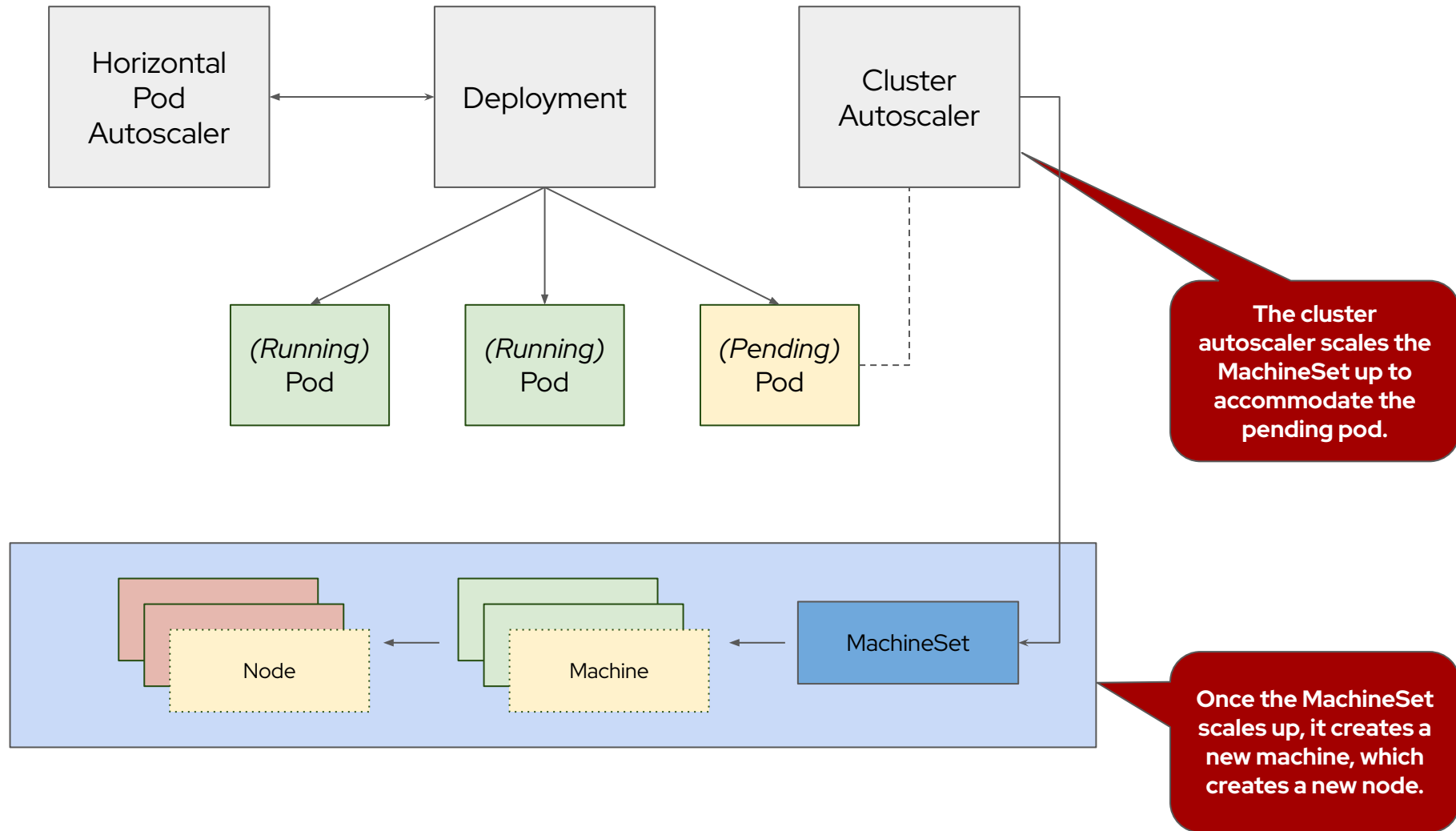


Cluster Autoscaling

Automatically responding to cluster demand.

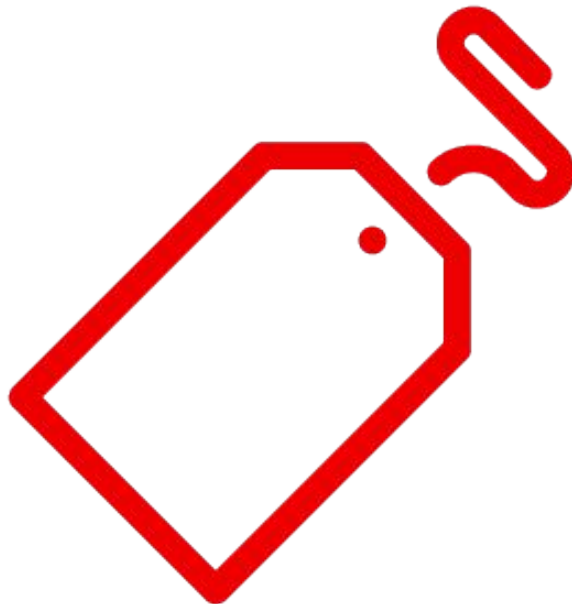


- ▶ MachinePools can be scaled to meet applications demands.
- ▶ Cluster AutoScaler will provision additional worker nodes when pods can not be scheduled due to resource constraints.
- ▶ Cluster AutoScaler will not scale beyond predefined limits.



Labeling Nodes

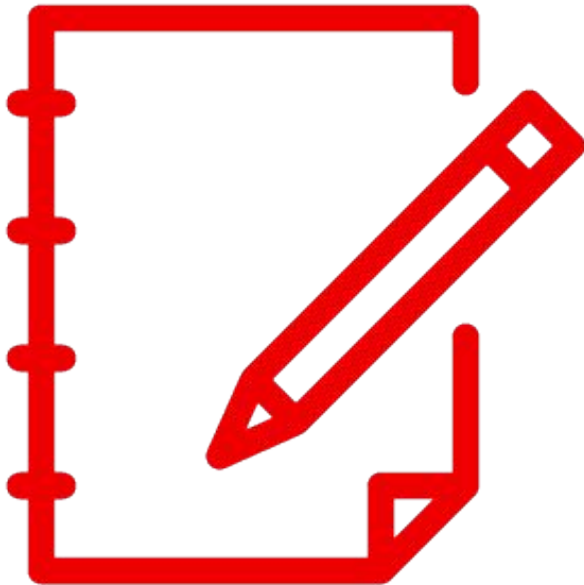
Deploy the right applications to the right compute resources.



- ▶ Labels allow application pods to automatically deploy to the correct compute resources.
- ▶ Examples include CPU or Memory intensive workloads, or workloads requiring GPU resources.

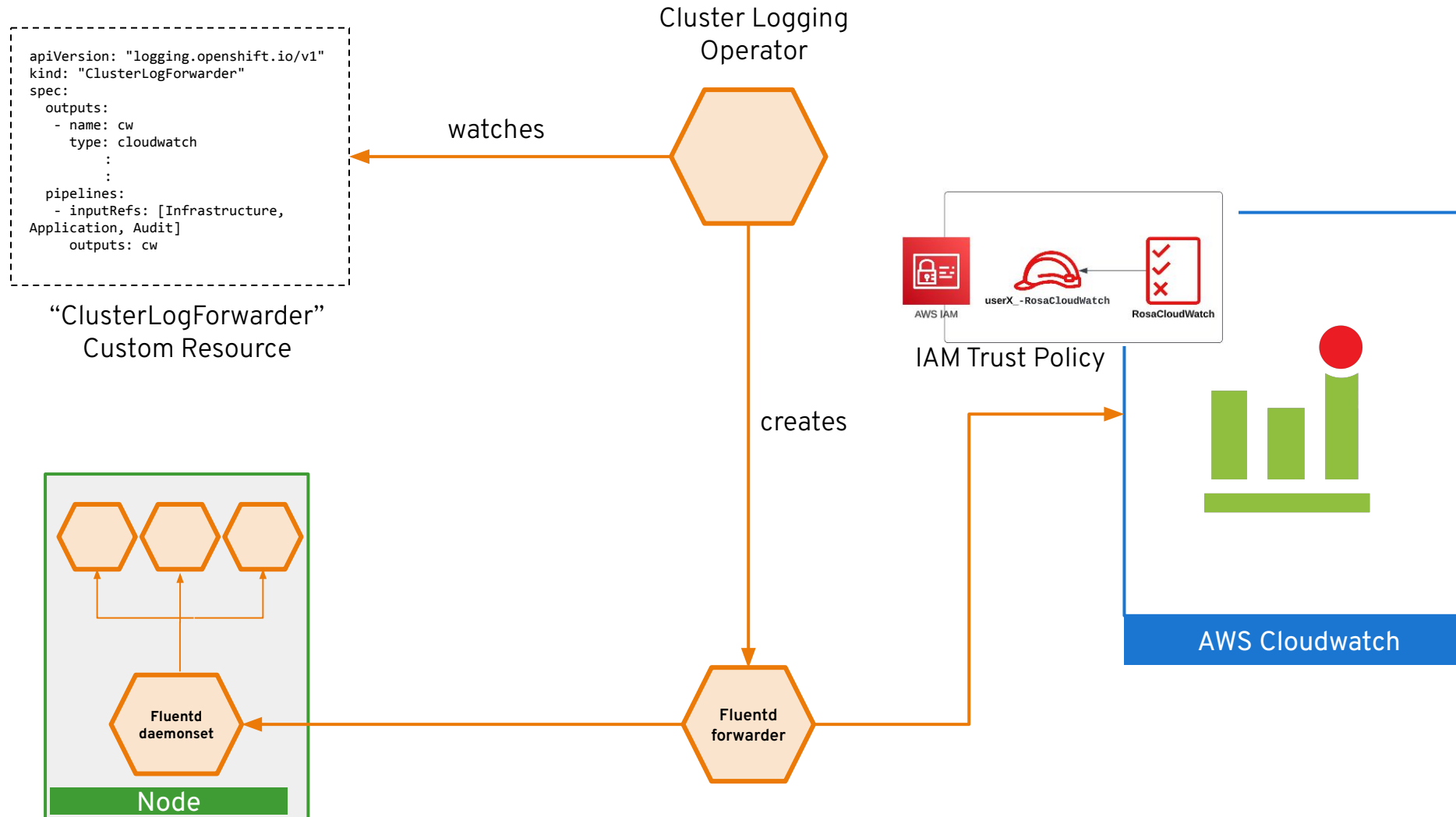
Logging with AWS Cloudwatch

Shipping logs to an enterprise-wide log management system.



- ▶ **OpenShift Cluster logs** are **stored in cluster** by **default**.
- ▶ **Cluster logs** can be **shipped** to a variety of log management systems such as **FluentD, ElasticSearch, Syslog, AWS CloudWatch, Loki, Kafka, and Splunk**.

Secure Log Forwarding to Cloudwatch

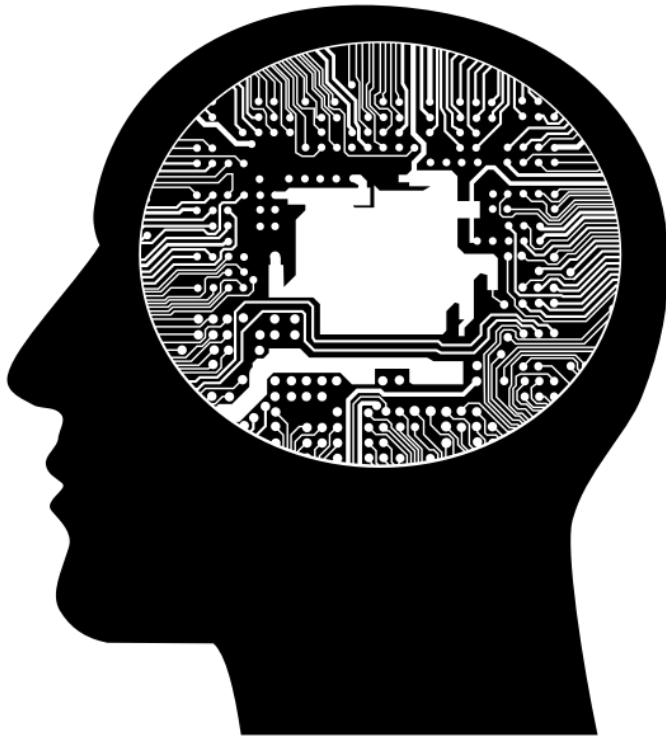


Section 2:

Deploy and Expose an Application

Deploy and Expose an Application

What you'll learn today.



- ▶ Deploying Applications
- ▶ Restricting Network Access
- ▶ Making Applications Resilient

Deploy and Expose an Application



- ▶ **Deploying Applications** - In this module we will deploy an application based on Quarkus that leverages a Amazon DynamoDB database. We will show how to leverage IAM service accounts for DB authentication, as well as how to leverage source-2-image for a true application platform experience.
- ▶ **Restricting Network Access** - OpenShift makes it easy to implement zero-trust networking policies. In this module we will restrict network access to our application.
- ▶ **Making Applications Resilient** - In this module we will learn how to make an application resilient by leveraging Pod Disruption Budgets, and the Horizontal Pod Autoscaler.

Deploying Applications

Deploy a Java based application using Quarkus and S2I.



- ▶ **Source-2-Image (S2I)** takes **application code** and **bundles it into a container** that can be **ran in OpenShift**.
- ▶ **Quarkus incorporates S2I** as part of it's build system, and can **automatically deploy** an application **to OpenShift** based on the application configuration.
- ▶ **Service Accounts** in OpenShift can **map to IAM roles** that **grant access to cloud resources** such as Amazon DynamoDB.

Restricting Network Access

Limit application access using NetworkPolicy.



- ▶ **NetworkPolicy** allows for applications to leverage the concepts of **Zero-Trust Networking: Deny by default, explicitly allow ingress/egress.**
- ▶ **NetworkPolicy** can **dynamically select** allowed or disallowed **clients** by leveraging **Pod or Namespace labels.**

Making Applications Resilient



- ▶ ROSA allows for applications to scale or recover from failure.
- ▶ **PodDisruptionBudgets** define the **minAvailable** and **maxUnavailable pods** for a given application (based on labels).
- ▶ **HorizontalPodAutoscaler** (HPA) allows for applications to **scale based on resource consumption** such as **CPU or RAM** utilization.

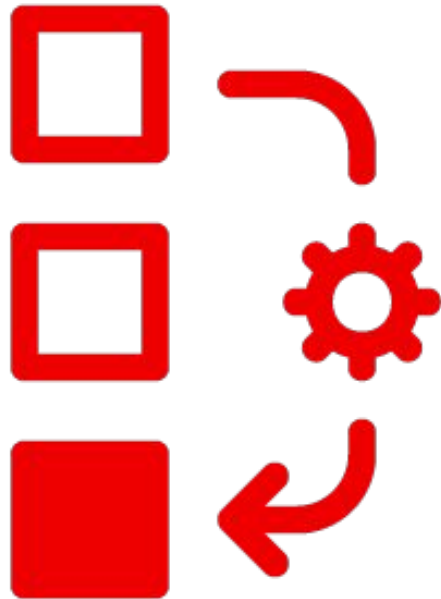
Using OpenShift GitOps

Consistent Code Across Environments



- ▶ **Treat everything as code:** Define the state of infrastructure, applications, and configurations with declarative code across environments
- ▶ **Single Source of Truth:** Infrastructure and applications are stored and versioned in Git allowing for traceability and visibility into changes that affect their entire state
- ▶ **Enhanced security:** Preview changes, detect configuration drifts, and take action
- ▶ **Visibility and audit:** Capture and trace any change to clusters through Git history
- ▶ **Multi-cluster consistency:** Combine GitOps with Advanced Cluster Manager for Kubernetes to configure multiple clusters and deployments reliably and consistently

Automate Deploying the App with Tekton

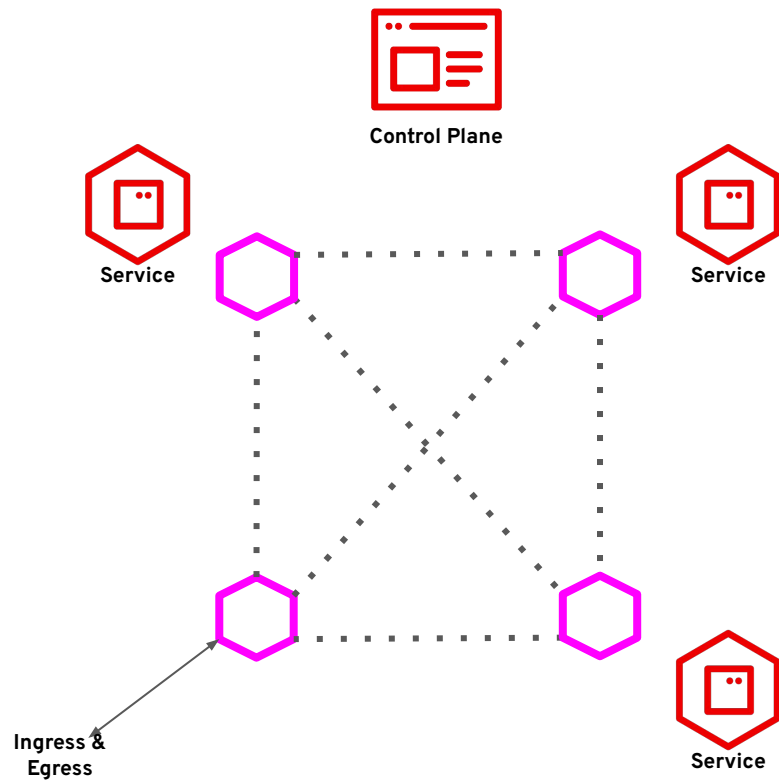


- ▶ **Cloud-Native Pipelines:** Scalable, portable, and containerized CI/CD workflows aligned with OpenShift's cloud-native architecture
- ▶ **Decoupled and Reusable Tasks:** Define and share reusable tasks, reducing duplication and improving maintainability.
- ▶ **Kubernetes-Native Custom Resources:** Manage pipelines using familiar Kubernetes tools and concepts.
- ▶ **Integration with OpenShift Pipelines:** Higher-level abstractions and tooling for quick setup via web console or CLI.
- ▶ **Security and Compliance:** Robust OpenShift security features extend to Tekton pipelines, ensuring protection and compliance.

Section 3:

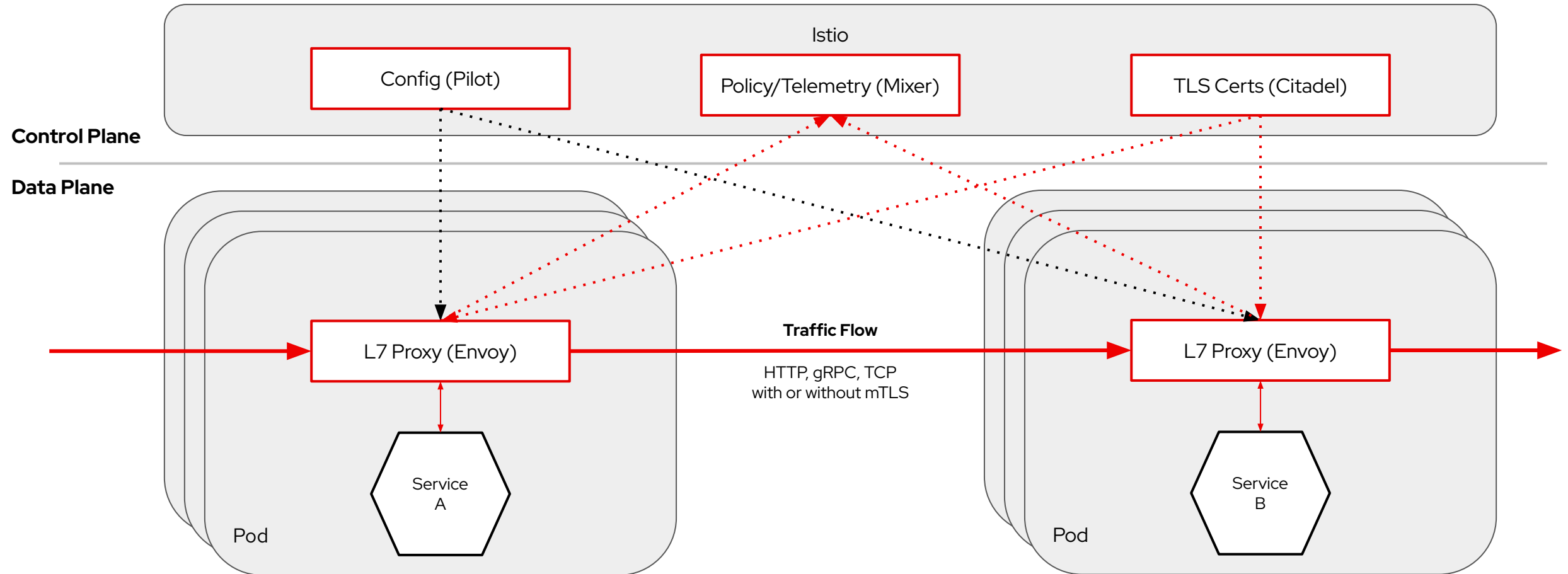
OpenShift Service Mesh

Intro to OpenShift Service Mesh




- ▶ Used to **connect, secure, and monitor microservices** in OpenShift
- ▶ Based on open source **Istio** project with add-ons **Kiali, Jaeger, Prometheus, Elasticsearch**, and **Grafana**
- ▶ Modern features such as **canary deployments, mutual TLS**, and **federation**
- ▶ Supported by Red Hat

Service Mesh Architecture



Deploying the Service Mesh Operator

**Red Hat OpenShift Service Mesh**

2.0.0-2 provided by Red Hat, Inc.

Install

Latest Version
2.0.0-2

Capability Level

- ☒ Basic Install
- ☒ Seamless Upgrades
- ☐ Full Lifecycle
- ☐ Deep Insights
- ☐ Auto Pilot

Provider Type
Red Hat

Provider
Red Hat, Inc.

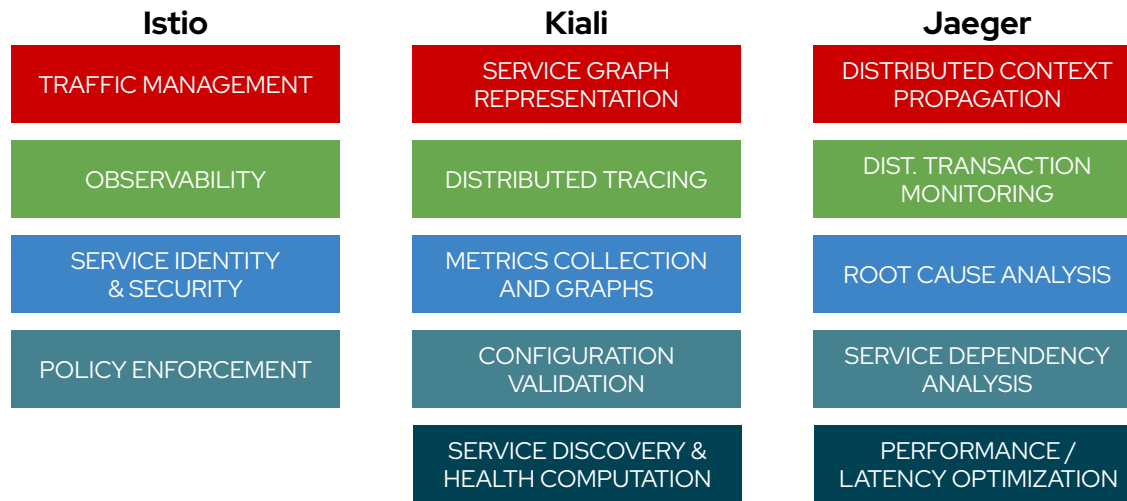
Red Hat OpenShift Service Mesh is a platform that provides behavioral insight and operational control over a service mesh, providing a uniform way to connect, secure, and monitor microservice applications.

Overview

Red Hat OpenShift Service Mesh, based on the open source [Istio](#) project, adds a transparent layer on existing distributed applications without requiring any changes to the service code. You add Red Hat OpenShift Service Mesh support to services by deploying a special sidecar proxy throughout your environment that intercepts all network communication between microservices. You configure and manage the service mesh using the control plane features.

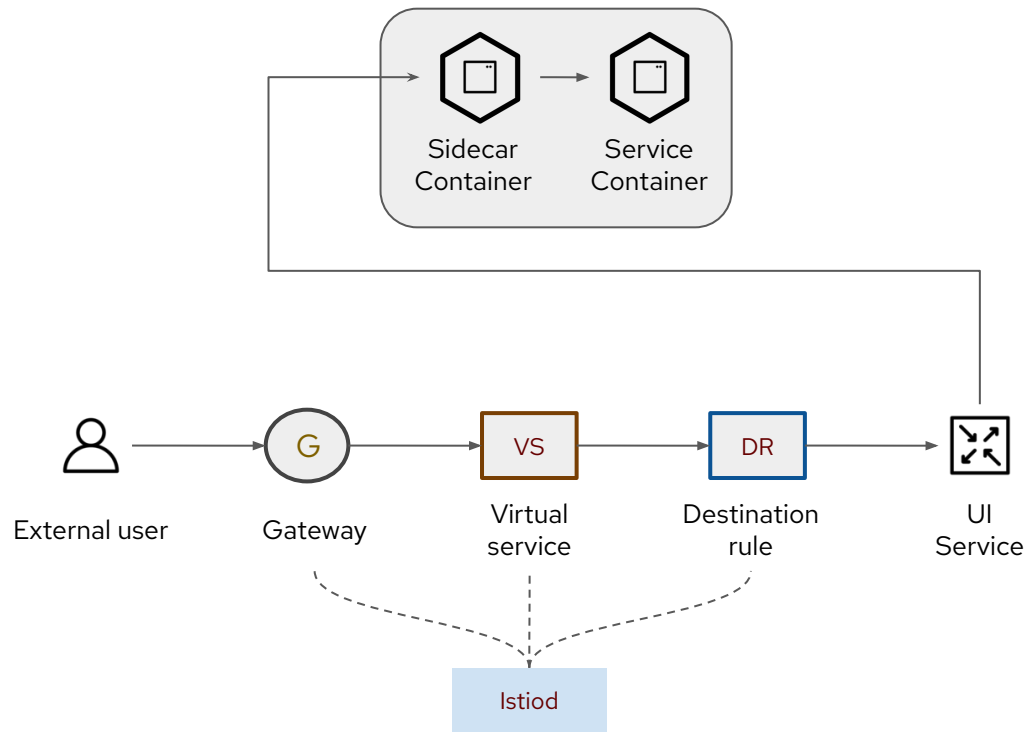
- ▶ Requires prerequisite operators **Elasticsearch, Jaeger, and Kiali**
- ▶ All operators installed via **OperatorHub** or an OpenShift **Subscription** resource
- ▶ All operators **supported** by Red Hat
- ▶ Provides **CustomResourceDefinition** resources to configure and manage **control plane** and **data plane**

Deploy Service Mesh Control Plane



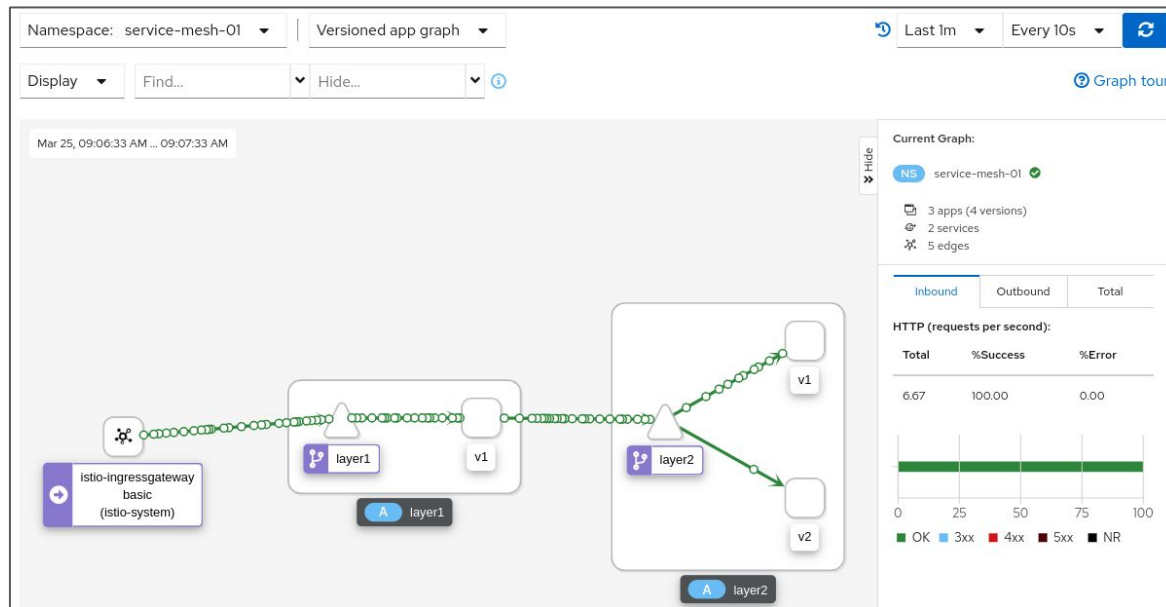
- ▶ Manage installation via a **ServiceMeshControlPlane** Custom Resource Definition
- ▶ Used to **manage data plane** resources
- ▶ May be **federated** with control planes existing in different clusters

Deploy Workloads



- ▶ **Data plane** formed by Envoy sidecar proxy container and a service container
- ▶ External communication flows through **gateway proxies**
- ▶ **ServiceMeshMemberRoll** and **ServiceMeshMember** resources allow workloads in projects to be part of the mesh

Configure and Observe Traffic



- ▶ Kiali integration with the OpenShift console provides a **single interface** for traffic visualization and management
- ▶ **Grafana and Prometheus** provide out of the box **metrics and monitoring** for all services
- ▶ **Jaeger** and **Elasticsearch** capture **distributed traces** for isolating bottlenecks between services

Learn More

Official Red Hat Course:

- ▶ [Building Resilient Microservices with Istio and Red Hat OpenShift Service Mesh](#)

Official Red Hat Documentation:

- ▶ [Service Mesh 2.x Documentation](#)

Open Source Documentation:

- ▶ [Istio Documentation](#)

Linux Foundation Course:

- ▶ [Introduction to Istio](#)

Red Hat Developer Resources:

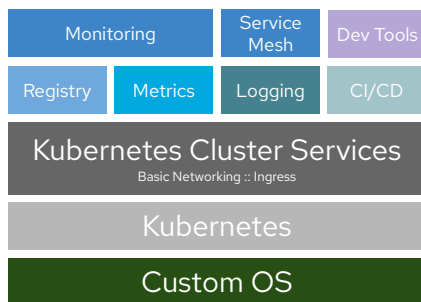
- ▶ [Featured Service Mesh Resources](#)

Wrapping Up!

Building & running a platform vs a turnkey Cloud Service



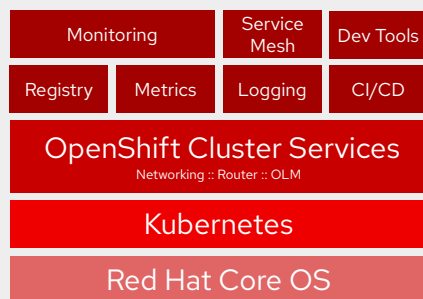
The Parts



xKS + 'native'
services



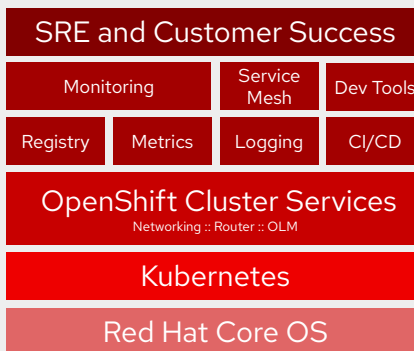
The Assembled Car



- Application Platform -
Self-managed Red Hat OpenShift



The Car & Pit Crew



- Turnkey Application Platform -
Red Hat OpenShift cloud services

"Batteries Included"

... but swappable

Individual components can be swapped out

Eg.

- Using AWS CloudWatch for logging on AWS
- Use specific cloud services or ISV offerings

An opinionated platform for building, deploying and running applications



Service Mesh

App-Services

DB-Services

CI/CD

DNS

Authentication

Monitoring

Kubernetes

Automation

Logging

Registry

Security

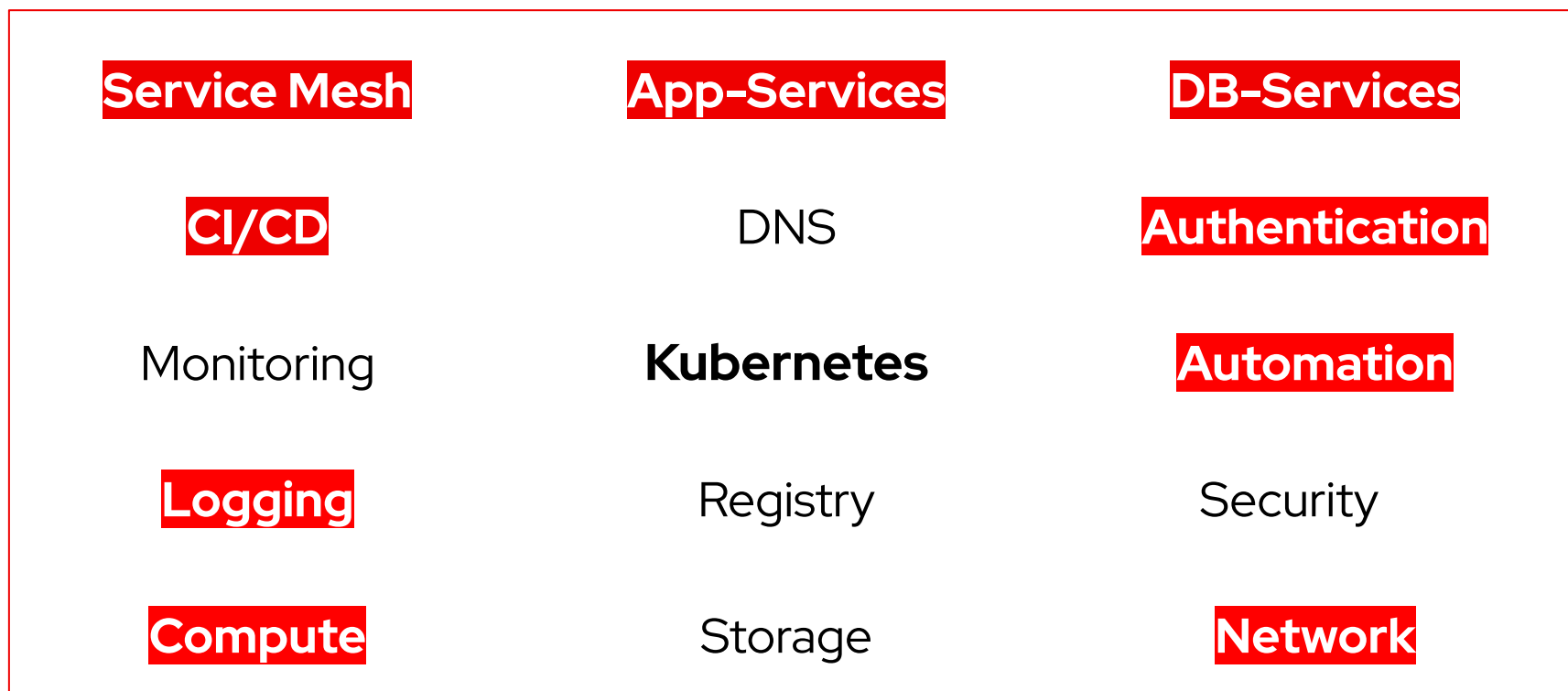
Compute

Storage

Network

- ▶ Fully integrated and supported components
- ▶ Expert SRE and Customer Success support
- ▶ Abstracts away technical details
- ▶ Consistent experience across clouds

Building blocks of a Modern Application Platform



Helpful Links

ROSA Documentation

- ▶ <https://docs.openshift.com/aro/4/welcome/index.html>

MOBB.Ninja ROSA Guides

- ▶ <https://mobb.ninja/#rosa>

Introduction to ROSA - Red Hat Training

- ▶ <https://www.redhat.com/en/services/training/DO120-introduction-to-red-hat-openshift-service-on-aws>

ROSA Lightboard Videos

- ▶ <https://www.redhat.com/en/about/videos/rosa-lightboard>

ROSA User Guide - AWS

- ▶ <https://docs.aws.amazon.com/ROSA/latest/userguide/what-is-rosa.html>

Introduction to ROSA - Red Hat Ebook

- ▶ https://access.redhat.com/documentation/en-us/red_hat_openshift_service_on_aws/4/pdf/introduction_to_rosa/red_hat_openshift_service_on_aws-4-introduction_to_rosa-en-us.pdf

Thank you!



linkedin.com/company/red-hat



youtube.com/user/RedHatVideos



facebook.com/redhatinc



twitter.com/RedHat