



# Modern Application Development Roadshow

## Ops Track

Red Hat K.K.

Solution Architect

Mayumi Koshimizu

2025年03月06日

# Modern Application Development Roadshow Ops Track

- Module 1
  - Red Hat OpenShift Platformの基礎
- Module 2
  - ハイブリッドクラウド環境でのセキュリティ強化
    - Red Hat Advanced Cluster Security for Kubernetes
- Module 3
  - マルチクラスター管理
    - Red Hat Advanced Cluster Management for Kubernetes

# Module 1

Admin Ops

# OpenShift Admin Ops

1. 演習環境の概要
2. OpenShiftクラスタのインストールの検証
3. アプリケーション管理の基礎
4. アプリケーションストレージの基礎
5. MachineSets, Machines, and Nodes
6. インフラストラクチャノードとOperator
7. OpenShift Logging with Loki
8. 外部認証プロバイダ(LDAP)の設定
9. OpenShift Monitoring
10. プロジェクト・リクエスト・テンプレートとクォータ/制限
11. OpenShift の Network Policy ベースの SDN
12. Projectのセルフプロビジョニングの無効化
13. クラスタリソースのクォータ
14. Taint と Toleration

# OpenShift Admin Ops 概要 1

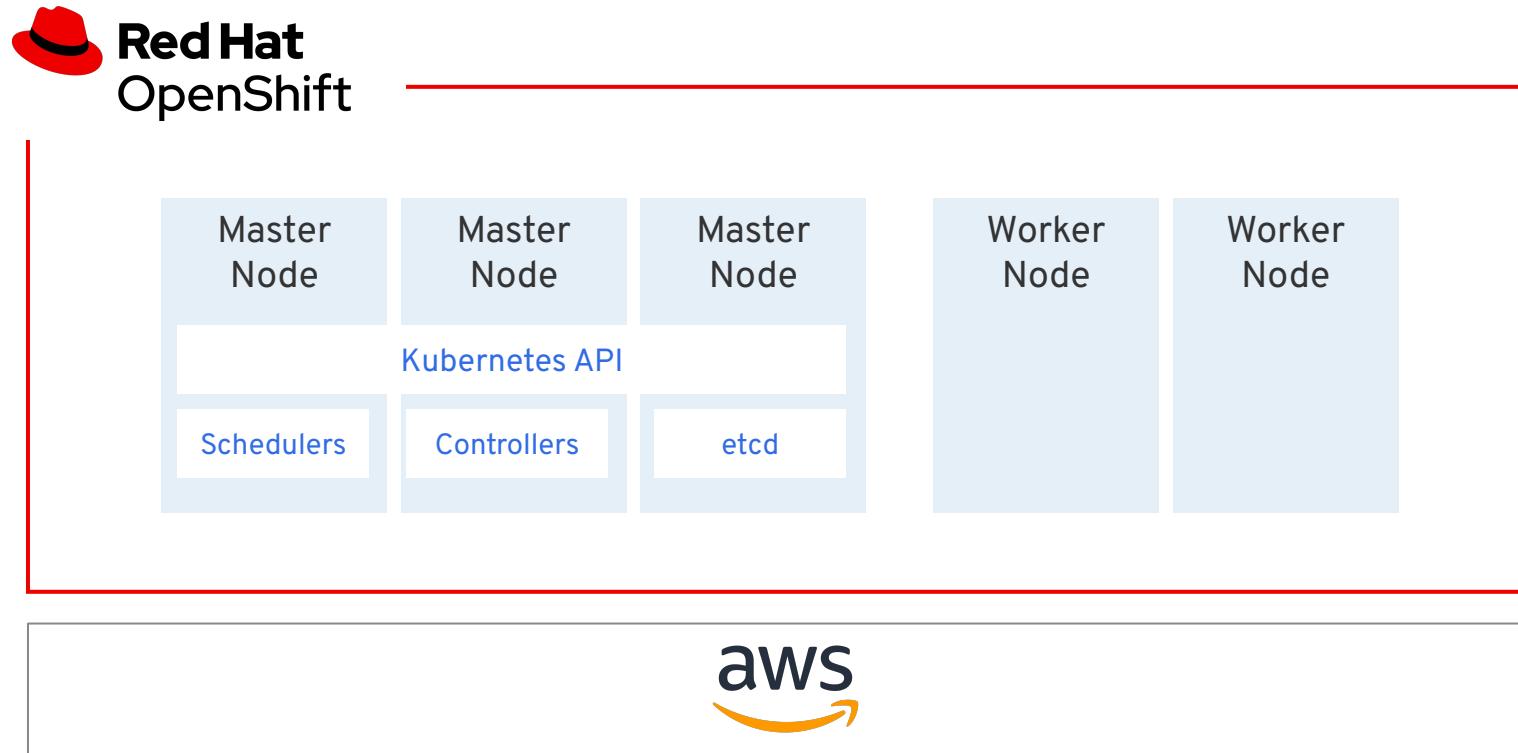
- ▶ **Installation & Verification** - インストーラープロビジョニングインフラ(IPI)を使用した場合のOpenShift 4のインストールがどのように見えるかを学びます。マスター・コンポーネント、API/認証、データストアとスケューラー、そしてヘルスとスケーリングについて詳しく学びます。
- ▶ **Application Management Basics** - ocツールを使用してサンプルアプリケーションをデプロイし、OpenShift Container Platform上のアプリケーション管理のコアコンセプト、基本オブジェクト、および基礎のいくつかについて学習します。
- ▶ **Application Storage Basics** - アプリケーションストレージの基本を学びます。具体的には、永続的なボリュームクレームの追加方法、ストレージクラス、永続的なストレージのテストなどがあります。
- ▶ **MachineSets, Machines, and Nodes** - ここでは、MachineSetの基礎と、MachineSetがマシンオブジェクトのセットに対してどのように望ましい状態を定義するのかを学びます。
- ▶ **Infra Nodes & Operators** - 追加の MachineSets とそのメタデータについて学びます。また、セレクタ、テンプレート・メタデータ、テンプレート仕様、およびカスタムMachineSetsについても学びます。
- ▶ **OpenShift Log Aggregation (Optional)** - OpenShiftのロギングをデプロイし、ネームスペースの仕組みを学び、クラスタ内で LokiとクラスタロギングOperatorを使用する方法を理解します。
- ▶ **External (LDAP) Authentication Providers, Users & Groups** - OpenShiftは、多くの異なる認証プロバイダーをサポートしています。このモジュールでは、OAuthの設定を調べ、LDAPグループをOpenShiftのグループに同期し、グループポリシーに取り組みます。また、Prometheusを見ながら、コラボレーションのためのプロジェクトも作成します。

# OpenShift Admin Ops 概要 2

- ▶ **OpenShift Monitoring with Prometheus** - 組み込みのOpenShiftモニタリングのさまざまな側面を探ります。
- ▶ **Project Template, Quota, & Limits** - デフォルトのプロジェクトリクエストテンプレートを表示する方法と、プロジェクトリクエストテンプレートを変更する方法を学びます。また、ResourceQuotasとLimitRangesについて、そしてプロジェクトリクエストテンプレートのインストール方法について学びます。
- ▶ **OpenShift Networking & Network Policy** - kubernetes NetworkPolicyによって、プロジェクトはOpenShiftのソフトウェア定義ネットワーク内でネットワークインフラを分離することができます。このモジュールでは、その仕組みについて説明します。
- ▶ **Disabling Project Self-Provisioning** - OpenShift内のプロジェクトのセルフプロビジョニングを無効にする方法について学びます。ClusterRoleBindingを調べる方法、セルフプロビジョニングプロジェクトを削除する方法、リクエストメッセージをカスタマイズする方法について学びます。
- ▶ **Cluster Resource Quotas** - クォータを設定するためのさまざまな方法を説明します。クォータは、ユーザー、個々のプロジェクト、または複数のプロジェクトにわたって設定することができます。
- ▶ **Taints & Tolerations (Optional)** - このラボでは、taintとtolerationを使用してワークロードの配置を制御する方法について説明します。また、テストアプリケーションをデプロイし、テインティングノードを実行することになります。

# OpenShiftクラスタのインストールの検証

ワークショップの環境は、AWS上にIPI(Installer-provisioned infrastructure)インストールされています



## Master Node (3 nodes)

Master Nodesには、コンテナを制御するコンポーネントと、クラスタの状態を構成管理するデータ(etcd)があります。

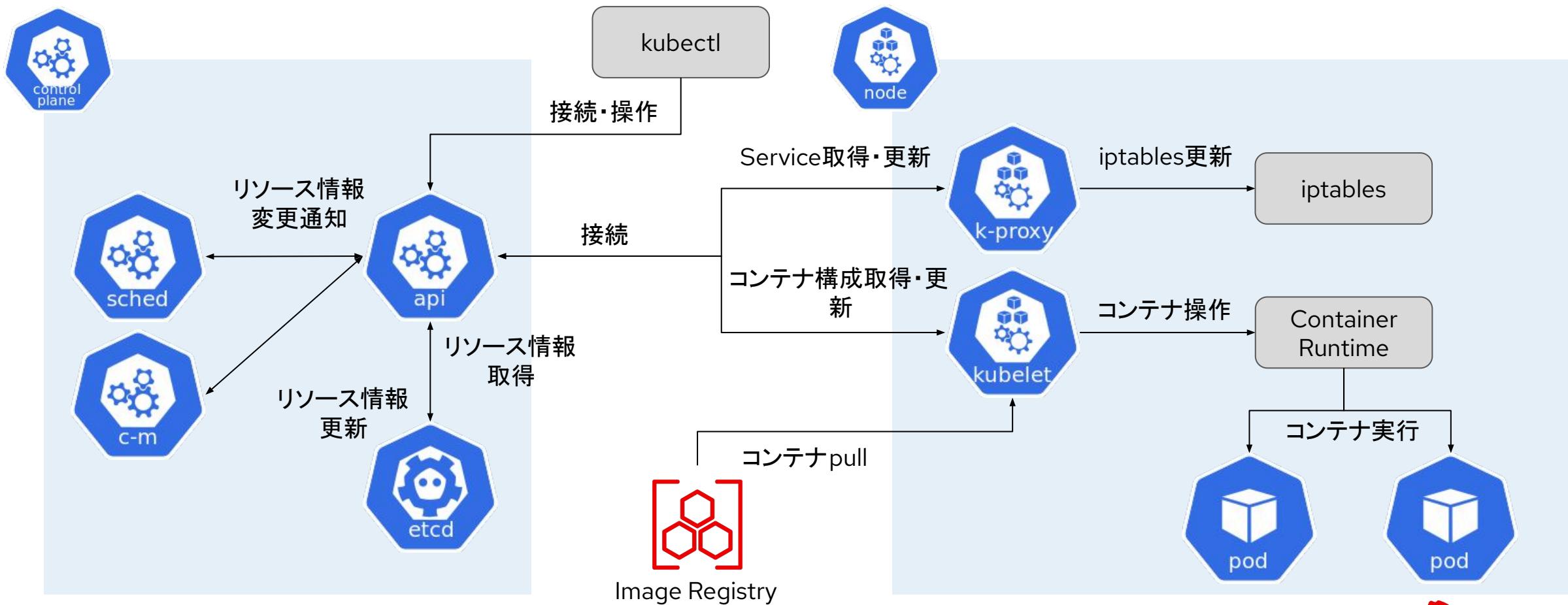
**Controllers:** リソースのデプロイを行う機能

**Schedulers:** リソースの空き状況を管理する機能

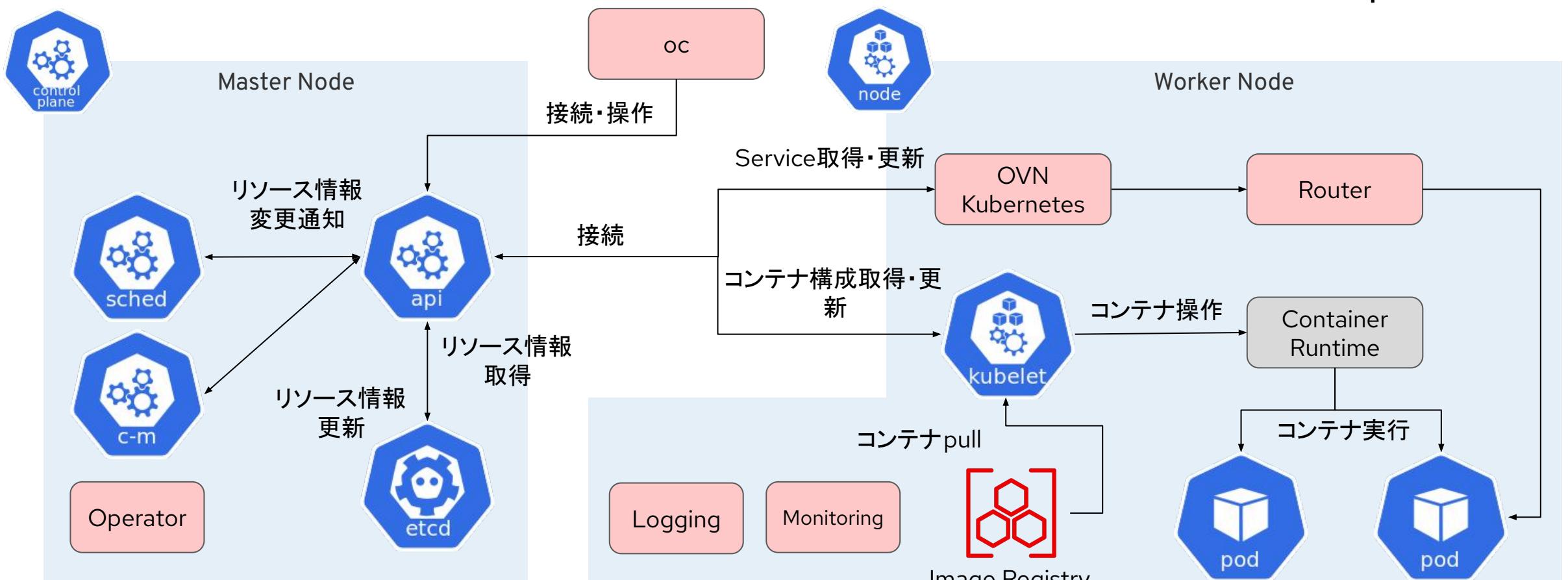
## Worker Node (2~N nodes)

Worker Nodesでは、スケジュールされたコンテナがデプロイされ、コンテナを死活監視します。

# Kubernetesのアーキテクチャ

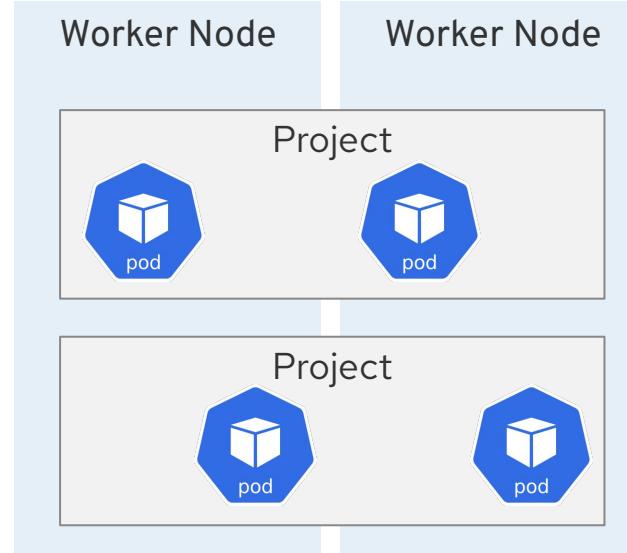


# OpenShiftのアーキテクチャ



# アプリケーション管理の基礎

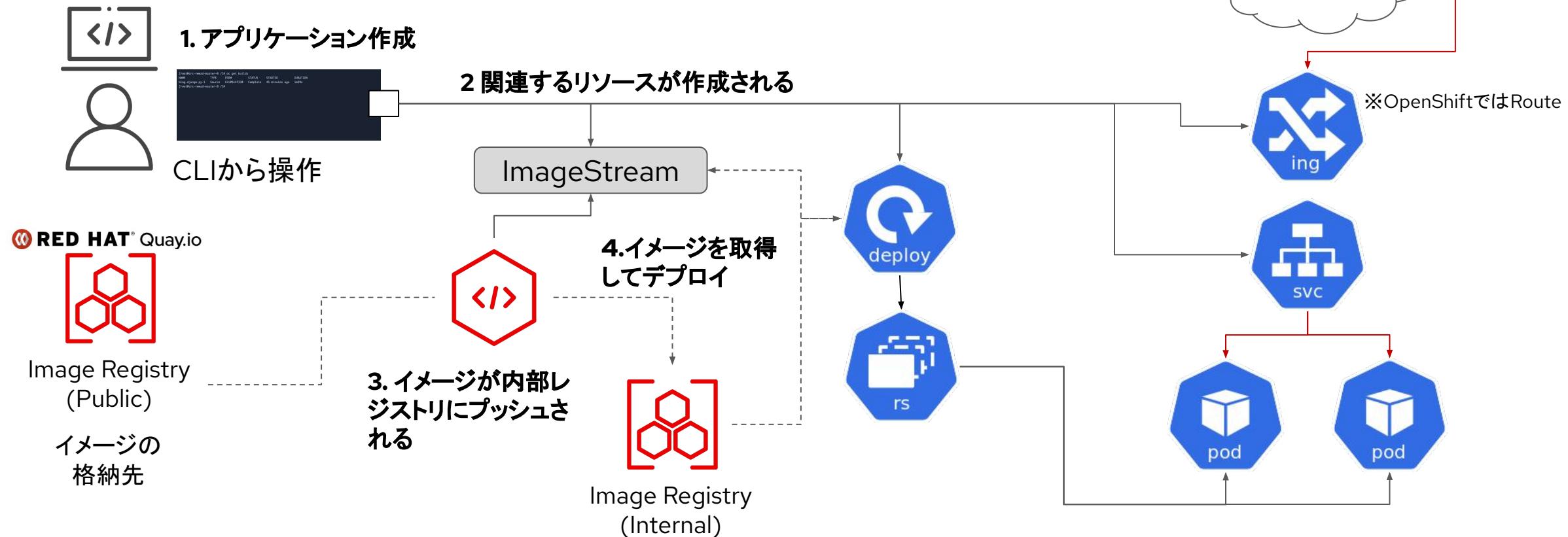
OpenShiftのコアとなる概念について学びます



リソース	概要
Project	Project単位でリソースを分離。管理の観点からは、各Projectはテナントのように考えることができます
Pod	ホスト上に一緒にデプロイされた1つまたは複数のコンテナ。PodはOpenShiftで定義、デプロイ、管理できるコンピュートリソースの最小の単位です
Service	OpenShift環境内でPodのようなグループを見つけるための抽象化レイヤーを提供。OpenShift環境内からPodにアクセスする必要があるものとの間の内部プロキシ/ロードバランサーとしても機能します
Deployment	OpenShift内に何をどのようにデプロイするかを定義します
ReplicaSet	必要な数のPodを確実に存在させるために使用されます。ReplicaSetはOpenShiftが自己修復する方法を提供します
Route	OpenShift外のクライアントがOpenShift内で実行されているアプリケーションにアクセスする方法を提供します

# アプリケーション管理の基礎

イメージを利用したアプリケーションのデプロイを行い作成されたリソースを確認します



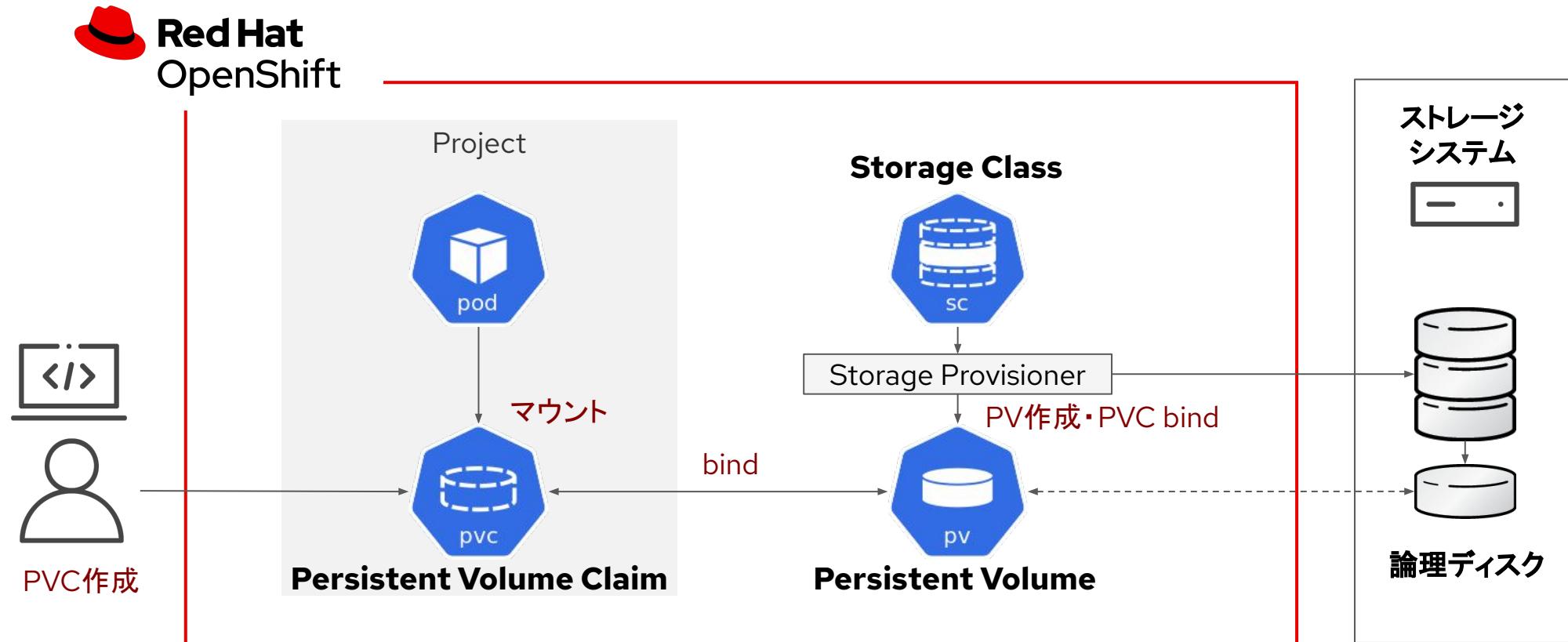
# アプリケーション管理の基礎

イメージを利用したアプリケーションのデプロイを行い作成されたリソースを確認します

1. プロジェクト作成: `oc new-project app-management`
2. アプリケーション作成: `oc new-app quay.io/openshiftroadshow/mapit`
3. アプリケーションに関するリソースが作成されます  
各リソースの内容を確認します
4. routeを作成し、外部からアクセスできることを確認します

# アプリケーションストレージの基礎

OpenShiftにおける永続ボリューム(Persistence Volume)の動作を確認します



# アプリケーションストレージの基礎

OpenShiftにおける永続ボリューム(Persistence Volume)の動作を確認します

1. アプリケーション mapit で下記の内容の永続ボリュームを使用します

PVC名 : mapit-storage

ReadWriteOnce モード

サイズ : 1Gi

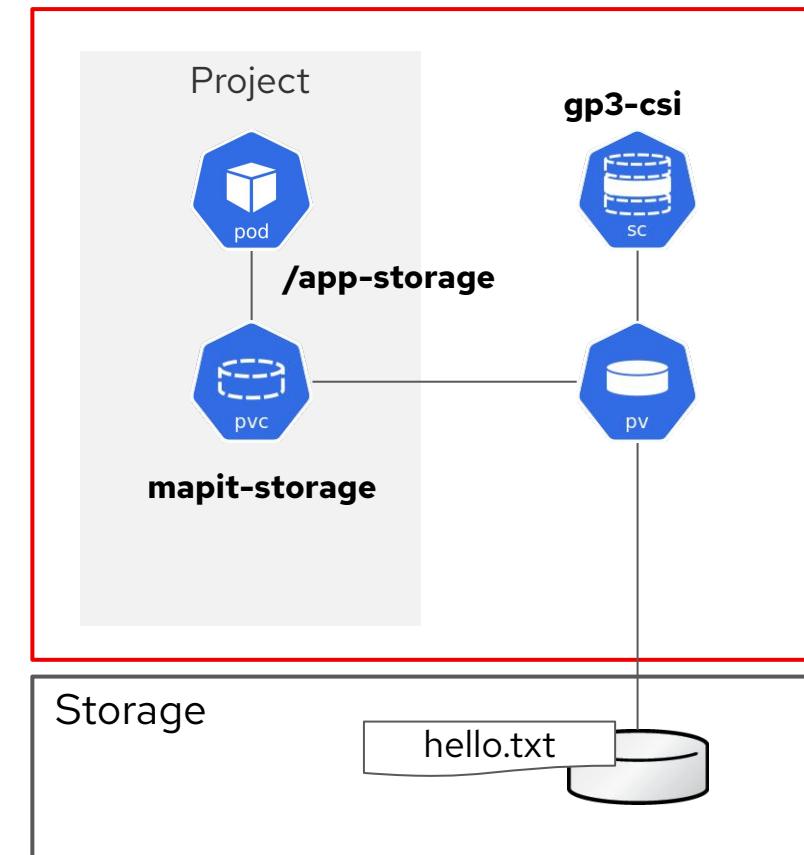
mount-path : /app-storage

2. 永続ボリュームにファイルを作成します

3. 稼働している Pod を削除します

4. 新しい Pod がすぐに稼働します

5. 永続ボリュームに作成したファイルを継続して利用できることを確認します



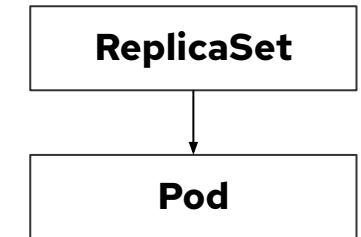
# MachineSet, Machine, and Node

MachineSet/Machineを利用して、ノードを環境に追加(削除)します

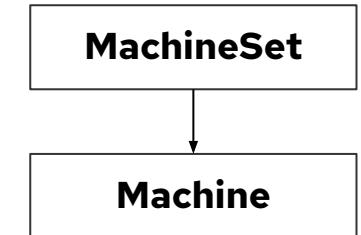


cluster-9fddq-vk46x-worker-ap-southeast-1c  
cluster-9fddq-vk46x-worker-ap-southeast-1b  
cluster-9fddq-vk46x-worker-ap-southeast-1a

ReplicaSetがPodの数を  
コントロール



MachineSetがMachine(node)の  
数をコントロール



## MachineSet

cluster-9fddq-vk46x-worker-ap-southeast-1a	1	1	1	31h
cluster-9fddq-vk46x-worker-ap-southeast-1b	1	1	1	31h
cluster-9fddq-vk46x-worker-ap-southeast-1c	0	0	0	31h

## Machine

cluster-9fddq-vk46x-worker-ap-southeast-1a-hpwng	Running
cluster-9fddq-vk46x-worker-ap-southeast-1b-d7dll	Running

cluster-9fddq-vk46x-worker-ap-southeast-1a-hpwng	Running
cluster-9fddq-vk46x-worker-ap-southeast-1b-d7dll	Running
cluster-9fddq-vk46x-worker-ap-southeast-1c-l5jjw	Provisioned

## Node

ip-10-0-198-39.ap-southeast-1.compute.internal	
ip-10-0-209-58.ap-southeast-1.compute.interna	



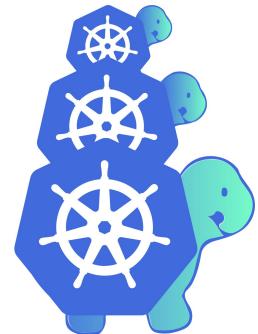
# Machine-API

Machine-APIはアップストリームのKubernetes Cluster APIをOpenShiftに機能実装したもの

## Cluster API

kubernetesクラスタのプロビジョニング、アップグレード、運用を簡素化するための宣言型 APIとツールの提供にフォーカス  
クラスタのライフサイクル管理を自動化する  
多種多様なインフラストラクチャ環境において、一貫性のある再現可能なクラスタのデプロイを可能にする

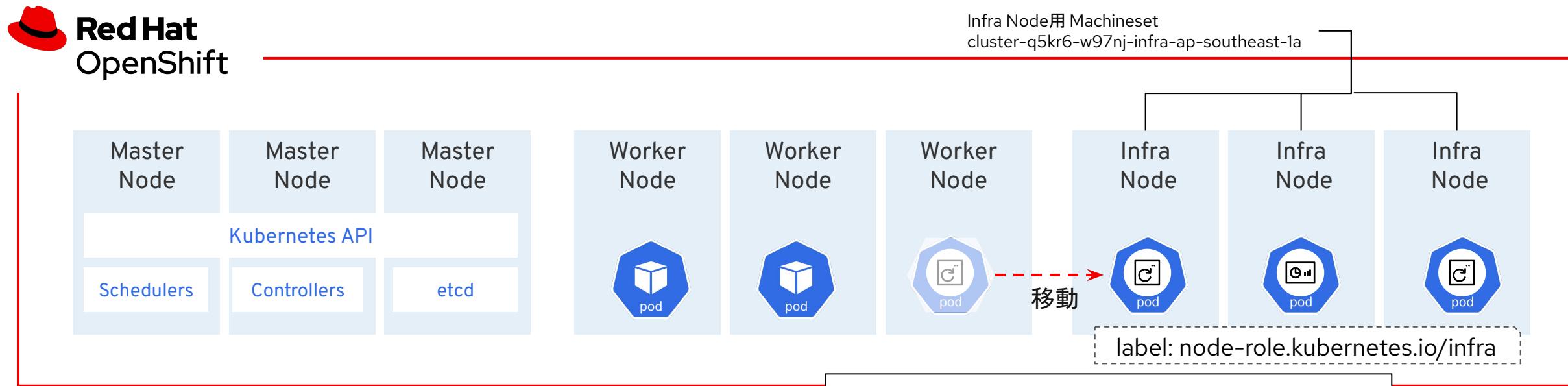
- ・ノードのスケールイン・スケールアウトがコマンドラインから可能
- ・ノードのスケール調整そのものを自動化( autoscale)
- ・ノードのローリング青王データ



参考資料 : <https://www.slideshare.net/ssuser804f1a/machine-configoperator>

# インフラストラクチャノード と Operator

Infra Nodeを作成し、Worker Node上で稼働しているPod(Router,Registry等)をInfra Nodeに移動させます



Infra Nodeには、OpenShiftの追加サービスを管理するためのコンポーネントが配置されます

Infra Nodeに配置できる機能

- ・Router
- ・OpenShift-included Registry
- ・OpenShift cluster monitoring
- ・OpenShift log aggregation
- など

```
apiVersion: operator.openshift.io/v1
kind: IngressController
metadata:
finalizers:
- ingresscontroller.operator.openshift.io/finalize
name: default
namespace: openshift-ingress-operator
spec:
nodePlacement:
nodeSelector:
matchLabels:
node-role.kubernetes.io/infra: ""
```

例) Router PodがInfra Nodeで稼働するようnodeSelectorを指定



# インフラストラクチャノード と Operator

例) cluster-5gqm-pkmg-worker-ap-southeast-1a

```
spec:  
replicas: 1  
selector:  
matchLabels:  
    machine.openshift.io/cluster-api-cluster: cluster-5gqm-pkmg  
    machine.openshift.io/cluster-api-machineset: cluster-5gqm-pkmg-worker-ap-southeast-1a  
template:  
metadata:  
labels:  
    machine.openshift.io/cluster-api-cluster: cluster-5gqm-pkmg  
    machine.openshift.io/cluster-api-machine-role: worker  
    machine.openshift.io/cluster-api-machine-type: worker  
    machine.openshift.io/cluster-api-machineset: cluster-5gqm-pkmg-worker-ap-southeast-1a  
spec:  
lifecycleHooks: {}  
metadata: {}  
providerSpec:  
value:  
ami:  
    id: ami-0f827a1be73b9de83 AmazonマシンイメージID
```

レプリカ数  
クラスタID  
machineset名  
クラスタID  
machineset名

# インフラストラクチャノード と Operator

例) cluster-5gqmq-pkmg1-worker-ap-southeast-1aをもとにInfra Node用の新しいmachinesetを作成しスケールアウト

## machineset-generator.shの内容

```
#!/bin/bash
export AMI=$(oc get machineset -n openshift-machine-api -l 'machine.openshift.io/os-id!=Windows' -o
jsonpath='{.items[0].spec.template.spec.providerSpec.value.ami.id}')
AMI情報
export CLUSTERID=$(oc get infrastructures.config.openshift.io cluster -o jsonpath='{.status.infrastructureName}')
クラスタID
export REGION=$(oc get infrastructures.config.openshift.io cluster -o jsonpath='{.status.platformStatus.aws.region}')
リージョン情報
export COUNT=${1:-3}
export NAME=${2:-workerocs}
export SCALE=${3:-1}

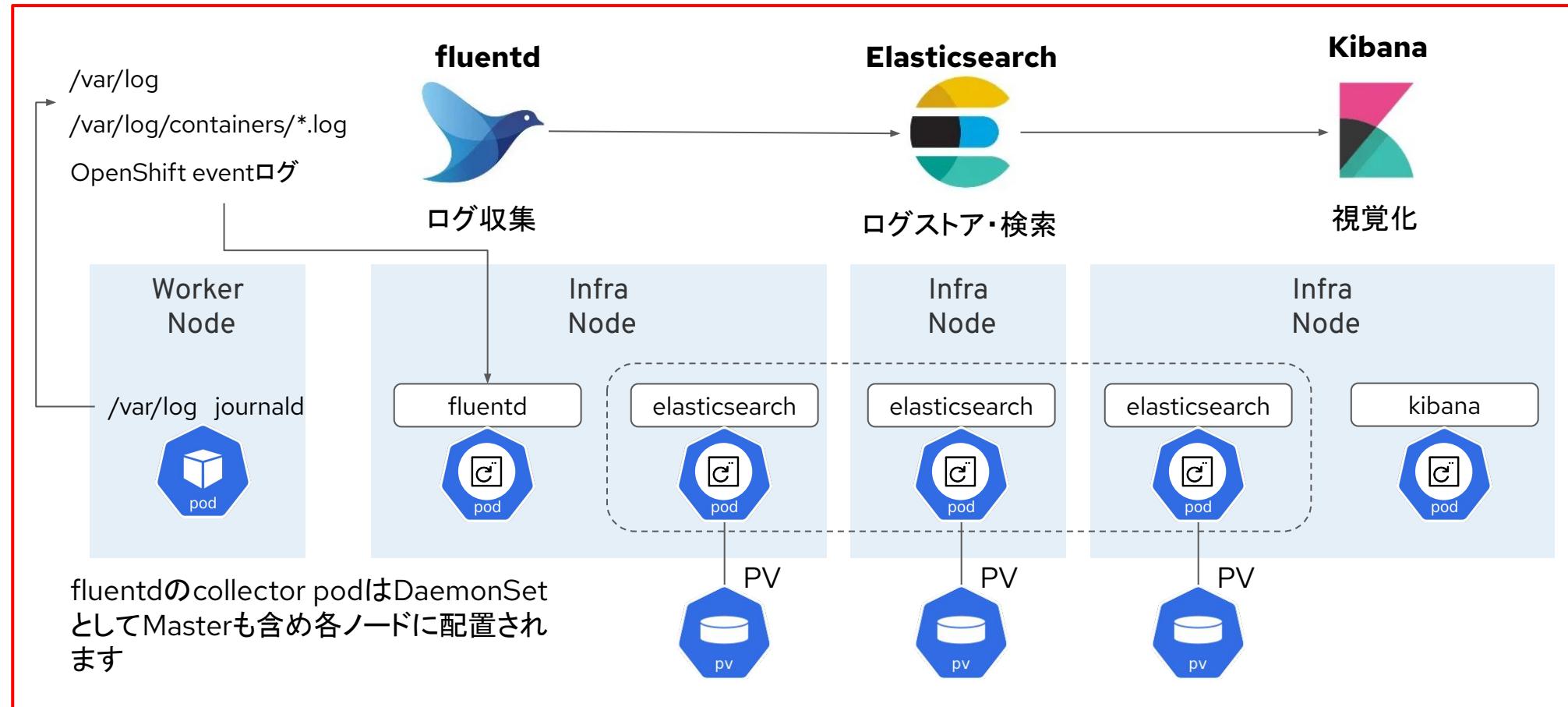
$( cd "$( dirname "${BASH_SOURCE[0]}" )" >/dev/null 2>&1 && pwd )/machineset-cli -scale $SCALE -name $NAME -count $COUNT
-ami $AMI -clusterID $CLUSTERID -region $REGION
```

## Infra Node用Machineset作成とスケールアウトの実行

```
bash {{ HOME_PATH }}/support/machineset-generator.sh 1 infra 0 | oc create -f -
export MACHINESET=$(oc get machineset -n openshift-machine-api -l machine.openshift.io/cluster-api-machine-role=infra -o
jsonpath='{.items[0].metadata.name}')
oc patch machineset $MACHINESET -n openshift-machine-api --type='json' -p='[{"op": "add", "path":
"/spec/template/spec/metadata/labels", "value": {"node-role.kubernetes.io/worker": "", "node-role.kubernetes.io/infra": ""}} ]'
oc scale machineset $MACHINESET -n openshift-machine-api --replicas=3
```

# OpenShift ログ集約

演習は従来のEFKスタックを利用してのログ集約の法になります。(現在はLoki+Vectorを使用します)



# OpenShift ログ集約

## Custom Resource

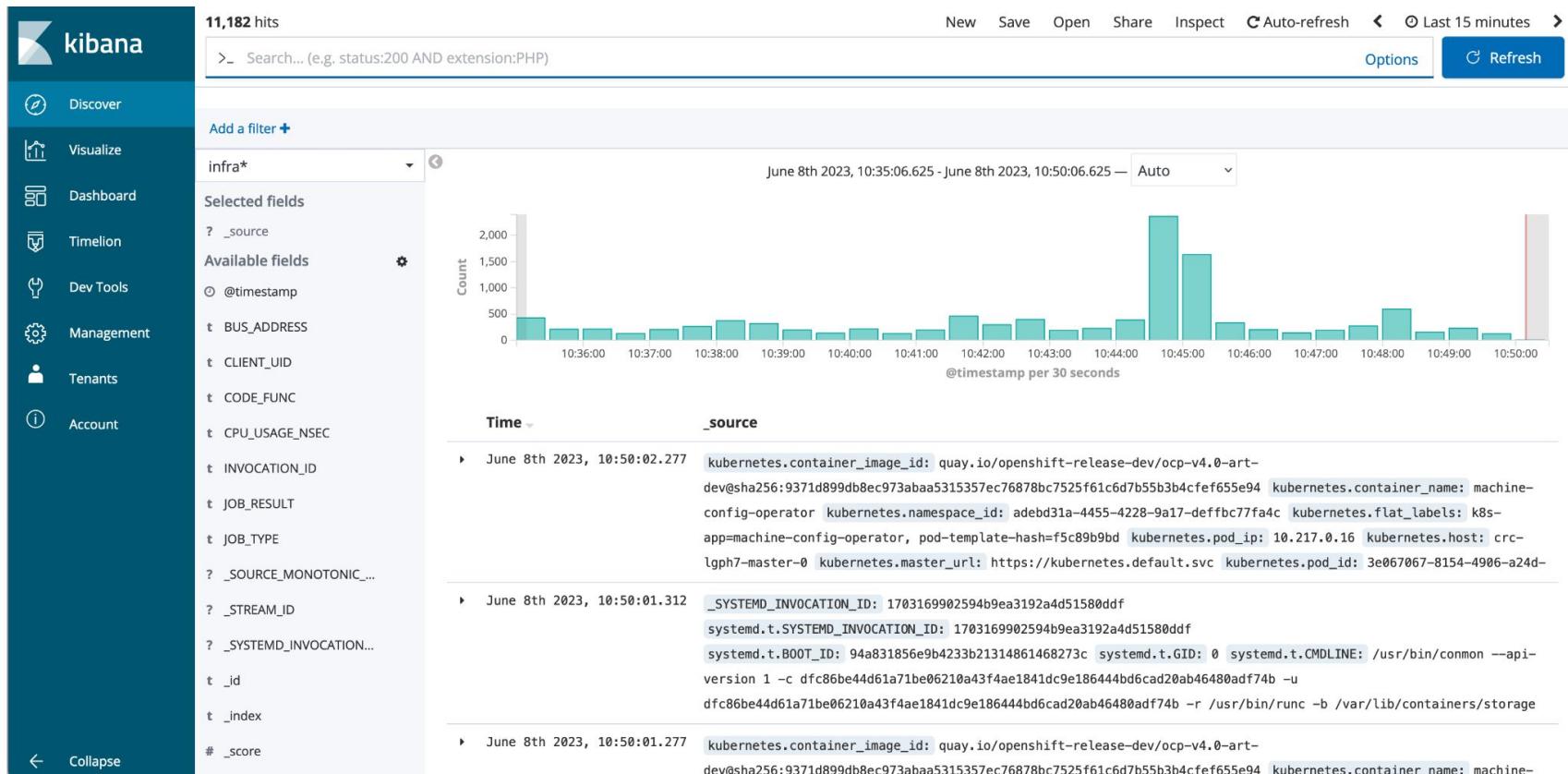
```
apiVersion: "logging.openshift.io/v1"
kind: "ClusterLogging"
metadata:
  name: "instance"
  namespace: "openshift-logging"
spec:
  managementState: "Managed"
  logStore:
    type: "elasticsearch"
    elasticsearch:
      nodeCount: 3
      storage:
        storageClassName: gp3-csi
        size: 100Gi
      redundancyPolicy: "SingleRedundancy"
    nodeSelector:
      node-role.kubernetes.io/infra: ""
    resources:
      request:
        memory: 4G
```

```
visualization:
  type: "kibana"
  kibana:
    replicas: 1
    nodeSelector:
      node-role.kubernetes.io/infra: ""
curation:
  type: "curator"
  curator:
    schedule: "30 3 * * *"
    nodeSelector:
      node-role.kubernetes.io/infra: ""
collection:
  logs:
    type: "fluentd"
    fluentd: {}
    nodeSelector:
      node-role.kubernetes.io/infra: ""
```

nodeSelectorでInfra Nodeを指定  
  
elasticsearchはストレージを使用するので、storage classとしてgp3-csi(デフォルトを指定)

# OpenShift ログ集約

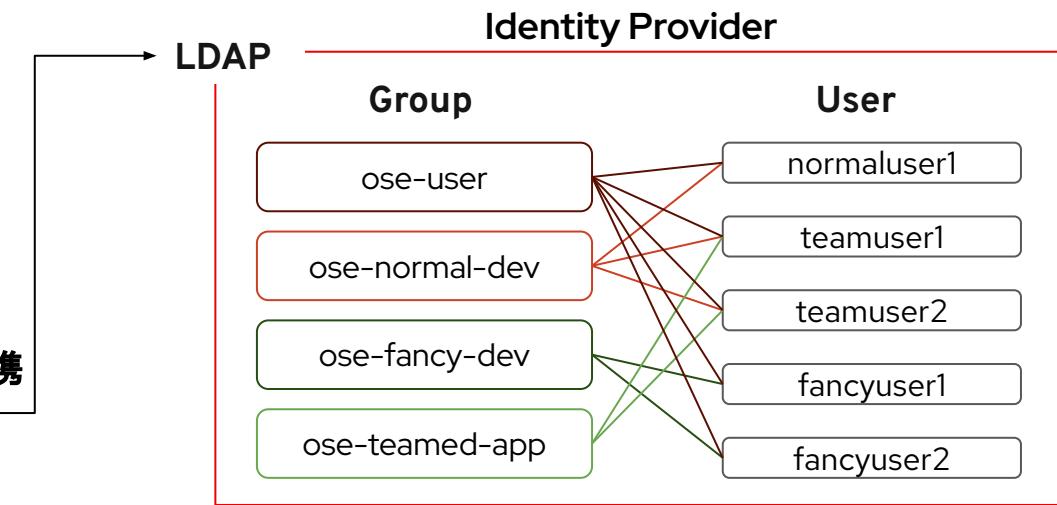
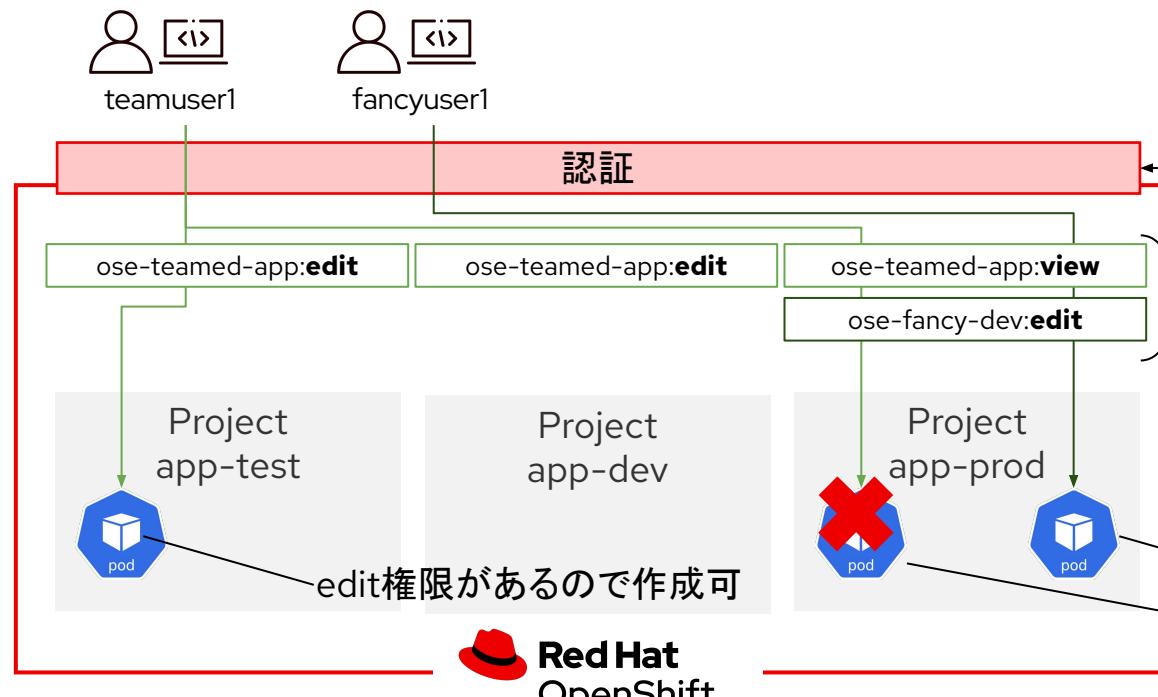
現在はLoki+Vectorを使用します



# 外部認証プロバイダ (LDAP) の設定

- LDAP連携の設定を行います
- Role Based Access Control設定と動作確認を行います

グループへのクラスタロールの追加: oc adm policy add-cluster-role-to-group  
 グループへのロールの追加: oc adm policy add-role-to-group

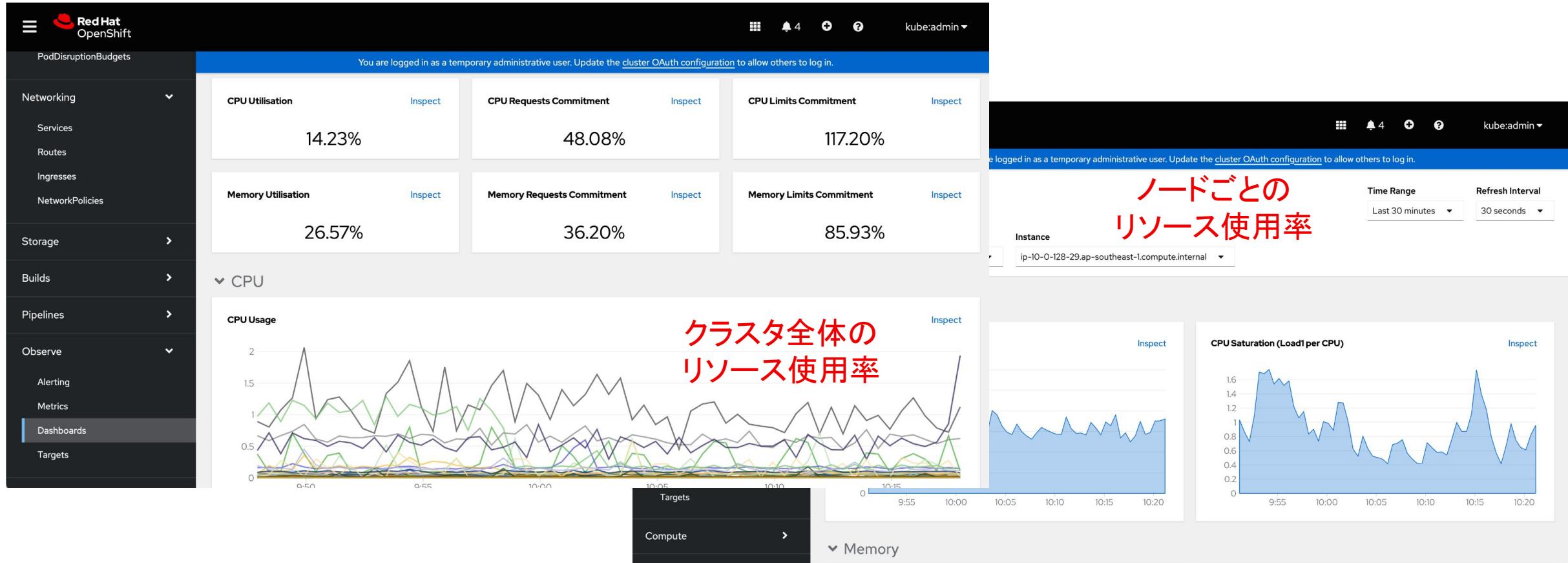


## Role Based Access Control (ロールベースアクセス制御)

edit権限があるので作成可  
view権限のみなので作成不可・閲覧は可

# OpenShift Monitoring

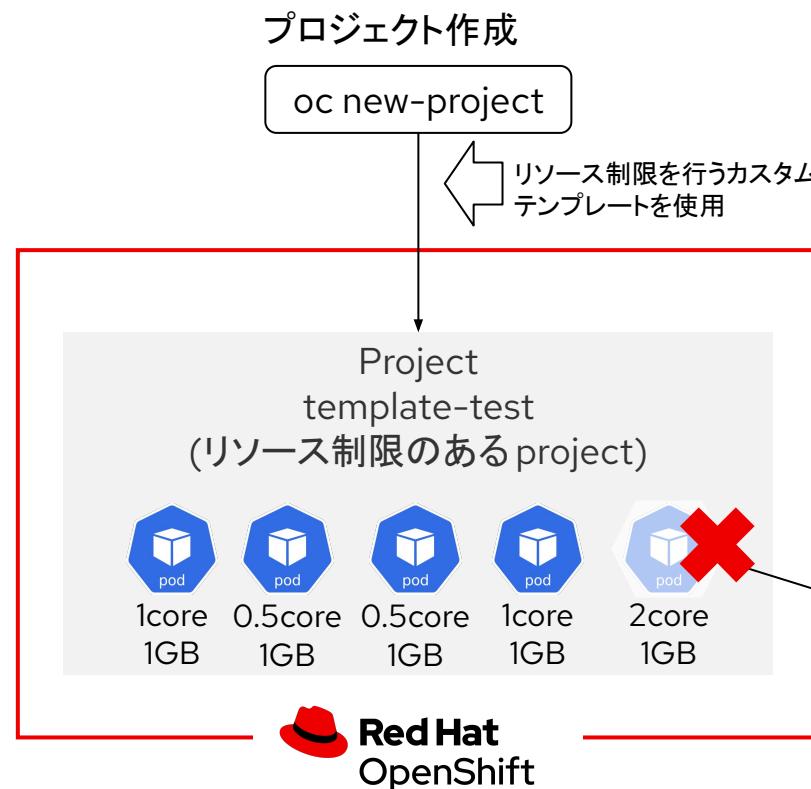
OpenShiftをインストールした時点で、クラスタに関する監視はCluster Monitoring(Prometheus)によって設定済みです。



# プロジェクト・リクエスト・テンプレートとクオータ / 制限

プロジェクト内で使用するオブジェクト数やリソース(CPU,メモリ等)を制限するテンプレートを設定します

oc new-projectを利用するとデフォルトのTemplateを使用してプロジェクトを作成する(デフォルトではリソース制限なし)



## ResourceQuota

プロジェクトごとの総リソース消費量を制限

```

- apiVersion: v1
kind: ResourceQuota
metadata:
  name: ${PROJECT_NAME}-quota
spec:
  hard:
    pods: 10
    requests.cpu: 4000m
    requests.memory: 8Gi
    resourcequotas: 1
    requests.storage: 50Gi
    persistentvolumeclaims: 5
  
```

作成可能なPodの数

CPU総量

メモリ総量

ストレージ総量

PVC数

ResourceQuotaの制約で  
CPUは合計4coreまでなので追加で  
2coreのPodはデプロイ不可

## LimitRange

プロジェクト内のリソース毎の消費量を指定

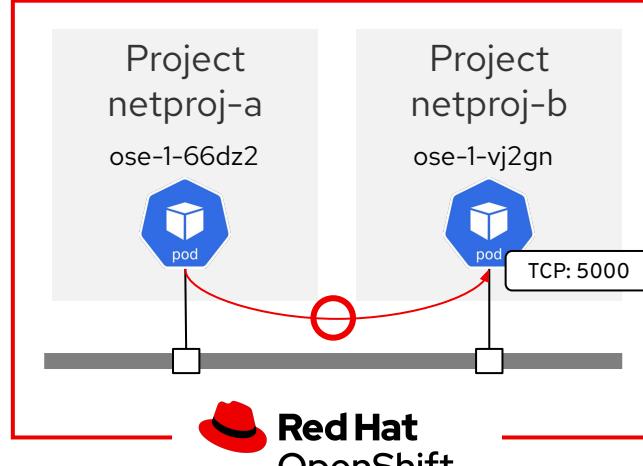
```

- apiVersion: v1
kind: LimitRange
metadata:
  name: ${PROJECT_NAME}-limits
  creationTimestamp: null
spec:
  limits:
  - type: Container
    max: limitやrequestに指定できる最大値
    cpu: 4000m
    memory: 1024Mi
    min: limitやrequestに指定できる最小値
    cpu: 10m
    memory: 5Mi
    default: 指定がない場合の最大値
    cpu: 4000m
    memory: 1024Mi
    defaultRequest: 指定がない場合の最小値
    cpu: 100m
    memory: 512Mi
  
```

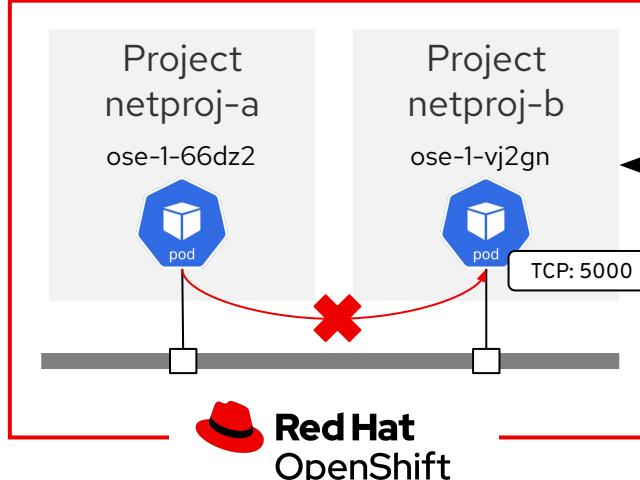
# OpenShift の Network Policy ベースの SDN

NetworkPolicyカスタムリソースを利用してプロジェクト間のネットワークを分離することができます

NetworkPolicyカスタムリソースなし  
デフォルトの状態ではすべての Pod間通信が可能



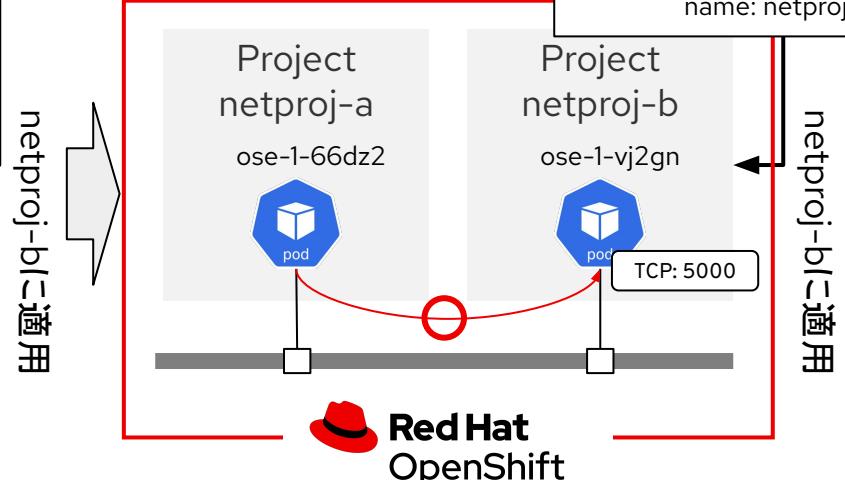
NetworkPolicyカスタムリソース  
上記の設定は ALL Denyに相当



```
kind: NetworkPolicy
apiVersion: networking.k8s.io/v1
metadata:
  name: deny-by-default
spec:
  podSelector: すべてのPodが対象
  ingress: []
```

project-aからの TCP5000を許可

```
kind: NetworkPolicy
apiVersion: networking.k8s.io/v1
metadata:
  name:
allow-tcp-5000-from-netproj-a-namespace
spec:
  podSelector:
    matchLabels:
      run: ose
      run: oseというlabelにマッチするPod
  ingress:
    - ports:
        - protocol: TCP
        port: 5000
        name: netproj-a
      from:
        - namespaceSelector:
            matchLabels:
              name: netproj-a
```



netproj-bに適用

# Projectのセルフプロビジョニングの無効化

- ・デフォルトでは認証済みのユーザはプロジェクトを作成することが可能です
- ・プロジェクトのセルフプロビジョニングを無効化することができます

## ClusterRoleBinding

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  annotations:
    rbac.authorization.kubernetes.io/autoupdate: "true"
  creationTimestamp: "2022-06-07T02:52:33Z"
  name: self-provisioners
  resourceVersion: "9988"
  uid: 24f23fdd-4a10-40f2-8ca2-0068d4b51a8a
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: self-provisioner
subjects:
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: system:authenticated:oauth
```

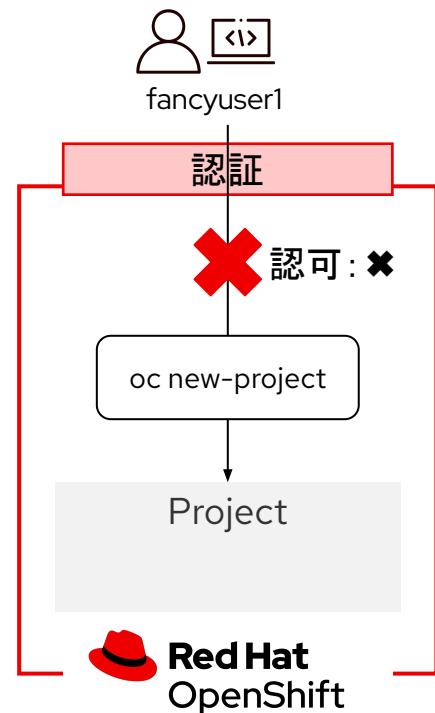
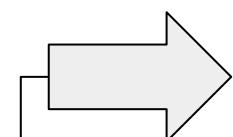
認証済みのユーザが属するグループは、  
self-provisioners roleにバインドされているた  
めセルフプロビジョニングが可能

## ClusterRoleBinding

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  annotations:
    rbac.authorization.kubernetes.io/autoupdate: "false"
  creationTimestamp: "2022-06-07T02:52:33Z"
  name: self-provisioners
  resourceVersion: "66495"
  uid: 24f23fdd-4a10-40f2-8ca2-0068d4b51a8a
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: self-provisioner
```

oc patchコマンドで設定変更

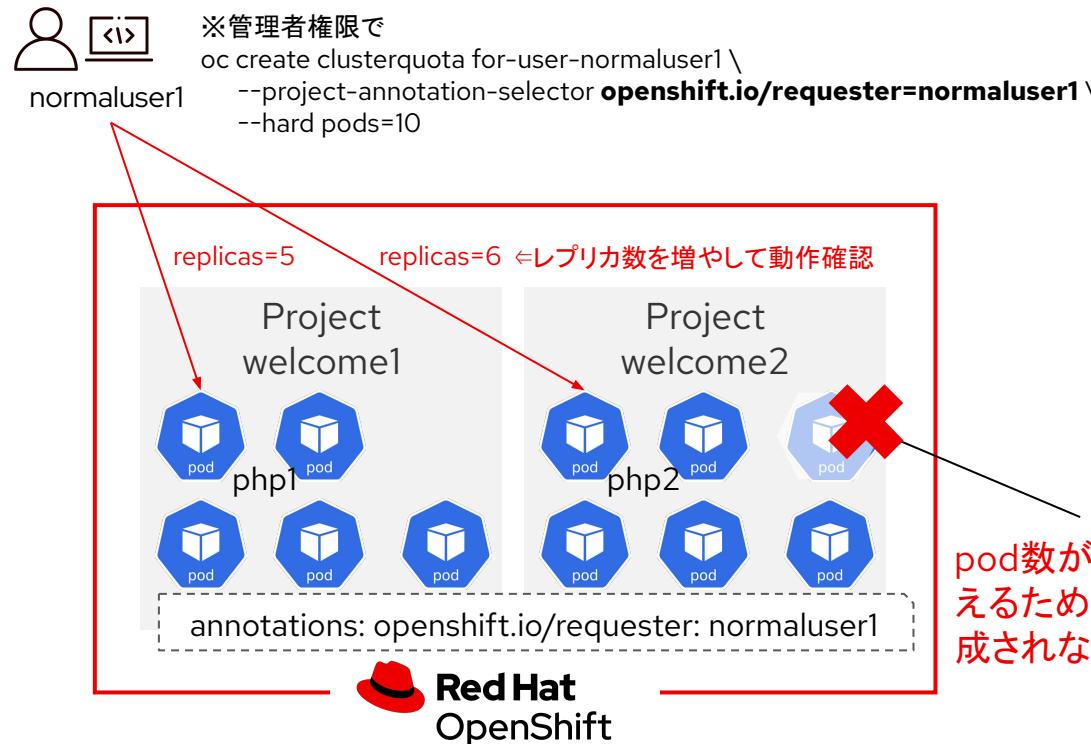
```
oc patch clusterrolebinding.rbac self-provisioners -p '{"subjects": null}'
oc patch clusterrolebinding.rbac self-provisioners -p '{"metadata": {"annotations": {"rbac.authorization.kubernetes.io/autoupdate": "false" }}}'
```



# クラスタリソースのクオータ

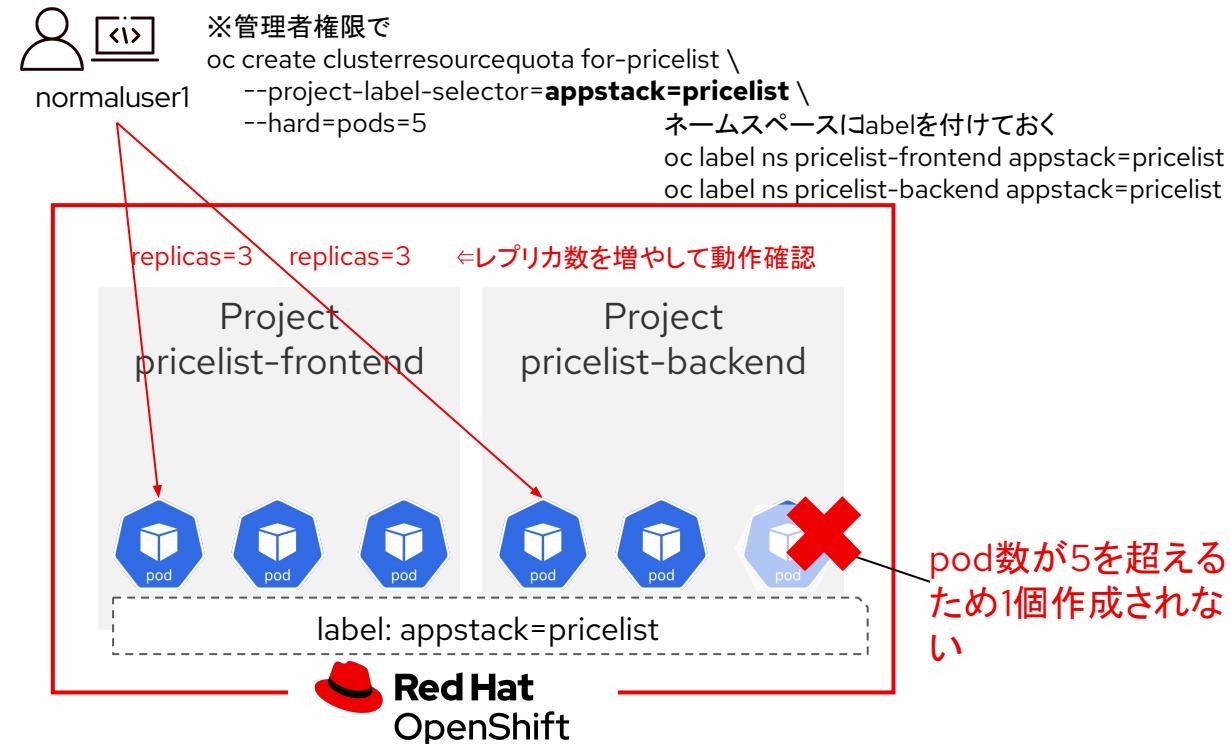
- ・特定のユーザにクオータを設定します

例) 特定のユーザに対して Pod数を10以下に制限する  
 clusterresourcequotaを作成します  
[openshift.io/requester](https://openshift.io/requester)のannotationキーを利用します



- ・labelを利用してアプリケーションにクオータを設定します

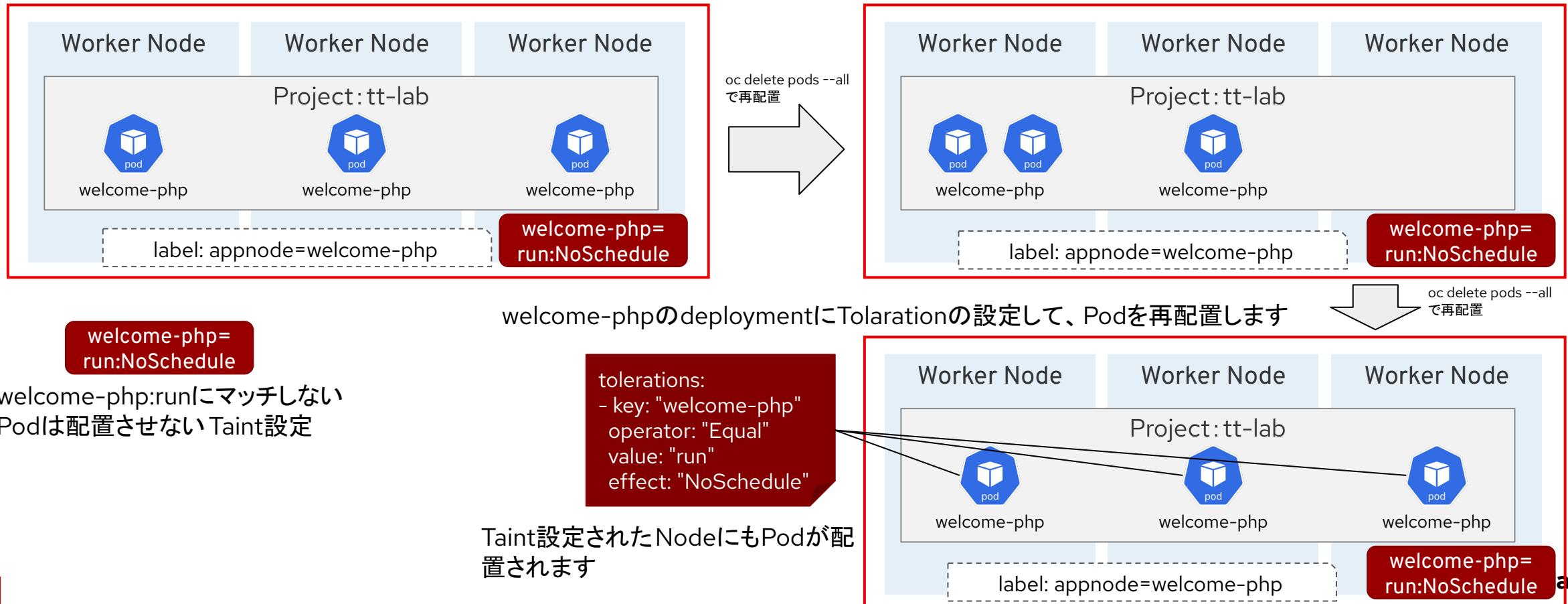
例) 特定のアプリに対して Pod数を5以下に制限する  
 clusterresourcequotaを作成します  
 labelを利用してします



# Taint と Toleration

Taint と Toleration は、協調して動作し、ワークロードが他のノードにスケジュールされないようにします

1つのNodeにTaint設定を行い、Podを再配置すると、Tolerationの設定がないため、TaintされたNodeではPodが起動しません



# Module 2

Red Hat Advanced Cluster Security for Kubernetes

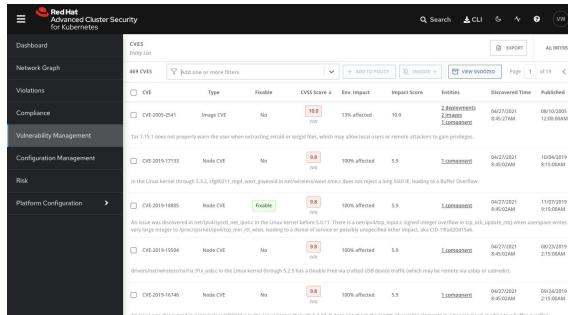
# Red Hat Advanced Cluster Security for Kubernetes

クラウド・ワークロード・プロテクション・プラットフォームとクラウド・セキュリティ・ポスマネジメントにより"シフトレフト"を可能にします

## シフトレフト

### サプライチェーンのセキュリティ

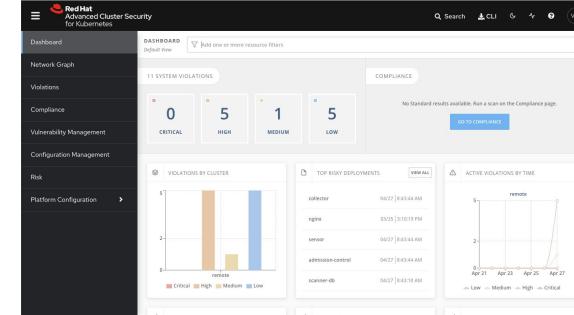
スキャンとコンプライアンスを開発に拡張する(DevSecOps)



## Kubernetesセキュリティ状態管理(KSPM)

### セキュアなインフラ

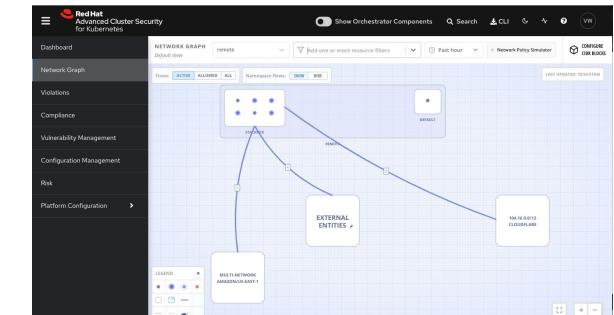
組み込みのKubernetes CSPMを活用して、リスクのある構成を特定し、修正する



## クラウドワークロード保護(CWPP)

### セキュアなワークロード

ワークロード保護に対する "zero-trust execution" アプローチの維持と実施



# Red Hat Advanced Cluster Security: ユースケース

アプリケーションのライフサイクル全体にわたるセキュリティ



## 脆弱性管理

イメージと実行中のコンテナにおける既知の脆弱性から身を守る



## ネットワークセグメンテーション

各アプリケーションのネットワーク分離とアクセス制御の適用と管理



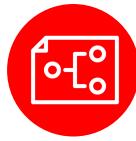
## セキュリティ構成管理

デプロイがセキュリティのベストプラクティスに従って構成されていることを確認する



## コンプライアンス

契約上および規制上の要件を満たし、それに対する監査を容易に行う



## リスクプロファイリング

OpenShift クラスターと Kubernetes クラスター全体のセキュリティ問題の優先順位を決めるためのコンテキストを得る



## 検知と対応

環境内のアクティブな脅威に対処するためのインシデントレスポンスの実施

# ダッシュボード

Red Hat Advanced Cluster Security for Kubernetes

Dashboard      1 Cluster      5 Nodes      336 Violations      214 Deployments      168 Images      820 Secrets      Last updated 2023/8/23 at 19:00

Network Graph

Violations

Compliance

Vulnerability Management (2.0)

Vulnerability Management (1.0)

Configuration Management

Risk

Platform Configuration

Dashboard

Review security metrics across all or select resources

Resources: All clusters      All namespaces

336 policy violations by severity

97	53	179	7
Low	Medium	High	Critical

Most recent violations with critical severity

Violation	Image	Timestamp
Log4Shell: log4j Remote Code Execution ...	spring4shell-app	08/23/2023   11:50:44AM
Spring4Shell (Spring Framework Remote ...	spring4shell-app	08/23/2023   11:50:44AM
Apache Struts: CVE-2017-5638	visa-processor	08/23/2023   11:50:36AM

Images at most risk

Image	Risk priority	Critical CVEs	Important CVEs
rhel8/postgresql-13	1	0 fixable	8 fixable
library/centos	1	0 fixable	40 fixable
rhacs-demo/visa-processor	2	54 fixable	93 fixable
rhacs-demo/asset-cache	2	54 fixable	93 fixable
rhacs-demo/backend-atlas	2	54 fixable	93 fixable
rhacs-demo/asset-cache	2	22 fixable	66 fixable

Deployments at most risk

Deployment	Resource location	Risk priority
visa-processor	in "production / payments"	1
dashboard	in "production / lab-ocp-cns"	2
ovnkube-node	in "production / openshift-ovn-kubernetes"	3
backend-atlas	in "production / backend"	4

100 Aging images

Feedback

- 環境全体のセキュリティ状態の概要を確認することができる
- ポリシー違反
- リスクの高いイメージやデプロイメント
- 古いイメージ
- コンプライアンス遵守状況

# 脆弱性管理ダッシュボード

The screenshot displays the Red Hat Advanced Cluster Security for Kubernetes Vulnerability Management Dashboard. The left sidebar includes links for Dashboard, Network Graph, Violations, Compliance, Vulnerability Management (2.0), Vulnerability Management (1.0), Configuration Management, Risk, and Platform Configuration. The main area features several data visualizations:

- Top risky deployments by CVE count & CVSS score:** A scatter plot showing Weighted CVSS Score (Y-axis, 5 to 10) versus Critical Vulnerabilities & Exposures (X-axis, 0 to 1,400). Data points are colored by severity: Low (blue), Medium (yellow), High (orange), and Critical (grey).
- Top Riskiest Images:** A table listing the top 8 riskiest Docker images with their respective CVE counts and fixable status. The table includes columns for Image Name, CVE Count, and Fixable Status.
- Recently Detected Image Vulnerabilities:** A table listing recent vulnerabilities detected across images, categorized by CVE ID, CVSS score, and environment impact.
- Most Common Image Vulnerabilities:** A table listing the top 8 most common image vulnerabilities, ordered by CVSS score.
- Clusters With Most Orchestrator & Istio Vulnerabilities:** A table showing clusters with the highest number of vulnerabilities, currently showing 1 production cluster with 0 CVEs.

- ・CVE, CVSSスコアに基づくリスクのあるデプロイメントやリスクの高いイメージの一覧を表示

- ・該当するデプロイメントのサマリー表

- ・CVE情報、FIXの有無など関連する情報に容易に到達できる

# 脆弱性管理(イメージサマリー)

The screenshot shows the Red Hat Advanced Cluster Security for Kubernetes interface. The left sidebar includes sections for Dashboard, Network Graph, Violations, Compliance, Vulnerability Management (2.0), Vulnerability Management (1.0) (selected), Dashboard, Risk Acceptance, Reporting, Configuration Management, Risk, and Platform Configuration. The main content area displays an image summary for the image `quay.io/rhacs-demo/visa-processor:latest-v2`. It includes a warning message about stale OS CVE data. Below this, there are sections for Details & Metadata (Risk priority: 2, Top CVSS: 10.0, CVSS Version: V3, SHA: sha256:15670ce73d09bb93a93bf81e82740a516145269f99126fb9442402b0d24c4c23, Created: 05/16/2020 | 10:18:31AM, Scanner: Stackrox Scanner, Scan time: 08/23/2023 | 3:21:33PM, Image OS: debian:8), CVEs by CVSS Score (a pie chart showing 532 total vulnerabilities: 71 Critical, 145 Important, 140 Moderate, 176 Low), and Top Riskiest Image Components (a list of components with their CVE counts and fixability status). A sidebar on the right shows Related entities: Matches (1 DEPLOYMENT), Contains (170 IMAGE COMPONENTS, 532 IMAGE CVES), and a Feedback button.

- ・イメージ全体の重要なセキュリティの詳細が、コンポーネントへのリンクとともに表示される

# リスクプロファイリング

The screenshot shows the Red Hat Advanced Cluster Security for Kubernetes dashboard. The left sidebar includes links for Dashboard, Network Graph, Violations, Compliance, Vulnerability Management (2.0), Vulnerability Management (1.0), Configuration Management, Risk (selected), and Platform Configuration. The main area displays a table of 91 deployments with columns for Name, Created, Cluster, Namespace, and Priority. A detailed view for the 'log4shell-app' deployment is open, showing Risk Indicators, Deployment Details, and Process Discovery tabs. The Risk Indicators tab lists policy violations related to Log4Shell and Spring4Shell vulnerabilities, fixable severity, resource requests, and pod service account tokens. The Image Vulnerabilities tab indicates 109 CVEs between low and critical severity for the image 'quay.io/mfoster/log4shell-demo:0.1.0'. The Service Configuration tab shows no dropped capabilities.

Name	Created	Cluster	Namespace	Priority
visa-processor	08/23/2023   11:50:28AM	production	payments	1
dashboard	08/23/2023   11:10:20AM	production	lab-ocp-cns	2
backend-atlas	08/23/2023   11:49:52AM	production	backend	4
asset-cache	08/23/2023   11:50:00AM	production	frontend	5
log4shell-app	08/23/2023   11:50:12AM	production	log4shell	6
spring4shell-app	08/23/2023   11:50:33AM	production	spring4shell	6
jump-host	08/23/2023   11:50:23AM	production	operations	7
reporting	08/23/2023   11:50:15AM	production	medical	9
yelb-appserver	08/23/2023   11:50:45AM	production	yelb	10
yelb-ui	08/23/2023   11:50:36AM	production	yelb	11
monitor	08/23/2023   11:50:09AM	production	frontend	13
mastercard-processor	08/23/2023   11:50:30AM	production	payments	14
yelb-db	08/23/2023   11:50:42AM	production	yelb	16
redis-server	08/23/2023   11:50:39AM	production	yelb	16

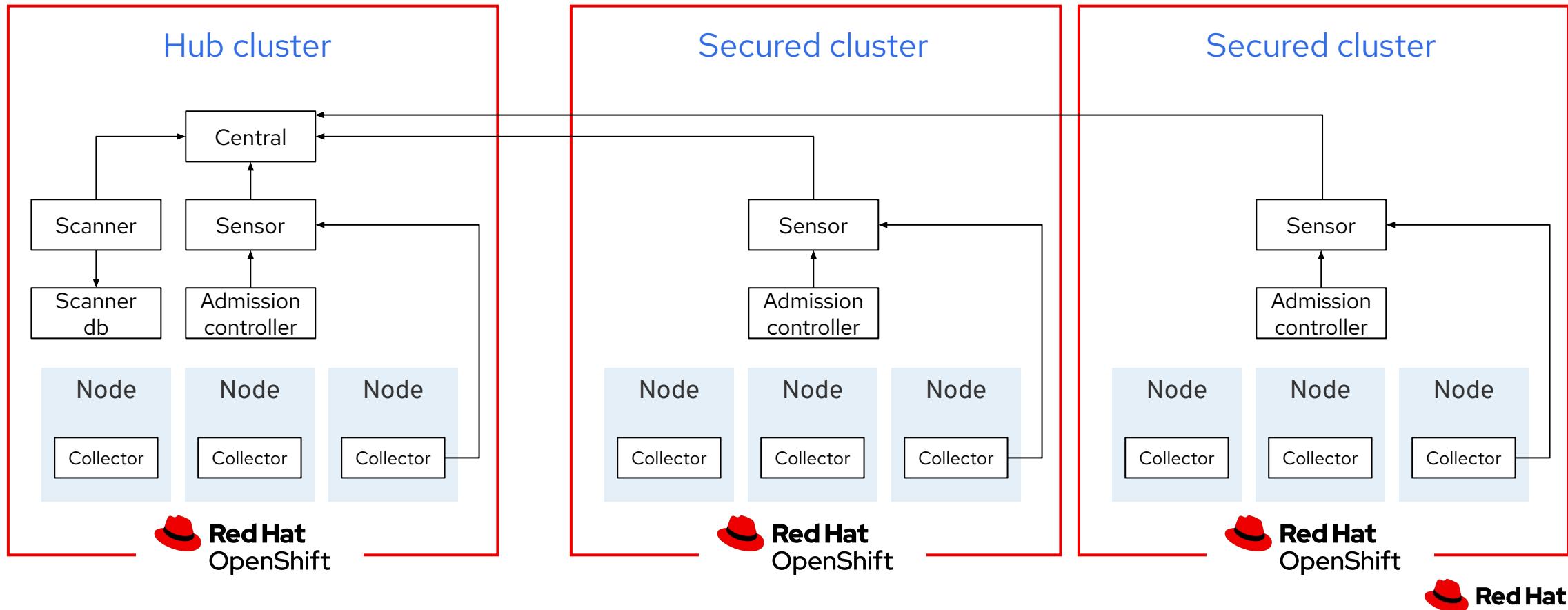
- ・リスクのあるデプロイメントをプライオリティをつけて表示
- ・プライオリティに影響する要素
  - ・ポリシー違反
  - ・イメージの脆弱性
  - ・サービス構成
  - ・サービス到達性
  - ・攻撃者にとって有用なコンポーネント
  - ・イメージに含まれるコンポーネントの数
  - ・イメージの鮮度
  - ・RBACの構成等

# Red Hat Advanced Cluster Security for Kubernetes

Hub cluster : Centralが稼働しているクラスタ

Secured cluster: 保護対象のクラスタ

※本ワークショップではhub clusterとSecured clusterが同一クラスタです

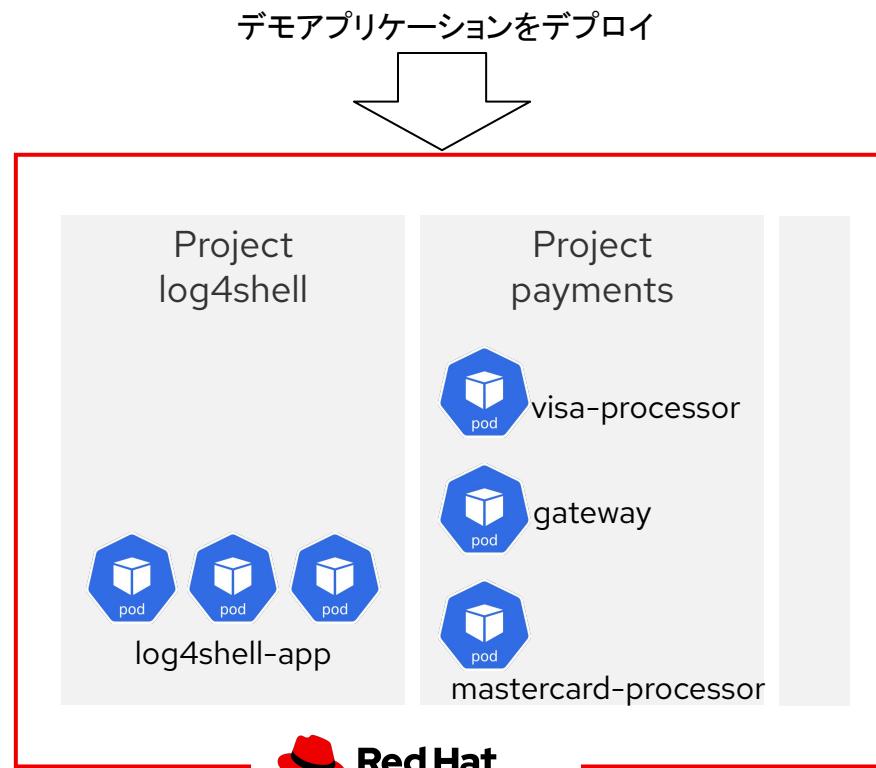


# Red Hat Advanced Cluster Security for Kubernetes

## 演習内容

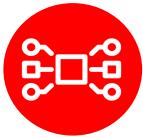
# ACSによる脆弱性のスキャン

Advanced Cluster Security (ACS) ダッシュボードで修正可能な脆弱性(log4shellなど)を正確に特定し、トリアージする方法を学びます



- ・デモアプリケーションをデプロイします
- ・ACS ダッシュボードへアクセスします
- ・ユーザーインターフェースの操作方法
- ・脆弱性管理ダッシュボードを使用して、クラスタ内的一般的な脆弱性と問題点を分析します
- ・得られた知識を使用して、クラスタ内のlog4shellエクスプロイトを検出します
- ・log4shellの脆弱性を含む脆弱なアプリケーションのデプロイをブロックするポリシーを作成します

# log4shellの脆弱性の発見と解析



## log4shellとは

Log4Shellは、アプリケーションのエラーメッセージを記録するための一般的なJavaライブラリであるApache Log4j 2のソフトウェア脆弱性です。CVE-2021-44228として公開されているこの脆弱性は、デバイスが特定のバージョンのLog4j 2を実行している場合、リモート攻撃者がインターネット上のデバイスを制御することを可能にします。



## なぜ、脆弱性の発見が重要なのか？

セキュリティチームや運用チームには、深刻度の異なる脆弱性に対応するためのプロセスや手順を用意しておくことが不可欠です。これらは、あらゆる脆弱性の正確な評価とともに、深刻度の高い脆弱性には迅速な対応に値する緊急性をもって対処し、深刻度の低い脆弱性にはより慎重な態度で対処することを可能にします。



## 検出と対応

コンテナの脆弱性を迅速に検出し、対応することができれば、チームはより早くサービスの開発に戻ることができます。

重大な脆弱性が発生した場合、脆弱性の範囲にアクセスし、トリアージする間、ほとんどのデプロイは保留されます。

# visa-prosessorの脆弱性の発見と解析

**脆弱性ダッシュボード**

**Top Riskiest Images → View All**

**Top Riskiest Images → visa-processorを検索**

**CVE情報**

**image:visa-processorに関する情報**



Red Hat

## log4shellの脆弱性の発見と解析

The screenshot shows the Red Hat Advanced Cluster Security for Kubernetes dashboard. A red box highlights the 'Top Riskiest Images' section, which displays a table of 166 images. The table includes columns for Image, Image CVEs, Top CVSS, Created, Scan Time, Image OS, Image Status, Entities, and Risk Priority. A specific row for 'quay.io/log4shell-demo:log4shell-demo:0.1.0' is highlighted with a red box. A red arrow points from this row to a detailed view of the image information.

Top Riskiest Images → View All

Top Riskiest Images → CVE-2021-44228を検索

image:log4shell-demoに関する情報

# デプロイメントvisa-prosessorとlog4shellのリスク指標

The screenshot shows the Red Hat Advanced Cluster Security for Kubernetes dashboard. The left sidebar has a 'Risk' section selected. The main area displays a table of 91 deployments. Two specific rows are highlighted with red boxes and labeled with red text: 'visa-prosessor' and 'log4shell-app'. A vertical red bar on the right side contains the word 'Feedback'.

Name	Created	Cluster	Namespace	Priority
visa-prosessor	11/12/2023   11:11:36AM	production	payments	2
log4shell-app	11/12/2023   11:11:20AM	production	log4shell	6
visa-prosessor	11/12/2023   9:21:24AM	production	lab-ocp-cns	1
backend-atlas	11/12/2023   11:10:59AM	production	backend	3
asset-cache	11/12/2023   11:11:09AM	production	frontend	4
yelb-ui	11/12/2023   11:11:44AM	production	yelb	5
spring4shell-app	11/12/2023   11:11:41AM	production	spring4shell	6
reporting	11/12/2023   11:11:23AM	production	medical	9
yelb-appserver	11/12/2023   11:11:53AM	production	yelb	10
monitor	11/12/2023   11:11:17AM	production	frontend	11
yelb-db	11/12/2023   11:11:50AM	production	yelb	12
redis-server	11/12/2023   11:11:47AM	production	yelb	12
search-postgres	11/12/2023   9:38:40AM	production	open-cluster-management	14
tekton-operator-webhook	11/12/2023   9:25:09AM	production	openshift-operators	15
wordpress	11/12/2023   11:11:14AM	production	frontend	17

# ACSポリシー

Policies > Log4Shell: log4j Remote Code Execution vulnerability

Log4Shell: log4j Remote Code Execution vulnerability Enabled ポリシーが有効になっている

Actions ▾

Policy behavior

Policy details

Severity	Critical	どの段階で評価するか？
Categories	Vulnerability Management	検出した場合の振る舞い
Type	System default	
Description	Alert on deployments with images containing the Log4Shell vulnerabilities (CVE-2021-44228 and CVE-2021-45046). There are flaws in the Java logging library Apache Log4j in versions from 2.0-beta9 to 2.15.0, excluding 2.12.2.	
Rationale	These vulnerabilities allows a remote attacker to execute code on the server if the system logs an attacker-controlled string value with the attacker's JNDI LDAP server lookup.	
Guidance	Update the log4j library to version 2.16.0 (for Java 8 or later), 2.12.2 (for Java 7) or later. If not possible to upgrade, then remove the JndiLookup class from the classpath: zip -q -d log4j-core-*.jar org/apache/logging/log4j/core/lookup/JndiLookup.class	
MITRE ATT&CK		
Policy has no MITRE ATT&CK vectors		

Lifecycle stages	Build, Deploy
Event source	N/A
Response	Inform

Policy criteria 判断基準

1

CVE	CVEがCVE-2021-44228または CVE-2021-45046に該当した場合
CVE identifier is:	
CVE-2021-44228	
— OR —	
CVE-2021-45046	



# ACSポリシーの複製と編集

Policies > Log4Shell: log4j Remote Code Execution vulnerability (COPY)

## Log4Shell: log4j Remote Code Execution vulnerability (COPY) Enabled

Actions ▾

### Policy details

Severity	<span>Critical</span>
Categories	Vulnerability Management
Type	User generated
Description	Alert on deployments with images containing the Log4Shell vulnerabilities (CVE-2021-44228 and CVE-2021-45046). There are flaws in the Java logging library Apache Log4j in versions from 2.0-beta9 to 2.15.0, excluding 2.12.2.
Rationale	These vulnerabilities allows a remote attacker to execute code on the server if the system logs an attacker-controlled string value with the attacker's JNDI LDAP server lookup.
Guidance	Update the log4j library to version 2.16.0 (for Java 8 or later), 2.12.2 (for Java 7) or later. If not possible to upgrade, then remove the JndiLookup class from the classpath: zip -q -d log4j-core-*.jar org/apache/logging/log4j/core/lookup/JndiLookup.class
MITRE ATT&CK	
Policy has no MITRE ATT&CK vectors	

### Policy behavior

Lifecycle stages	Build, Deploy	BuildとDeployの段階
Event source	N/A	
Response	Enforce	強制的にブロック
Enforcement	Deploy	Deploy時

### Policy criteria

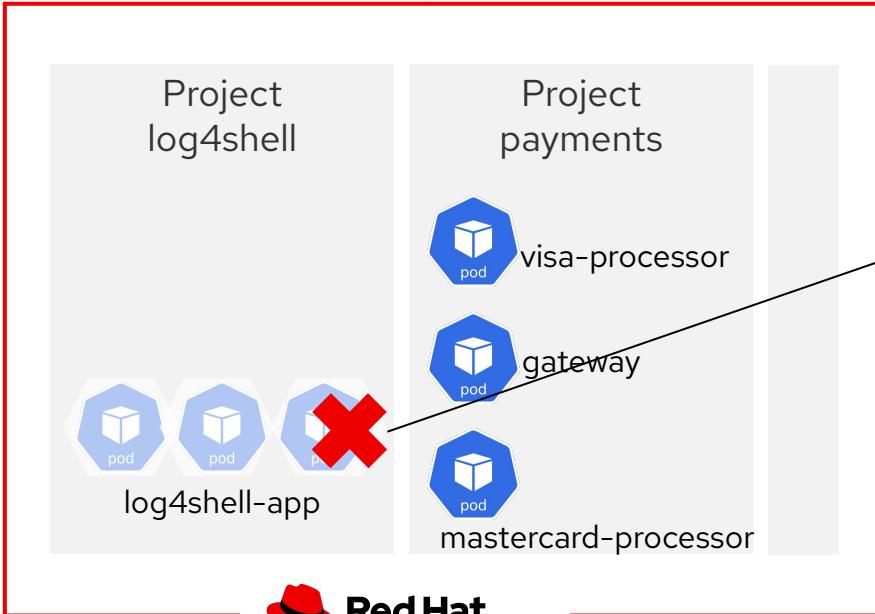
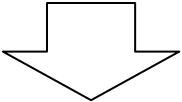
1	CVEがCVE-2021-44228または CVE-2021-45046に該当した場合
CVE	CVE identifier is:
CVE-2021-44228	— or —
CVE-2021-45046	



## ACSポリシーの動作確認

Log4Shell: log4j Remote Code Execution vulnerability (COPY)が有効になっているため、該当する脆弱性を含むデプロイメントはブロックされて Podは起動しません

log4shell-appを新規にデプロイ



Red Hat  
OpenShift

**ACSコンソールの[Violations]**

**Violations > Log4Shell: log4j Remote Code Execution vulnerability (COPY)**

**Log4Shell: log4j Remote Code Execution vulnerability (COPY)  
in "log4shell-app" deployment**

Violation Enforcement Deployment Policy

Enforcement on this deployment is enabled for this policy

Deployment data was evaluated against this security policy. Deployment scaled to 0 replicas in response to this policy violation.

If the enforcement action is being a

**OpenShift Webコンソールの[Topology]**

Project: log4shell Application: All applications

Topology

Pods

Services

Routes

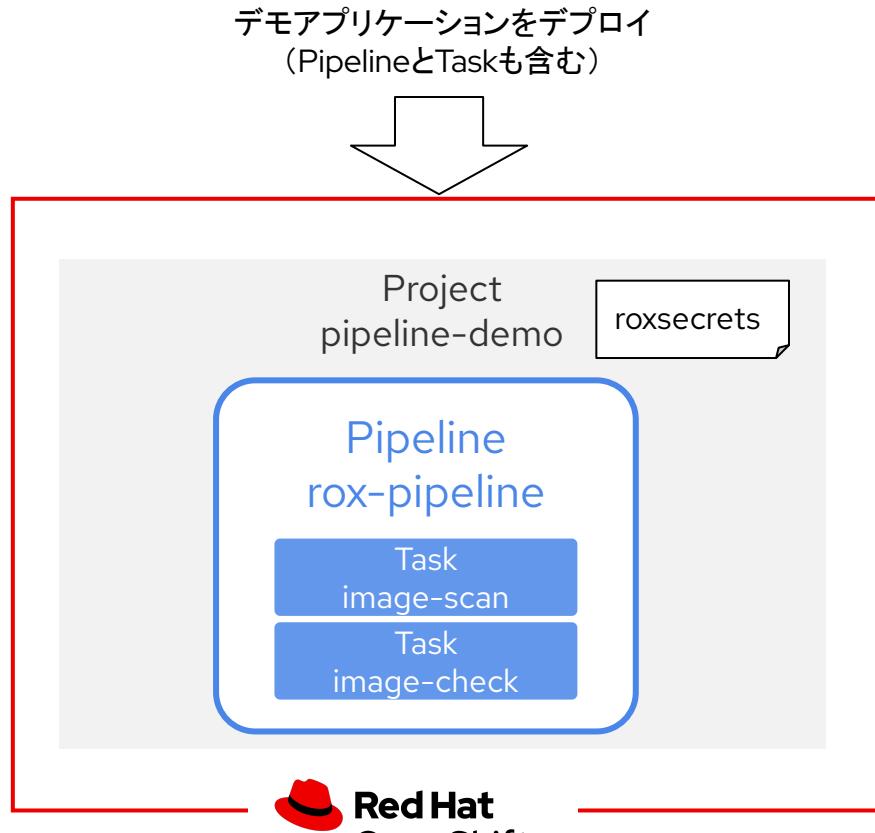
No Pods found for this resource.

No Services found for this resource.

No Routes found for this resource.

# DevSecOps

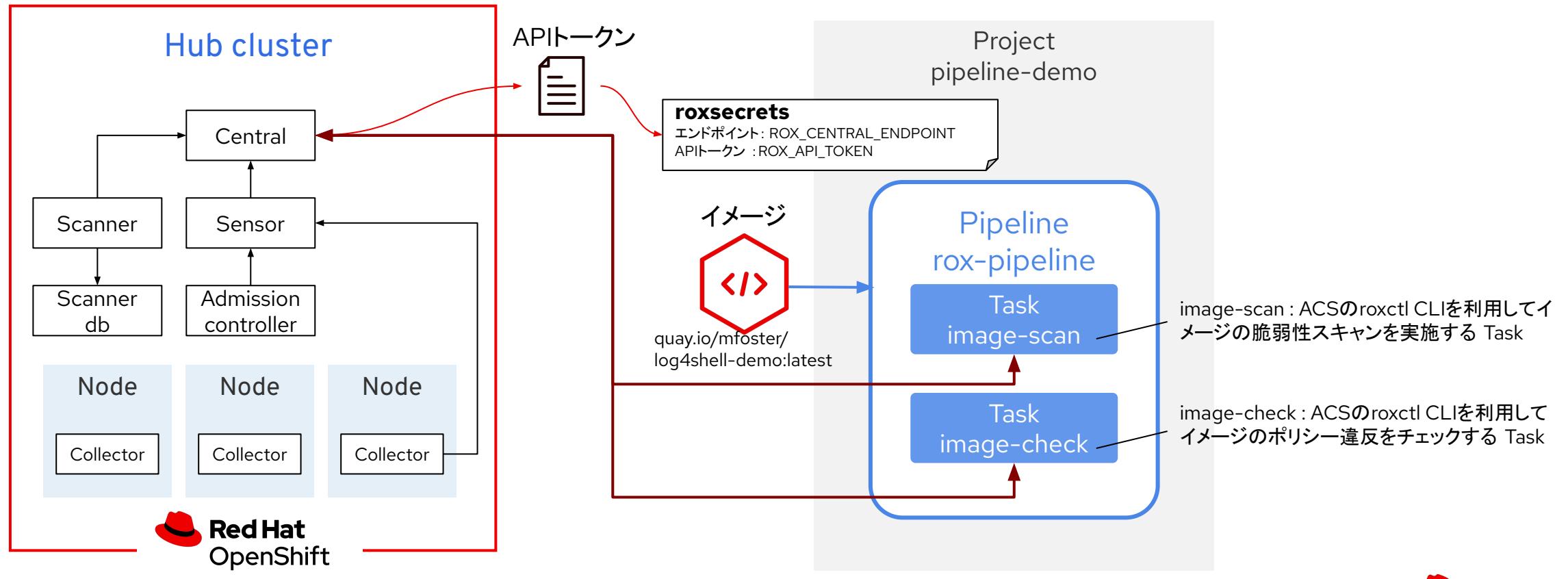
OpenShift PipelinesのワークフローにACS脆弱性ポリシーを実装します



1. デモアプリケーションをデプロイします  
(PipelineとTask、Secretが作成されます)
2. OpenShiftのパイプラインがACSセントラルと通信するために  
必要な認証を設定します
3. パイプラインを実行し、脆弱性検知によりパイプラインが  
中断されることを確認します

# DevSecOps

ACSのroxctl CLIを利用するTaskをPipelineに組み込むことにより、イメージスキャンやポリシーチェックを自動的に行うことができ、脆弱性のあるアプリケーションやイメージのビルトやデプロイを中断させることができます



# DevSecOps

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

Project: pipeline-demo ▾

Administrator ▾

Home ▶

Operators ▶

Workloads ▶

Networking ▶

Storage ▶

Builds ▶

Pipelines ▾

- Pipelines
- Tasks
- Triggers

rox-pipeline-odah4n

PLR rox-pipeline-odah4n Failed

Actions ▾

Details YAML TaskRuns Parameters Logs Events

PipelineRun details

image-scan 1/1

image-check 0/1

Pipelineが中斷される

Name: rox-pipeline-odah4n Status: Failed

Namespace: NS pipeline-demo Message: Failure on task - check logs for details.

Labels: tekton.dev/pipeline=rox-pipeline

Annotations: 2 annotations

Created at: ~

Log snippet:

```
WARN: A total of 6 policies have been violated
ERROR: failed policies found: 1 policies violated that are failing the check
ERROR: Policy "Fixable Severity at least Important" - Possible remediation: "Use your package manager to update to a fixed version in future builds or speak with your security team to mitigate the vulnerabilities."
ERROR: checking image failed after 3 retries: failed policies found: 1 policies violated that are failing the check
```

# Module 3

Red Hat Advanced Cluster Management for Kubernetes



# Red Hat

## Advanced Cluster Management for Kubernetes

### 操作とメンテナンスの簡素化

単一のコンソールで、表示、管理、操作、問題解決のすべてを行うことができる

### OpenShift上で動作

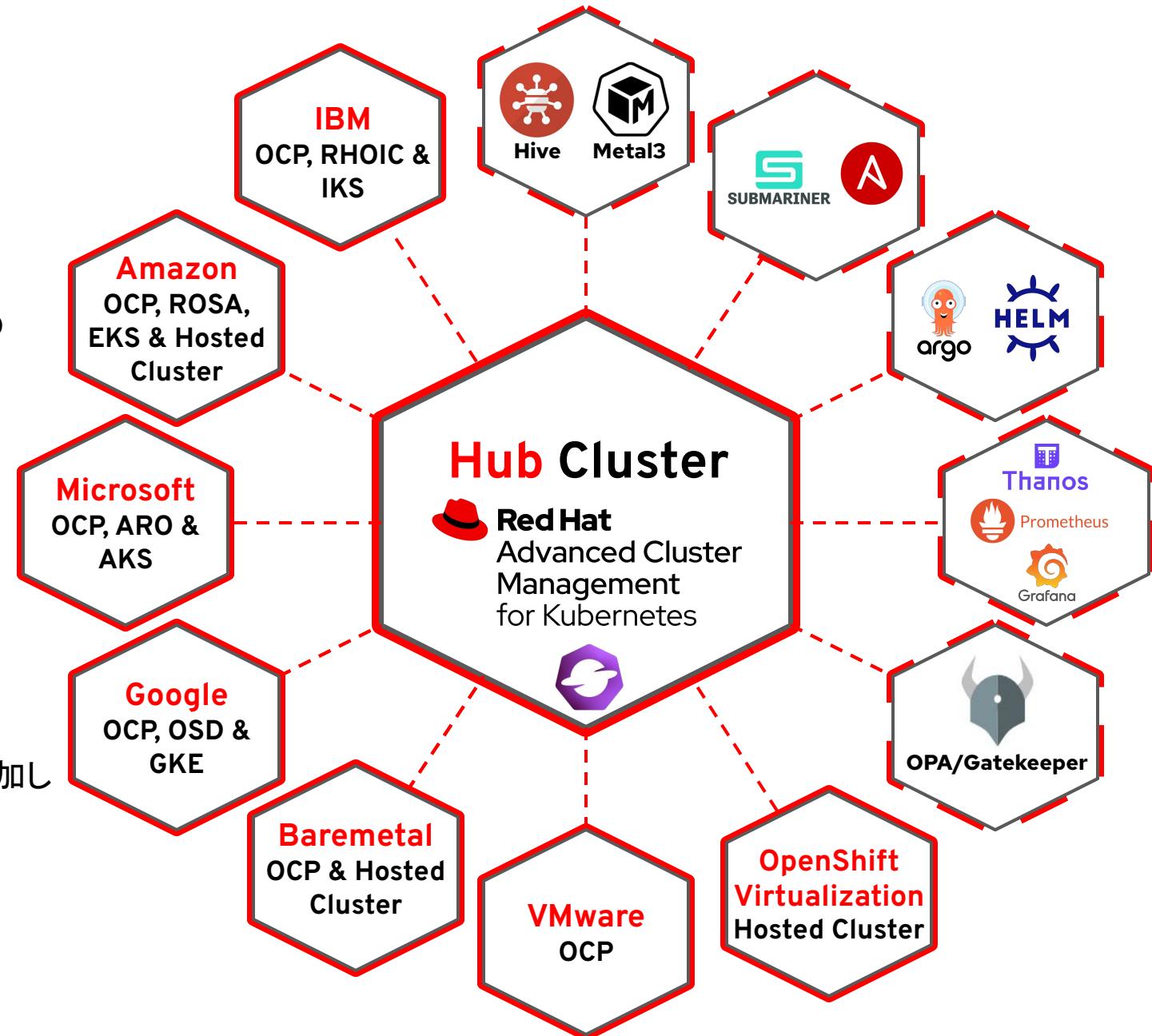
他のKubernetesアプリと同様に、OpenShiftクラスタ上で簡単に実行・管理できる

### ハブ・スポークアーキテクチャ

すべての設定をHubクラスタコンポーネントで管理し、Spoke Kubernetesクラスタをハブにシームレスに追加します。

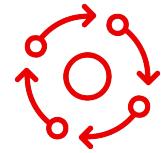
### タイトな統合

RHACMには豊富なAPI、アドオンが付属しており、他の主要な企業ツールと統合することができる。



# Red Hat Advanced Cluster Management for Kubernetes

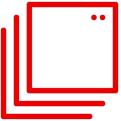
## End-to-end automation with Red Hat Ansible Automation Platform integration



マルチクラスター・ライフサイクル管理



ポリシードリブンガバナンス、リスト、コンプライアンス



高度なアプリケーション・ライフサイクル管理



健全性と最適化のためのマルチクラスター・オブザーバビリティとサーチ



相互接続のためのマルチクラスター・ネットワーキング

The screenshot displays the Red Hat Advanced Cluster Management for Kubernetes interface. The top navigation bar includes 'Red Hat OpenShift', 'All Clusters', 'Home', 'Welcome', 'Overview', 'Search', 'Infrastructure', 'Applications', 'Governance', and 'Credentials'. The 'Overview' section provides a high-level summary of clusters across various cloud providers and regions. The 'Governance' section allows users to manage policies and monitor policy violations. A detailed 'Deployment' workflow diagram illustrates the flow from Ansible job creation to deployment and replication. The 'Metrics' section provides real-time data on cluster performance, including memory usage and CPU usage.

# 統合マルチクラスタ管理

すべてのKubernetesクラスタを一元管理

The screenshot displays two main pages of the Red Hat OpenShift Multicloud Management interface:

- Overview Page:** Shows a summary of clusters across various cloud providers. The data is as follows:

Cloud Provider	Clusters
Amazon	6
VMware	1
Other	2
Microsoft	2
IBM	1
Google	1
- Clusters Page:** A detailed list of individual Kubernetes clusters. The table includes columns for Name, Namespace, Status, Infrastructure, Control plane type, Distribution version, Labels, Nodes, and Creation date. The clusters listed are:

Name	Namespace	Status	Infrastructure	Control plane type	Distribution version	Labels	Nodes	Creation date
carolina	carolina	Ready	VMware vSphere	Standalone	OpenShift 4.11.26	<a href="#">Upgrade available</a>	3	09/09/2022, 19:44:48
dev-iks-eu	dev-iks-eu	Ready	IBM Cloud	Standalone	v1.23.16+IKS		3	11/02/2022, 04:50:38
local-cluster	local-cluster	Ready	aws Amazon Web Services	Hub	OpenShift 4.11.9	<a href="#">Upgrade available</a>	10	12/11/2021, 17:33:35
migration	migration	Ready	aws Amazon Web Services	Standalone	OpenShift 4.11.26	<a href="#">Upgrade available</a>	7	08/02/2022, 18:38:25

- 複数のプライベートクラウドとパブリッククラウドにわたってKubernetesクラスタの一元的に作成、更新、削除
- OCPクラスタ管理を簡素化するためのCluster Sets & Cluster Pools の設定
- ドメイン全体であらゆるkubernetesリソースの検索、検出、変更
- 連携するドメイン全体の問題を迅速にトラブルシュートして解決

# ポリシーに基づくガバナンス、リスク、コンプライアンス

The screenshot shows the Open Cluster Management Governance interface. On the left, there's a navigation sidebar with 'Standards' (ACM-Health, CM-2 Baseline, NIST, NIST 800-53, NIST SP 800-53, NIST-CSF, generic) and a 'Create policy' button. The main dashboard displays three circular metrics: 'Policy set violations' (2), 'Policy violations' (57), and 'Clusters' (rosacluster, local-cluster, migration, sberens-aro-central). A large red circle highlights the 'Policy set violations' metric. Below the dashboard, a 'Create policy' dialog is open, showing 'Details' (Name: compliance-operator, Namespace: policies), 'Templates' (Policy templates: comp-operator-ns-2, comp-operator-operator-group-2, comp-operator-subscription-2, comp-operator-status-2), 'Placement' (Label expressions: name equals local-cluster), and 'Policy annotations' (Standards: NIST SP 800-53, Categories: CA Security Assessment and Authorization, Controls: CA-2 Security Assessments, CA-7 Continuous Monitoring). On the right, a 'Policy YAML' editor shows the generated YAML code:

```
apiVersion: policy.open-cluster-management.io/v1
kind: Policy
metadata:
  name: compliance-operator
  namespace: policies
annotations:
  policy.open-cluster-management.io/categories: CA Security Assessment and Authorization
  policy.open-cluster-management.io/standards: NIST SP 800-53
  policy.open-cluster-management.io/controls: CA-2 Security Assessments, CA-7 Continuous Monitoring
spec:
  disabled: false
  policy-templates:
    - objectDefinition:
        apiVersion: policy.open-cluster-management.io/v1
        kind: ConfigurationPolicy
        metadata:
          name: comp-operator-ns-2
        spec:
          remediationAction: inform
          severity: high
        object-templates:
          - complianceType: musthave
            objectDefinition:
              apiVersion: v1
              kind: Namespace
              metadata:
                name: openshift-compliance
    - objectDefinition:
        apiVersion: policy.open-cluster-management.io/v1
        kind: ConfigurationPolicy
        metadata:
          name: comp-operator-operator-group-2
        spec:
          remediationAction: inform
          severity: high
        object-templates:
          - complianceType: musthave
            objectDefinition:
              apiVersion: operators.coreos.com/v1
```

- セキュリティ、アプリケーション、インフラストラクチャのポリシーを一元的に設定および適用
- アプリケーションとクラスタの構成に関する詳細な監査を迅速に可視化
- Ansible Automation Platformとの統合を活用して修正アクションの実行
- GitOps統合を含む、組み込みのコンプライアンスピリシーと監査チェック
- 定義された基準に基づき、コンプライアンス状況を即座に可視化

# 高度なアプリケーションライフサイクル管理

## アプリケーションライフサイクルの簡素化

The screenshot displays the Red Hat Application R&D interface, which includes:

- Create application:** A form for entering the application name and namespace.
- Application YAML:** An editor showing the YAML configuration for an application, specifically a Subscription resource.
- Topology:** A visual representation of the application's architecture, showing relationships between Application, Subscription, Cluster, DeploymentConfig, Route, Service, and Pod components.
- Pod details:** A panel showing the details for a specific Pod named "rocketchat-db-1-g7vst". It includes information such as Namespace: rocketchat, Labels: app=rocketchat-db, deployment=rocketchat-db-1, deploymentconfig=rocketchat-db, Status: Running, and IP address: 10.0.146.124.

- ・ アプリケーションビルダーを使用してアプリケーションを簡単にデプロイ(サブスクリプション)
- ・ 複数のソース(Git/Helm/Object Storage)からアプリケーションをデプロイ
- ・ OpenShift GitOps(ArgoCD)との統合
- ・ RHACM での Argo CD アプリケーションの自動検出と視覚化
- ・ クラスタ間およびクラスタにまたがるアプリケーションの関係を迅速に視覚化

# Red Hat Advanced Cluster Management for Kubernetes

## 演習内容

# Red Hat Advanced Cluster Management for Kubernetes

マルチクラウド環境におけるKubernetesクラスタのフリートを効率的に管理する方法を学びます。RHACMダッシュボードから以下の方法を学びます

## 1. クラスター・ライフサイクルの操作

- a. クラウドインフラへのアクセスを提供するクラウドプロバイダーの認証情報を設定します
- b. AWS上にOpenShiftクラスタを作成します
- c. AWS上にSNO(Single-node OpenShift) クラスタを作成します

## 2. アプリケーションの作成と管理

- a. マネージドクラスタにアプリケーションをデプロイします
- b. application placementを理解します

## 3. ガバナンス(ACMでポリシーを作成・適用)

- a. ETCD暗号化を有効にするK8sポリシーを作成します
- b. ポリシーがinform modeまたはenforce modeに設定できることを理解します

# クラスターライフサイクルの操作

- local-clusterのコンソールをAll Clustersに切り替えます
- 本環境はAWS上に構築しているため、AWSの認証情報を設定します
- 認証情報を設定後、ウィザードに従ってクラスタを作成します

Here is some important information about your environment:

OpenShift Console: <https://console-openshift-console.apps.cluster-9fddq.9fddq.sandbox1562.opentlc.com> OpenShift API for command line 'oc' client: <https://api.cluster-9fddq.9fddq.sandbox1562.opentlc.com:6443> Download oc client from <http://mirror.openshift.com/pub/openshift-v4/clients/ocp/stable-4.12/openshift-client-linux.tar.gz>

Access the workshop at <https://dashboard-lab-ocp-cns.apps.cluster-9fddq.9fddq.sandbox1562.opentlc.com>

Login with 'kubeadmin' and 'druMK-SAcLK-8At2l-6qb7k'

Workshop may not be accessible until rollout finishes shortly.

Your RHACS console is available at:

<https://central-stackrox.apps.cluster-9fddq.9fddq.sandbox1562.opentlc.com>

RHACS portal username: admin

RHACS portal password: MjcwNzc2

Use the following credentials to deploy in the AWS sandbox account where your Hub is running.

AWS\_ACCESS\_KEY\_ID: AKIAUVI7DCRELC64VV2E +  
AWS\_SECRET\_ACCESS\_KEY: 0RXUvE6WV14IYnmq5N8oytLsNCaVBqUlBYjJVcLm +  
Top level domain: <.sandbox1562.opentlc.com> +

AWSの認証情報

You can access your bastion via SSH: ssh [demo-user@bastion.9fddq.sandbox1562.opentlc.com](ssh://demo-user@bastion.9fddq.sandbox1562.opentlc.com)

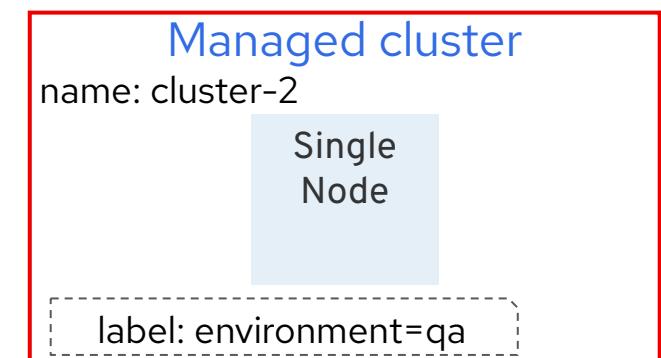
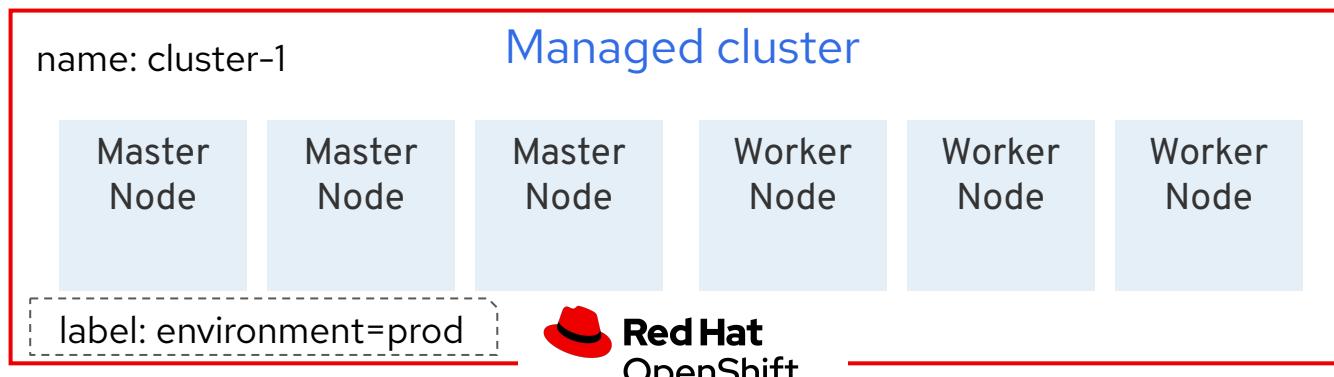
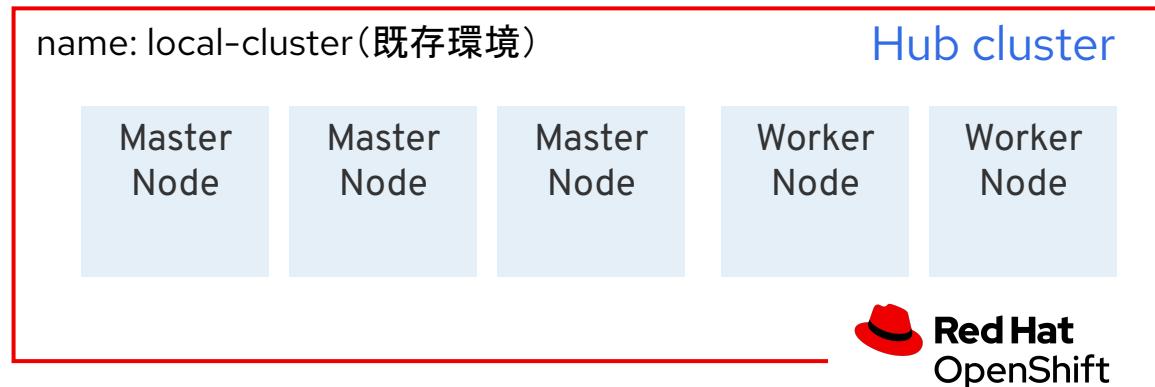
Make sure you use the username 'demo-user' and the password 'QuXuR4HNTqGn' when prompted.

You may manage your RHDP service at:

<https://demo.redhat.com/services/user-mkoshimi-redhat-com/sandboxes-gpte.ocp4-acm-acs-ops-wksp.prod>

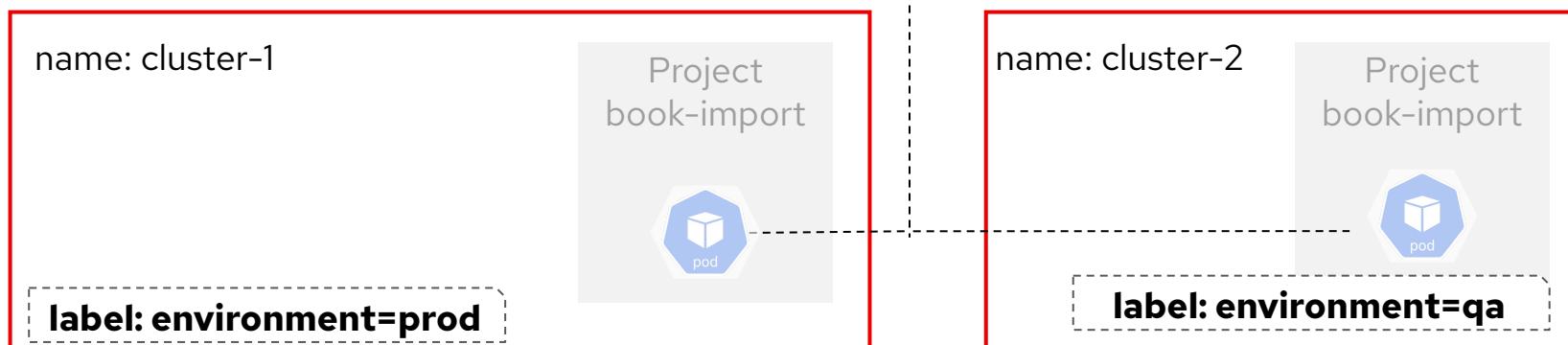
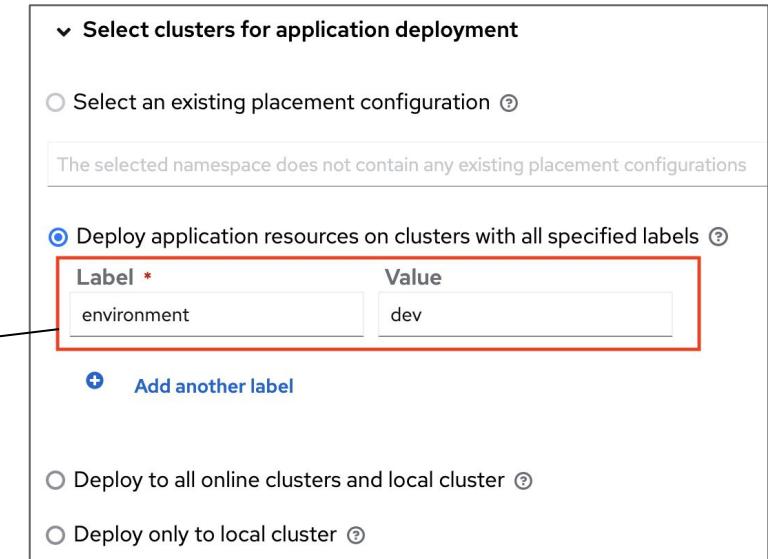
# クラスターライフサイクルの操作

- ・クラスタの構築が完了すると以下の構成になります



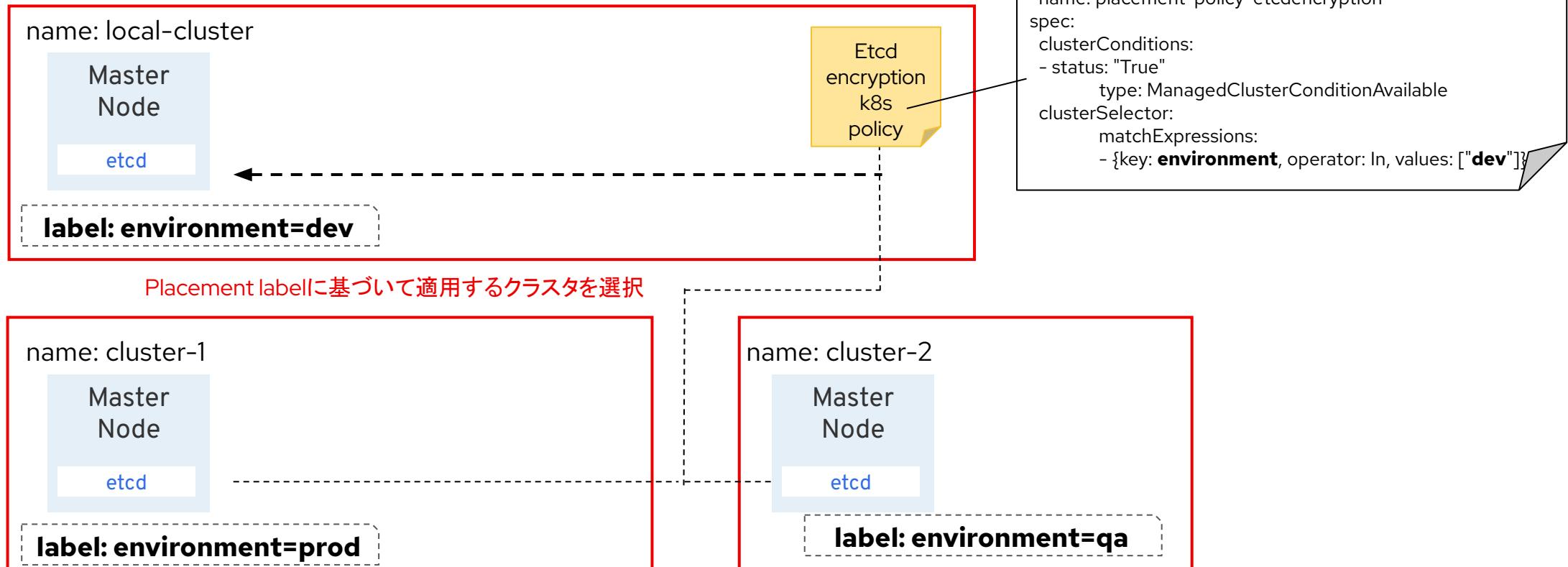
# アプリケーションの作成と管理

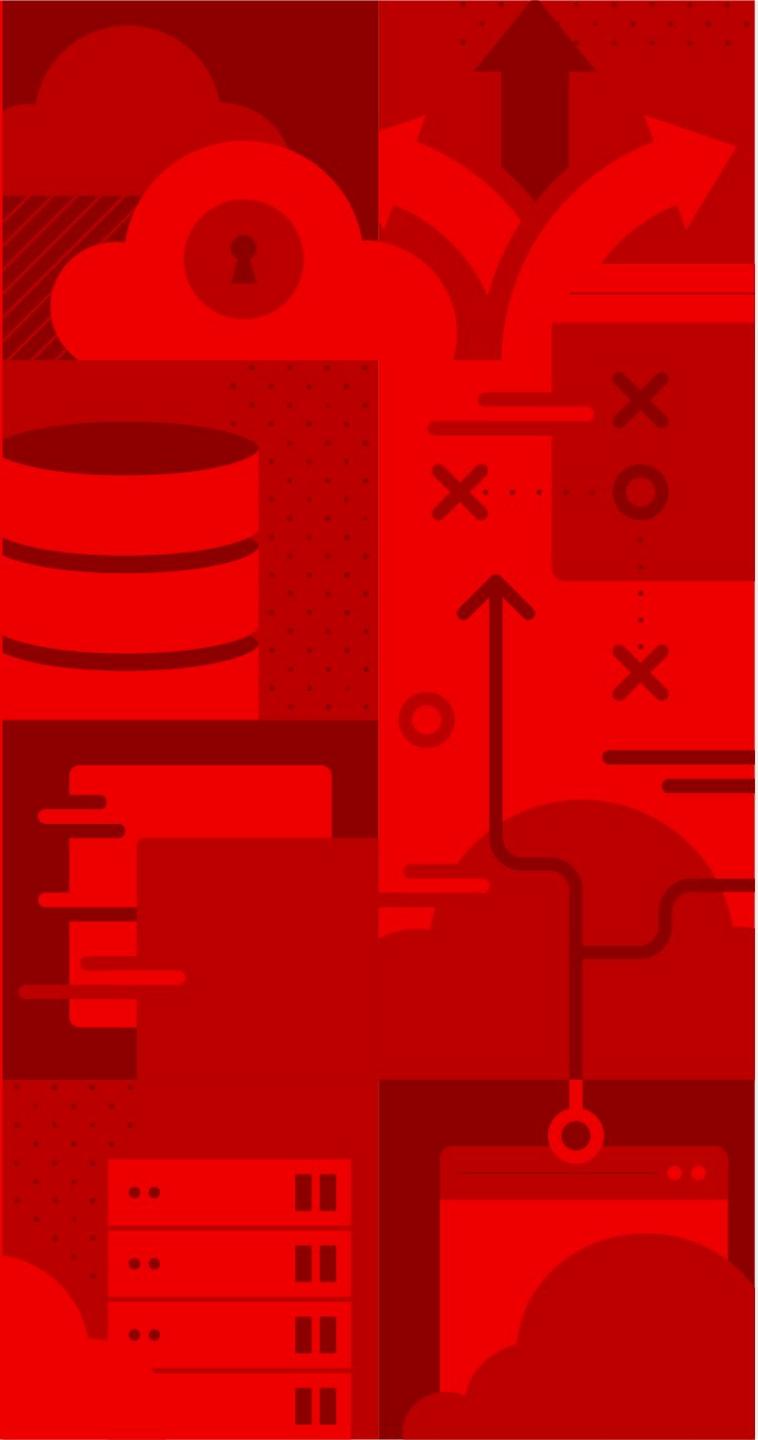
アプリケーションはlabelに基づいてデプロイされます



# ACMでポリシーを作成・適用

ポリシーの中でlabelに基づいてクラスタを選択し、適用します





# Thank you

Red Hat is the world's leading provider of enterprise open source software solutions. Award-winning support, training, and consulting services make Red Hat a trusted adviser to the Fortune 500.



[linkedin.com/company/red-hat](https://www.linkedin.com/company/red-hat)



[facebook.com/redhatinc](https://www.facebook.com/redhatinc)



[youtube.com/user/RedHatVideos](https://www.youtube.com/user/RedHatVideos)



[twitter.com/RedHat](https://twitter.com/RedHat)