OpenShift ACM

Red Hat

Open Cluster Manager

TRAINING

CHOI GOOK HYUN
Freelancer Instructor

CONFIDENTIAL designator

DAY 1

- About the OpenShift Advanced Cluster

Red Hat

THE FEDERATION

CONFIDENTIAL designator

THE FEDERATION

with kubernetes

"The Federation" API는 올해 2022년도 10월에 더 이상 개발을 하지 않기로 결정하였음. 다만, API사양은 계속 개발하며, 프로그램 부분은 OCM, KARMADA와 같은 커뮤니티에 개발을 맡기기로 결정.

공식적으로 Federation API는 v1, v2기반으로 되어 있으며, 쿠버네티스는 API정의 애플리케이션 개발은 외부 그룹에서 맡기로 하였음.

현재는 다음과 같이 두 그룹이 주요 리더로 끌고 가고 있음.

- Open Cluster Manager
- Karmada

V0000000

Red Hat

CONFIDENTIAL designator

THE FEDERATION

with kubernetes

Follow-up: discussion on archiving Kubefed 조회수 177회

pmo...@gmail.com
받는사람 kubernetes-sig-multicluster

Everyone-

In recent Kubecon SIG updates we have previously communicated that SIG leadership has been considering archiving the Kubefed[1] repository. This topic was given dis discuss offline. This communication is a follow-up on that action item.

TLDR: Our current favored option is to archive the kubefed project but we are soliciting community input before making a final decision. Continue reading for rationale.

In our discussions, Jeremy and I considered the following questions:

Red Hat

CONFIDENTIAL designator

KARMADA

Red Hat

CONFIDENTIAL designator

KARMADA

with kubernetes

KARMADA는 쿠버네티스 federation v1, v2프로토콜 기반으로 작성이 되어 있음. 쿠버네티스 기반으로 동작하며, 동작 방식은 OCM과 매우 흡사하나, 중간에 "Karmada API Server"중심으로 "etcd", "scheduler"와 통신하며 중간에 "Karmada controller"를 통해서 정책 배포 및 관리를 한다.

현재 KARMADA는 "CLOUD NATIVE"의 샌드박스(Sandbox) 프로젝트로 선언이 되어 있다.
CNCF에서는 좀 더 KARMADA를 지원하고 있는 분위기.

V0000000

Red Hat

CONFIDENTIAL designator

KARMADA

with kubernetes

주요 목적은 다음과 같다.

- KIS native API compatible
- Out of the Box
- Avoid Vendor Lock-in
- Centralized Management
- Fruitful Multi-Cluster Scheduling Policiese
- Open and Neutral

CONFIDENTIAL designator

KARMADA

Red Hat

CONFIDENTIAL designator

Karmada Concepts

Red Hat

CONFIDENTIAL designator

OCM

Red Hat

CONFIDENTIAL designator

OCM

with OpenShift

OpenShift는 Kubernetes기반으로 작성이 되어 있음. API는 두 가지 API로 구별이 됨.

- Cluster API
- Federation API

Cluster API는 Kubernetes SIG에서 맨 처음 릴리즈 한 API. Federation API는 다중 클러스터 운영을 위해서 SIG에서 개발을 시작. OCM에는 현재 레드햇이 개발을 참여하고 있으며, 이 프로젝트도 "CLOUD NATIVE COMPUTING FOUNDATION"의 일부분.

프로젝트 주소

https://github.com/stolostron

V0000000

Red Hat

CONFIDENTIAL designator

Federation

Viewer does not support full SVG 1.1

V0000000

CONFIDENTIAL designator

OCM Arch

V0000000

CONFIDENTIAL designator

OCM

MultiCluster API

클러스터 API는 다음과 같은 기능을 기본적으로 다루고 있다.

- 자원을 라이프 사이클(LifeCycle) 관리
- 설정 및 정책(Deployment, Policy) 적용 및 배포
- Pod 및 컨테이너 복제
- 네임스페이스 관리

CONFIDENTIAL designator

OCM

Federation V1

Fede v1는 SIG에서 멀티 클러스터를 관리하기 위해서 프로젝트를 시작하였다. 쿠버네티스 SIG에서 제일 큰 프로젝트 중 하나 이었으며, 앞서 말한 것처럼 멀티 클러스터 관리를 위해서 v1를 설계를 하였다. 주요 목적은 다음과 같다.

- Difficulties in re-implementing the Kubernetes API at the cluster level, as federation-specific extensions were stored in annotations.
- Limited flexibility in federated types, placement and reconciliation, due to 1:1 emulation of the Kubernetes API.
- No settled path to GA, and general confusion on API maturity; for example, Deployments are GA in Kubernetes but not even Beta in Federation v1.

V0000000

Red Hat

CONFIDENTIAL designator

OCM

Federation V1

Federation은 v1기반으로 v2로 진화가 되었으며, 기본적인 컨셉은 v1에서 가져와서 v2에서 기능 추가를 시작하였다.

Federation은 "Hub and Spoke(pull and push)"구조를 가지고 있고 다음과 같이 동작한다.

V0000000

CONFIDENTIAL designator

Native Federation Diagram

CONFIDENTIAL designator

OCM

Federation

연합의 주요 목표는 쿠버네티스 자원을 연합 그리고, "API to API"를 그룹으로 정의 하는게 주요 목표이자 목적이다.

쿠버네티스의 기능을 확장하기 위해서 **CRD(CustomResourceDefinitions)**를 통해서 구현이 된다. Federation API를 통해서 요청이 들어오면, 단일 클러스터 API를 배포하는 메커니즘이다.

V0000000

CONFIDENTIAL designator

Kubernetes

Federation

During the journey of defining building blocks of the federation APIs, one of the near term goals also evolved as 'to be able to create a simple federation a.k.a. simple propagation of any Kubernetes resource or a CRD, writing almost zero code'. What ensued further was a core API group defining the building blocks as a Template resource, a Placement resource and an Override resource per given Kubernetes resource, a TypeConfig to specify sync or no sync for the given resource and associated controller(s) to carry out the sync.

간단한 요약

CRD기반으로 별도의 코드 작성이 필요 없이 CORE API에게 정의하여 자원을 배포하는 형식. 이때 템플릿 자원들로 빌드 블록을 만들어서 Placement자원을 각각 쿠버네티스 자원에 오버라이드 하여 동기화 혹은 비동기화 여부를 컨트롤러를 통해서 결정

Red Hat

CONFIDENTIAL designator

HUB and SPOKE

CONFIDENTIAL designator

HUB and SPOKE

V0000000

CONFIDENTIAL designator

Template

Federation

Federation에서 주요 자원 지시자는 다음과 같다.

- Template
  - A Template type holds the base specification of the resource - for example, a type called **FederatedReplicaSet** holds the base specification of a **ReplicaSet** that should be distributed to the targeted clusters

V0000000

Red Hat

CONFIDENTIAL designator

Placement

Federation

- Placement
  - A Placement type holds the specification of the clusters the resource should be distributed to - for example, a type called **FederatedReplicaSetPlacement** holds information about which clusters **FederatedReplicaSets** should be distributed to

V0000000

CONFIDENTIAL designator

Override

Federation

- Override
  - An optional Overrides type holds the specification of how the Template resource should be varied in some clusters - for example, a type called **FederatedReplicaSetOverrides** holds information about how a **FederatedReplicaSet** should be varied in certain clusters.

V0000000

CONFIDENTIAL designator

TypeConfig

Federation

- TypeConfig
  - specify sync or no sync for the given resource and associated controller(s) to carry out the sync.

V0000000

Red Hat

CONFIDENTIAL designator

Federation v2

Higher-level behaviour

API v2에서 핵심 기능은 API에서 "템플릿", "배치" 및 "오버라이드"를 통해서 컨트롤러(Controller)통해서 상위 API를 재구성(CRD) 합니다.

클러스터를 가로지르면서 자원을 생성 시 "ReplicaSchedulingPreference"를 통해서 Deployment 및 ReplicaSet를 통해서 연합 클러스터에 자원을 생성 및 구성이 가능하다. 역시 기존 클러스터에서 사용하였던 기능 "limits", "quote"같은 자원 관리도 API를 통해서 배포가 가능하다.

V0000000

Red Hat

CONFIDENTIAL designator

OpenShift vs OCM

Components

오픈 시프트의 ACM은 쿠버네티스의 Federation기반으로 되어 있으며, 레드햇은 OCM기반으로 ACM를 개발하기 시작 하였다. 레드햇 ACM은 다음과 같은 부분에 OCM과 다르다.

- 손쉬운 설치 및 오픈 시프트와 통합된 환경
- 클라우드 환경에서 손쉬운 확장
- Quay.io, Ceph, InSight(OpenScape)와 같은 에코 환경 구성
- 강화된 보안모델 제공

V0000000

Red Hat

CONFIDENTIAL designator

OpenShift ACM

Components

ACM은 오픈시프트와 동일하게 통합된 CI/CD환경을 여러 클러스터에 제공한다. 이때 사용하는 개념은 Hub 및 Subscription기반으로 동작하게 된다. ACM는 Fleets라는 개념기반(앞서 이야기한 Placement)으로 클러스터를 관리한다.

오픈스택의 Placement와 비슷하지만 조금 다른 개념이다.

진행 전, 미리 한번 비교 해본다.

V0000000

Red Hat

OpenStack Placement

OCM/ACM Placement

HUB CLUSTER
User namespace
Policy Custom Resource (root)
Spec
Status
Watch A
Update B
Policy Propagator Controller
Update A
Watch B
Cluster namespace
Policy Custom Resource (replicated)
Spec
Status
Watch C
Placement Binding Custom Resource
Placement Rule Custom Resource
Watch
Update
MANAGED CLUSTER
Spec Sync Controller
Status Sync Controller
Update
Watch
Update
Policy Custom Resource
Spec
Status
Watch
Template Sync Controller
Update
Policy Template Custom Resources
Spec
Status
Watch
Update
Watch
Events for Policy
Create
Out-of-Box/ Custom Policy Controllers

CONFIDENTIAL designator

Klusterlet

Components

OCM 그리고 ACM에서는 Klusterlet은 매우 중요하다.

Klusterlet은 기존 쿠버네티스나 혹은 오픈 시프트에서 사용하는 kubelet하고 비슷한 기능을 한다.
다만, 차이점이라고 하면, "Node to Node"에서 "Cluster to Cluster"로 변경이 되었다.

V0000000

CONFIDENTIAL designator

Klusterlet

Components

V0000000

162_RHACMK_0521

Hub Cluster
Multi-cluster controller
(e.g. subscription)
list-watch
Workspace namespace
Placement
reactive
scheduling
Placement
Decision
bind
Clusterset #1
Managed Cluster #1
Managed Cluster #2
Managed Cluster #3
....
Managed Cluster #m
....
Managed Cluster #n
scheduling results
Predicate
- Label selector
- Claim selector
- Taints-based (Require)
Prioritize
- Number of clusters
- Scoring-based ranking
- Taints-based (Prefer)

CONFIDENTIAL designator

Placement

Components

```
apiVersion: cluster.open-cluster-management.io/v1beta1
kind: Placement
metadata:
  name: placement1
  namespace: default
spec:
  numberOfClusters: 3
  clusterSets:
    - prod
```


V0000000

https://open-cluster-management.io/concepts/placement/

CONFIDENTIAL designator

Placement

Components

```
predicates:
  - requiredClusterSelector:
      labelSelector:
        matchLabels:
          purpose: test
      claimSelector:
        matchExpressions:
          - key: platform.open-cluster-management.io
            operator: In
            values:
              - aws
```


V0000000

https://open-cluster-management.io/concepts/placement/

CONFIDENTIAL designator

Red Hat

CONFIDENTIAL designator

DAY 2

- Role based Access Control

CONFIDENTIAL designator

ROLE BASED

Red Hat

CONFIDENTIAL designator

ROLE-BASED

ACCESS CONTROL

V0000000

https://open-cluster-management.io/concepts/placement/

CONFIDENTIAL designator

ROLE-BASED

ACCESS CONTROL

open-source Identity and Access Management (IAM)
Red Hat Identity Management (IdM)

- **Identity** (machine, user, virtual machines, groups, authentication credentials)
- **Policy** (host based access control)
- **Audit** (this component is deferred)

Keycloak

V0000000

https://open-cluster-management.io/concepts/placement/

Propagator/Propaganda

CONFIDENTIAL designator

Propagator

watch and update

https://github.com/open-cluster-management-io/governance-policy-propagator

V0000000

https://open-cluster-management.io/concepts/placement/

CONFIDENTIAL designator

propagation.go

update

https://github.com/open-cluster-management-io/governance-policy-propagator/blob/main/controllers/propagator/propagation.go

```
const (
        startDelim              = "{{hub"
        stopDelim               = "hub}}"
        TriggerUpdateAnnotation = "policy.open-cluster-management.io/trigger-update"
)
```


V0000000

https://open-cluster-management.io/concepts/placement/

CONFIDENTIAL designator

propagation.go

watch

https://github.com/open-cluster-management-io/governance-policy-propagator/blob/main/controllers/propagator/propagation.go

```go
// decisionResult contains the result of handling a placement decision of a policy. It is intended
// to be sent in a channel by handleDecisionWrapper for the calling Go routine to determine if the
// processing was successful. Identifier is the PlacementDecision, with the ClusterNamespace and
// the ClusterName. TemplateRefObjs is a set of identifiers of objects accessed by hub policy
// templates. Err is the error associated with handling the decision. This can be nil to denote success.
type decisionResult struct {
        Identifier      appsv1.PlacementDecision
        TemplateRefObjs map[k8sdepwatches.ObjectIdentifier]bool
        Err             error
}
```


V0000000

https://open-cluster-management.io/concepts/placement/

CONFIDENTIAL designator

template-sync

the archived

V0000000

https://open-cluster-management.io/concepts/placement/

CONFIDENTIAL designator

template-sync

https://github.com/open-cluster-management-io/governance-policy-template-sync-archive

```
var log = ctrl.Log.WithName(ControllerName)

//+kubebuilder:rbac:groups=policy.open-cluster-management.io,resources=*,verbs=get;list;watch;create;update;patch;delete
//+kubebuilder:rbac:groups=core,resources=events,verbs=get;list;watch;create;update;patch;delete
```


V0000000

https://open-cluster-management.io/concepts/placement/

CONFIDENTIAL designator

policy controller

NIST-800-53

https://github.com/open-cluster-management-io/governance-policy-addon-controller

V0000000

https://open-cluster-management.io/concepts/placement/

CONFIDENTIAL designator

policy controller

NIST-800-53

https://csrc.nist.gov/publications/detail/sp/800-53/rev-5/final

https://csf.tools/reference/nist-sp-800-53/r5/cm/cm-2/

V0000000

https://open-cluster-management.io/concepts/placement/

CONFIDENTIAL designator

policy controller

image

```
  image: quay.io/open-cluster-management/governance-policy-propagator:latest
  imagePullPolicy: Always
  name: governance-policy-propagator
  ports:
  - containerPort: 8383
    name: http
    protocol: TCP
serviceAccountName: governance-policy-propagator
```


V0000000

https://open-cluster-management.io/concepts/placement/

CONFIDENTIAL designator

policy controller

image

V0000000

https://open-cluster-management.io/concepts/placement/

CONFIDENTIAL designator

IAM Policy

controller

https://github.com/stolostron/iam-policy-controller

V0000000

https://open-cluster-management.io/concepts/placement/

CONFIDENTIAL designator

IAM Policy

사용자화

https://github.com/stolostron/governance-policy-framework

V0000000

https://open-cluster-management.io/concepts/placement/

CONFIDENTIAL designator

subscription

사용자화

https://github.com/open-cluster-management-io/multicloud-operators-subscription

V0000000

CONFIDENTIAL designator

subscription

사용자화

https://github.com/open-cluster-management-io/multicloud-operators-subscription

CHANNEL
GitHub
HELM
Object Store
(S3, MinIO, etc.)
1
Hub Cluster
(OCM Control
Plane)
HUB SUBSCRIPTION
2
1 Hub Subscription discover resources and evaluate placement rules
2 Subscription propagation to Managed Clusters
3 Local Subscription apply resources on the Managed-clusters
Local Subscription
Local Subscription
3
3
Managed Clusters
(OCM Agents)

CONFIDENTIAL designator

Observability

Red Hat

CONFIDENTIAL designator

Observability

모니터링 및 분석

MANAGED CLUSTER
HUB SERVER
Grafana
Dashboard Loader
Management Ingress
Browser
Endpoint Operator
MCO Operator
Alert manager
RBAC Proxy
Platform Prometheus
Metrics collector
Observatorium API
Thanos ecosystem
External object store
Observatorium Operator
Legend
data flow
configures

CONFIDENTIAL designator

Thanos Project

사용자화

V0000000

https://open-cluster-management.io/concepts/placement/

CONFIDENTIAL designator

Thanos Project

모니터링

이 프로젝트의 기반은 **Prometheus**이며, 컨테이너 기반의 pv,pvc에서 오랜 기간 데이터 저장 및 분석하는 도구.

프로젝트 홈페이지는 다음과 같다.

https://thanos.io/

V0000000

https://open-cluster-management.io/concepts/placement/

CONFIDENTIAL designator

Thanos Project

조건

- Put Prometheus in the same failure domain. This means same network, same datacenter as monitoring services.
- Use persistent disk to persist data across Prometheus restarts.
- Use local compaction for longer retentions.
- Do not change min TSDB block durations.
- Do not scale out Prometheus unless necessary. Single Prometheus is highly efficient (:

V0000000

https://open-cluster-management.io/concepts/placement/

CONFIDENTIAL designator

Thanos Project

조건

- Sidecar: connects to Prometheus, reads its data for query and/or uploads it to cloud storage.
- Store Gateway: serves metrics inside of a cloud storage bucket.
- Compactor: compacts, downsamples and applies retention on the data stored in cloud storage bucket.
- Receiver: receives data from Prometheus's remote-write WAL, exposes it and/or upload it to cloud storage.
- Ruler/Rule: evaluates recording and alerting rules against data in Thanos for exposition and/or upload.
- Querier/Query: implements Prometheus's v1 API to aggregate data from the underlying components.
- Query Frontend: implements Prometheus's v1 API proxies it to Query while caching the response and optional splitting by queries day.

V0000000

https://open-cluster-management.io/concepts/placement/

CONFIDENTIAL designator

Thanos Project

what is the Sidecar?

- Thanos integrates with existing Prometheus servers through a Sidecar process, which runs on the same machine or in the same pod as the Prometheus server.
- The purpose of the Sidecar is to backup Prometheus data into an Object Storage bucket, and give other Thanos components access to the Prometheus metrics via a gRPC API.

V0000000

https://open-cluster-management.io/concepts/placement/

사이드카

https://open-cluster-management.io/concepts/placement/

CONFIDENTIAL designator

Thanos Project

사용자화

배포

https://open-cluster-management.io/conce

Red Hat

CONFIDENTIAL designator

gatekeeper

사용자화

https://github.com/open-policy-agent/gatekeeper

V0000000

CONFIDENTIAL designator

DAY 3

- CI/CD

CONFIDENTIAL designator

CI/CD

Red Hat

CONFIDENTIAL designator

CI/CD

ACM

V0000000

https://open-cluster-management.io/concepts/placement/

CONFIDENTIAL designator

CI/CD

ACM

- Convert any Kubernetes manifest files to RHACM configuration policies.
- Patch the input Kubernetes manifests before they are inserted into a generated RHACM policy.
- Generate additional configuration policies to be able to report on Gatekeeper and Kyverno policy violations through RHACM.
- Be run locally so that generated RHACM policies can be stored directly in Git and applied using GitOps.
- Be run by GitOps tools so that only the manifests and the policy generator configuration need to be stored in Git.

V0000000

https://open-cluster-management.io/concepts/placement/

CONFIDENTIAL designator

CI/CD

ACM

- The policy generator is implemented as a Kustomize generator plugin. This allows for any Kustomize configuration (e.g. patches, common labels) to be applied to the generated policies. This also means that the policy generator can be integrated into existing GitOps processes that support Kustomize, such as RHACM Application Lifecycle Management and OpenShift GitOps (i.e. ArgoCD).

V0000000

https://open-cluster-management.io/concepts/placement/

CONFIDENTIAL designator

CI/CD

ACM

Field	Description
spec.remediationAction	Optional. Specifies the **remediation** of your policy.The parameter values are **enforce** and inform. If specified, the **spec.remediationAction** value that is defined overrides the **remediationAction** parameter defined in the child policy, from the **policy-templates** section. For example, if **spec.remediationAction** value section is set to enforce, then the remediationAction in the **policy-templates** section is set to enforce during runtime. Important: Some policies might not support the enforce feature.

V0000000

https://open-cluster-management.io/concepts/placement/

CONFIDENTIAL designator

CI/CD

ACM

```
27          spec:
28            remediationAction: enforce # the policy-template spec.remediationAction is overridden by the preceding parameter value for spec.remediationAction.
29            severity: low
30            namespaceSelector:
31              exclude: ["kube-*"]
32              include: ["*"]
33            object-templates:
34              - complianceType: musthave
35                objectDefinition:
36                  apiVersion: v1
37                  kind: Namespace
38                  metadata:
39                    name: falco-operator
```


V0000000

https://open-cluster-management.io/concepts/placement/

CONFIDENTIAL designator

CI/CD

The Policy Template

적용 부분

https://www.sans.org/information-security-policy/

이론 부분

https://policylibrary.gatech.edu/Policy_Development_-_Policy_Template

소스코드

https://github.com/stolostron/policy-collection/blob/main/community/README.md

NIST Spec.

NIST.SP.800-53r4.

V0000000

https://open-cluster-management.io/concepts/placement/

CONFIDENTIAL designator

CI/CD

ACM

ACM현재 다음과 같은 CI/CD 도구를 지원하고 있다.

REGISTRY

- Amazon Elastic Container Registry (ECR)
- Docker Hub
- Google Container Registry (GCR)
- Google Artifact Registry
- IBM Cloud Container Registry (ICR)
- JFrog Artifactory
- Microsoft Azure Container Registry (ACR)
- Red Hat Quay
- Red Hat container registries
- Sonatype Nexus
- Any other registry that uses the Docker Registry HTTP API

V0000000

https://open-cluster-management.io/concepts/placement/

CONFIDENTIAL designator

CI/CD

ACM

CI(Continuous Integration)

CI도구는 일반적으로 git, svn같은 기능을 이야기한다.

CD(Continuous Delivery)

CD도구는 일반적으로 Tekton, Jenkins 그리고 ArgoCD같은 도구이다.

CONFIDENTIAL designator

CI/CD

ACM

GitOps

The core idea of GitOps is having a Git repository that always contains declarative descriptions of the infrastructure currently desired in the production environment and an automated process to make the production environment match the described state in the repository. If you want to deploy a new application or update an existing one, you only need to update the repository - the automated process handles everything else. It's like having cruise control for managing your applications in production.

Git Operation, versioned CI/CD on top of declarative infrastructure. Stop scripting and start shipping.

V0000000

Red Hat

CONFIDENTIAL designator

HoH GitOps Mechanism

Red Hat

217_OpenShift_0222

CONFIDENTIAL designator

CI/CD

ACM

CI/CD주요 부분

https://github.com/stolostron/hub-of-hubs-gitops

https://github.com/stolostron/acm-aap-aas-operations

관계가 있는 부분

https://github.com/stolostron/ci-cluster-config

https://github.com/stolostron/multicluster-observability-operator

https://github.com/noseka1/rhacm-kustomization

https://github.com/stolostron/react-data-view

V0000000

Red Hat

CONFIDENTIAL designator

Quay

저장소

Quay는 다음과 같은 기능을 가지고 있다.

clair Public
Vulnerability Static Analysis for Containers
Go 9.1k 1.1k

quay Public
Build, Store, and Distribute your Applications and Containers
Python 2.1k 221

jwtproxy Public
An HTTP-Proxy that adds AuthN through JWTs
Go 394 43

quay-operator Public
Operator for deploying and managing Quay
Go 106 70

claircore Public
foundation modules for scanning container packages and reporting vulnerabilities
Go 98 56

quayctl Public
quayctl is a command-line client for Quay
Go 90 17

Red Hat

CONFIDENTIAL designator

Quay

저장소

clair/claircore

SCAP기반으로 프로토콜 확인. 인덱스 기반으로 검사 후 취약점 검사하여 사용자에게 알려준다.

jwtproxy

JSON 토큰 처리시 사용하는 토큰 프록시 서버.

V0000000

CONFIDENTIAL designator

Quay

저장소

quay

OCI이미지 관리하는 이미지 레지스트리 서버. 저장소는 기본적으로 로컬 저장소를 사용하고 있으며, 사용자가 원하는 경우, 다른 백-엔드 스토리지 사용이 가능하다.

quay-operator

쿠버네티스나 혹은 오픈 시프트에서 사용 시, 필요한 프로비저닝 오퍼레이터.

V0000000

CONFIDENTIAL designator

Quay

Clair

CONFIDENTIAL designator

Quay

저장소

- High availability
- Geo-replication
- Repository mirroring
- Docker v2, schema 2 (multiarch) support
- Continuous integration
- Security scanning with Clair
- Custom log rotation
- Zero downtime garbage collection
- 24/7 support

CONFIDENTIAL designator

Quay

저장소

- Project Quay provides support for:
- Multiple authentication and access methods
- Multiple storage backends
- Custom certificates for Quay, Clair, and storage backends
- Application registries
- Different container image types

V0000000

CONFIDENTIAL designator

Quay

저장소

본래 호스티드 형태로 제작이 되었다가, 오픈 시프트가 릴리즈 되면서 컨테이너드 형태로 제작이 되었음.

오픈소스 버전과 상용 버전에 많은 차이가 있으며, Quay기존의 git기능에 다중 접근 및 저장이 가능하도록 프론트 및 백엔드가 작성이 되어있다. 또한, H/A , Scape를 통해서 보안이 강화가 되었다.

https://quay.github.io/claircore/what_is.html

https://docs.projectquay.io/deploy_quay.html

V0000000

Red Hat

CONFIDENTIAL designator

Quay

저장소

https://github.com/containers/skopeo

util/repomirror/skopeomirror.py

```
logger = logging.getLogger(__name__)

SKOPEO_TIMEOUT_SECONDS = 300

# success: True or False whether call was successful
# tags: list of tags or empty list
# stdout: stdout from skopeo subprocess
# stderr: stderr from skopeo subprocess
```

Python Showing the top two matches Last indexed 21 days ago

미러기능 및 이미지 정보 조회시 Skopeo meta조회 기능을 통해서 이미지 정보 확인한다.

Red Hat

CONFIDENTIAL designator

Quay

저장소

미러 정보 복사

```
        args = ["/usr/bin/skopeo"]
        if verbose_logs:
            args = args + ["--debug"]
        if unsigned_images:
            args = args + ["--insecure-policy"]

        args = args + [
            "copy",
            "--all",
            "--remove-signatures",
            "--src-tls-verify=%s" % src_tls_verify,
            "--dest-tls-verify=%s" % dest_tls_verify,
        ]
```


Red Hat

CONFIDENTIAL designator

Kustomize ACM

YAML

https://kustomize.io/

Kustomize introduces a template-free way to customize application configuration that simplifies the use of off-the-shelf applications. Now, built into kubectl as apply -k.

https://cloud.redhat.com/blog/generating-governance-policies-using-kustomize-and-gitops

V0000000

CONFIDENTIAL designator

Why Not NFS for the Quay?

YAML

https://github.com/kube-object-storage/lib-bucket-provisioner

```
15
16          ocsv1a1 "github.com/kube-object-storage/lib-bucket-provisioner/pkg/apis/objectbucket.io/v1alpha1"
17  )
```


Red Hat

CONFIDENTIAL designator

Why Not NFS for the Quay?

YAML

여러가지 이유가 있지만, 동시 쓰기가 발생시 파일 시스템의 inode 및 journal이 손상이 될 가능성이 있기 때문에, 현재는 안전상 이유로 S3 혹은 호환되는 object storage만 지원하고 있음.

Red Hat

CONFIDENTIAL designator

DAY 4

- SECURITY

Red Hat

CONFIDENTIAL designator

Security

Red Hat

CONFIDENTIAL designator

StackRox

EnForcement Policy

V0000000

CONFIDENTIAL designator

StackRox

EnForcement Policy

오픈소스 버전

https://www.stackrox.io/blog/open-source-stackrox-is-now-available/

Red Hat

CONFIDENTIAL designator

StackRox

EnForcement Policy

In 2021, Red Hat acquired StackRox and the first Kubernetes-native security platform with an innovative approach to container security. Last year we committed to supporting the open source community, enabling developer and security teams to deliver more secure applications faster. With StackRox, organizations will be able to solve their Kubernetes security challenges, including:

CONFIDENTIAL designator

StackRox

EnForcement Policy

- Automating DevSecOps
- Shift security left
- Operationalizing full life cycle application security in Kubernetes
- Overcome the challenges of Kubernetes security

CONFIDENTIAL designator

StackRox

EnForcement Policy

컨테이너 이미지

https://quay.io/organization/stackrox-io

V0000000

CONFIDENTIAL designator

StackRox

EnForcement Policy

레드햇의 제품이름은..

Red Hat Advanced Cluster Security for Kubernetes

https://docs.openshift.com/acs/3.72/welcome/index.html

CONFIDENTIAL designator

StackRox

214_RHACS_0322)000

Red Hat

StackRox

EnForcement Policy

214_RHACS_0322

CONFIDENTIAL designator

StackRox

THE POLICY OPA

Red Hat

CONFIDENTIAL designator

StackRox

THE POLICY OPA

V0000000

Red Hat

CONFIDENTIAL designator

StackRox

ENVOY

엔보이 웹 사이트

https://www.envoyproxy.io/

load balancer
resolves istio domain
Key
VM
Job
control plane
data plane
istio-router
envoy ingress
cloud-controller
CAPI
diego-api
BBS
istio-control
copilot
pilot
diego-cell
container
envoy
sidecar
app

CONFIDENTIAL designator

StackRox

ISTIO

ISTIO

https://istio.io/

CONFIDENTIAL designator

StackRox

OPENSHIFT

OpenShift Mesh

https://cloud.redhat.com/blog/introducing-openshift-service-mesh-2.0

StackRox

ISTIO

Istio Architecture

Red Hat

CONFIDENTIAL designator

StackRox

EnForcement Policy

definitions.stackrox.io:

RHACS aggregates the data from various vulnerability feeds at the definitions.stackrox.io endpoint and then passes this information to Central. The feeds include general, NVD, and distribution-specific, such as Alpine, Debian, and Ubuntu.

collector-modules.stackrox.io:

Central reaches out to collector-modules.stackrox.io to obtain supported kernel modules and passes on these modules to Sensor and Collector.

V0000000

CONFIDENTIAL designator

StackRox

EnForcement Policy

definitions.stackrox.io:

RHACS aggregates the data from various vulnerability feeds at the definitions.stackrox.io endpoint and then passes this information to Central. The feeds include general, NVD, and distribution-specific, such as Alpine, Debian, and Ubuntu.

collector-modules.stackrox.io:

Central reaches out to collector-modules.stackrox.io to obtain supported kernel modules and passes on these modules to Sensor and Collector.

V0000000

CONFIDENTIAL designator

StackRox

EnForcement Policy

ponents	1 for multiple clusters.	Central
		Scanner
Per-cluster components	1 for each cluster.	Sensor
		Admission controller
Per-node component	1 on each node.	Collector

Red Hat

CONFIDENTIAL designator

StackRox

EnForcement Policy

- Central

Central is the main component of Red Hat Advanced Cluster Security for Kubernetes and it is installed as a Kubernetes deployment. It handles data persistence, API interactions, and user interface (Portal) access. You can use the same Central instance to secure multiple OpenShift Container Platform or Kubernetes clusters.

V0000000

CONFIDENTIAL designator

StackRox

EnForcement Policy

- Scanner

Red Hat Advanced Cluster Security for Kubernetes includes an image vulnerability scanning component called Scanner. It analyzes all image layers to check for known vulnerabilities from the Common Vulnerabilities and Exposures (CVEs) list. Scanner also identifies vulnerabilities in packages installed by package managers and in dependencies for multiple programming languages.

CONFIDENTIAL designator

StackRox

EnForcement Policy

- Per-cluster components

You deploy the per-cluster components into each cluster that you want to monitor. Red Hat Advanced Cluster Security for Kubernetes includes the following per-cluster components:

- Sensor
- Admission controller

V0000000

CONFIDENTIAL designator

StackRox

EnForcement Policy

- Sensor

Red Hat Advanced Cluster Security for Kubernetes uses the Sensor component to monitor Kubernetes and OpenShift Container Platform clusters. It handles interactions with the OpenShift Container Platform or Kubernetes API server for policy detection and enforcement, and it coordinates with Collector.

CONFIDENTIAL designator

StackRox

EnForcement Policy

- Admission controller

The admission controller prevents users from creating workloads that violate security policies in Red Hat Advanced Cluster Security for Kubernetes.

V0000000

CONFIDENTIAL designator

StackRox

EnForcement Policy

- Per-node components

You deploy the per-node components in all nodes that you want to monitor. Red Hat Advanced Cluster Security for Kubernetes includes the following per-cluster components:

- Collector

V0000000

CONFIDENTIAL designator

StackRox

EnForcement Policy

- Collector

Collector collects and monitors information about container runtime and network activity. It then sends the collected information to Sensor.

CONFIDENTIAL designator

StackRox

THE GATEKEEPER

THE GATEKEEPER

https://github.com/open-policy-agent/gatekeeper

https://www.openpolicyagent.org/docs/latest/kubernetes-tutorial/

V0000000

CONFIDENTIAL designator

StackRox

THE GATEKEEPER

V0000000

Red Hat

CONFIDENTIAL designator

StackRox

THE GATEKEEPER

Red Hat

CONFIDENTIAL designator

StackRox

ENVOY

Red Hat

CONFIDENTIAL designator

StackRox

Force Policy

CONFIDENTIAL designator

StackRox

EnForcement Policy

Red Hat

StackRox: The First Kubernetes-Native Security Platform

StackRox Kubernetes Security Platform Architecture

User
Request
Service
Policy Query
[JSON]
Policy decision
[JSON]
OPA Engine
Policy evaluation
Policy
[Rego]
Data
[JSON]

CONFIDENTIAL designator

StackRox

Red Hat

CONFIDENTIAL designator

StackRox

EnForcement Policy

```yaml
# Only allow HTTP and HTTPS traffic based on Pod labels
kind: NetworkPolicy
apiVersion: networking.k8s.io/v1
metadata:
  name: allow-http-and-https-from-and-to-blue
spec:
  podSelector:
    matchLabels:
      color: blue
  ingress:
  - from:
    - podSelector:
        matchLabels:
          color: blue
  - ports:
    - protocol: TCP
      port: 80
    - protocol: TCP
      port: 443
```

Red Hat

CONFIDENTIAL designator

StackRox

EnForcement Policy

Red Hat