Red Hat Advanced Cluster Management for Kubernetes 2.10

GitOps

GitOps

Last Updated: 2024-08-16

Red Hat Advanced Cluster Management for Kubernetes 2.10 GitOps

GitOps

법적 공지

초록

통합 GitOps 및 Argo CD 사용 방법을 알아보려면 자세한 내용을 확인하십시오.

차례

1장. GITOPS 개요

Red Hat OpenShift Container Platform GitOps 및 Argo CD는 Red Hat Advanced Cluster Management for Kubernetes와 통합되며, 원래 애플리케이션 라이프사이클 *채널* 및 *서브스크립션* 모델에 비해 고급 기능을 제공합니다.

Argo CD 개발과 GitOps 통합은 물론 Argo CD에 기능 개선 및 업데이트를 제공하는 대규모 커뮤니티도 활성화됩니다. OpenShift Container Platform GitOps Operator를 사용하면 Argo CD 개발의 최신 개선 사항을 사용하고 GitOps Operator 서브스크립션에서 지원을 받을 수 있습니다.

OpenShift Container Platform GitOps 및 Argo CD와 Kubernetes 통합을 위한 Red Hat Advanced Cluster Management에 대한 자세한 내용은 다음 항목을 참조하십시오.

- GitOps 콘솔
- OpenShift GitOps Operator에 관리형 클러스터 등록
- GitOps에 대한 애플리케이션 배치 허용 오차 구성
- 푸시 및 가져오기 모델을 사용하여 Argo CD 배포
- GitOps Operator를 설치하기 위한 정책 생성
- OpenShift Container Platform GitOps(Argo CD)로 정책 정의 관리

1.1. GITOPS 콘솔

통합된 OpenShift Container Platform GitOps 콘솔 기능에 대해 자세히 알아보십시오. *ApplicationSet*, *Argo CD* 유형과 같은 애플리케이션을 생성하고 확인합니다. **ApplicationSet** 은 컨트롤러에서 생성된 Argo 애플리케이션을 나타냅니다.

- Argo CD **ApplicationSet** 을 생성하려면 동기화 **정책에서 클러스터 상태가 변경될 때 자동으로 동기화** 를 활성화해야 합니다.
- **kustomization** 컨트롤러가 있는 Flux의 경우 **kustomize.toolkit.fluxcd.io/name=<app_name>** 레이블이 있는 Kubernetes 리소스를 찾습니다.
- **helm** 컨트롤러가 있는 Flux의 경우 **helm.toolkit.fluxcd.io/name=<app_name>** 레이블이 있는 Kubernetes 리소스를 찾습니다.
- **ApplicationSet** 을 생성하려면 GitOps 클러스터 리소스 및 GitOps Operator가 설치되어 있어야 합니다. 이러한 사전 요구 사항이 없으면 콘솔에는 **ApplicationSet** 을 생성하기 위해 **Argo 서버** 옵션이 표시되지 않습니다.

중요: 사용 가능한 작업은 할당된 역할을 기반으로 합니다. 역할 기반 액세스 제어 설명서에서 액세스 요구 사항에 대해 알아봅니다.

- **검색에서 리소스 시작을** 클릭하여 관련 리소스를 검색합니다.
- *Search* 를 사용하여 각 리소스의 구성 요소 **종류로** 애플리케이션 리소스를 찾습니다. 리소스를 검색하려면 다음 값을 사용합니다.

1.1.1. Argo CD 애플리케이션 쿼리

Argo CD 애플리케이션을 검색하면 애플리케이션 페이지로 *이동합니다. 검색* 페이지에서 Argo CD 애플리케이션에 액세스하려면 다음 단계를 완료합니다.

1. Red Hat Advanced Cluster Management Hub 클러스터에 로그인합니다.
2. 콘솔 헤더에서 *검색* 아이콘을 선택합니다.
3. **kind:application** 및 **apigroup:argoproj.io** 값으로 쿼리를 필터링합니다.
4. 볼 애플리케이션을 선택합니다. *애플리케이션* 페이지에 애플리케이션에 대한 정보의 개요가 표시됩니다.

검색에 대한 자세한 내용은 콘솔 소개에서 검색을 참조하십시오.

1.2. OPENSHIFT GITOPS OPERATOR에 관리형 클러스터 등록

푸시 모델을 사용하여 GitOps를 구성하려면 Kubernetes 관리 클러스터에 대해 하나 이상의 Red Hat Advanced Cluster Management 세트를 Red Hat OpenShift Container Platform GitOps Operator 인스턴스에 등록할 수 있습니다. 등록한 후에는 해당 클러스터에 애플리케이션을 배포할 수 있습니다. 개발, 스테이징 및 프로덕션 환경의 클러스터 전체에서 애플리케이션 일관성을 자동화하도록 연속 GitOps 환경을 설정합니다.

1.2.1. 사전 요구 사항

1. Red Hat Advanced Cluster Management for Kubernetes에 Red Hat OpenShift GitOps Operator를 설치해야 합니다.
2. 하나 이상의 관리 클러스터를 가져옵니다.

1.2.2. 관리 클러스터를 GitOps에 등록

관리 클러스터를 GitOps에 등록하려면 다음 단계를 완료합니다.

1. 관리형 클러스터 세트 바인딩을 생성하고 관리 클러스터를 해당 관리 클러스터 세트 바인딩에 추가합니다. **multicloud-integrations managedclusterset** 의 관리 클러스터 세트의 예를 참조하십시오.
   자세한 내용은 Creating a *ManagedClusterSet* 설명서를 참조하십시오.
2. Red Hat OpenShift GitOps가 배포된 네임스페이스에 대한 관리형 클러스터 세트 바인딩을 생성합니다. 관리 클러스터를 **openshift-gitops** 네임스페이스에 바인딩하는 예는 **multicloud-integrations** managed clusterset 바인딩 예제를 참조하십시오. *추가 리소스* 섹션에서 *ManagedClusterSetBinding 생성에 대한 자세한 내용은* ***ManagedClusterSetBinding*** *리소스 생성* 을 참조하십시오. 배치 정보는 ManagedClusterSets에서 ManagedClusters 필터링 을 참조하십시오.
3. 관리형 클러스터 세트 바인딩에 사용되는 네임스페이스에서 **배치** 사용자 정의 리소스를 생성하여 OpenShift Container Platform GitOps Operator 인스턴스에 등록할 관리 클러스터 세트를 선택합니다. **multicloud-integration** 배치 예제를 템플릿으로 사용합니다. 배치 정보는 *배치와 함께 ManagedClusterSets 사용을* 참조하십시오.
   참고:
   - OpenShift Container Platform 클러스터만 다른 Kubernetes 클러스터가 아닌 Red Hat OpenShift Container Platform GitOps Operator 인스턴스에 등록됩니다.
   - 일부 불안정한 네트워크 시나리오에서는 관리 클러스터가 일시적으로 사용할 수 **없거나 연결할 수 없는** 상태가 될 수 있습니다. 자세한 내용은 *Red Hat Advanced Cluster Management 및 OpenShift GitOps의 배치 허용 오차 구성* 을 참조하십시오.

4. **GitOpsCluster** 사용자 지정 리소스를 생성하여 배치 결정에서 OpenShift GitOps의 지정된 인스턴스로 관리 클러스터 집합을 등록합니다. 이를 통해 OpenShift GitOps 인스턴스에서 해당 Red Hat Advanced Cluster Management 관리 클러스터에 애플리케이션을 배포할 수 있습니다. ***multicloud-integrations*** GitOps 클러스터 예제를 사용합니다.
참고: 참조된 **배치** 리소스는 **GitOpsCluster** 리소스와 동일한 네임스페이스에 있어야 합니다. 다음 예제를 참조하십시오.

```
apiVersion: apps.open-cluster-management.io/v1beta1
kind: GitOpsCluster
metadata:
  name: gitops-cluster-sample
  namespace: dev
spec:
  argoServer:
    cluster: local-cluster
    argoNamespace: openshift-gitops
  placementRef:
    kind: Placement
    apiVersion: cluster.open-cluster-management.io/v1beta1
    name: all-openshift-clusters (1)
```

(1) **placementRef.name** 값은 **all-openshift-clusters** 이며 **argoNamespace: openshift-gitops** 에 설치된 GitOps 인스턴스의 대상 클러스터로 지정됩니다. **argoServer.cluster** 사양에는 **local-cluster** 값이 필요합니다.

5. 변경 사항을 저장하십시오. GitOps 워크플로에 따라 애플리케이션을 관리할 수 있습니다.

1.2.3. GitOps 토큰

배치 및 **ManagedClusterSetBinding** 사용자 정의 리소스를 통해 GitOps 네임스페이스에 바인딩된 모든 관리 클러스터에 대해 GitOps Operator와 통합하면 **ManagedCluster** 에 액세스할 수 있는 토큰이 있는 시크릿이 네임스페이스에 생성됩니다. 이 작업은 GitOps 컨트롤러에서 리소스를 관리 클러스터에 동기화해야 합니다. 사용자가 애플리케이션 라이프사이클 작업을 수행하기 위해 GitOps 네임스페이스에 대한 관리자 액세스 권한을 부여하면 사용자는 관리 클러스터에 대한 이 시크릿 및 관리 수준에 대한 액세스 권한도 얻을 수 있습니다.

이 작업이 필요하지 않은 경우 사용자를 네임스페이스 범위 admin 역할에 바인딩하는 대신 사용자를 생성하고 바인딩하는 데 사용할 수 있는 애플리케이션 리소스로 작업하는 데 필요한 권한이 있는 더 제한적인 사용자 지정 역할을 사용합니다. 다음 ClusterRole 예제를 참조하십시오.

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: application-set-admin
rules:
- apiGroups:
  - argoproj.io
  resources:
  - applicationsets
  verbs:
  - get
```

```
- list
- watch
- update
- delete
- deletecollection
- patch
```

1.2.4. 추가 리소스

- 자세한 내용은 GitOps에 대한 애플리케이션 배치 허용 오차 구성 을 참조하십시오.
- multicloud-integrations 관리 클러스터 세트 예제를 참조하십시오.
- *ManagedClusterSet*생성을 참조하십시오.
- 다중 클라우드 통합 배치 예제를 참조하십시오.
- 배치 정보는 배치 개요 를 참조하십시오.
- multicloud-integrations GitOps 클러스터 예제를 참조하십시오.
- multicloud-integrations managed cluster set binding 예제를 참조하십시오.
- 자세한 내용은 Creating a *ManagedClusterSetBinding* resource documentation에서 참조하십시오.
- 자세한 내용은 GitOps 정보를 참조하십시오.

1.3. GITOPS에 대한 애플리케이션 배치 허용 오차 구성

Red Hat Advanced Cluster Management를 사용하면 Red Hat OpenShift GitOps에 애플리케이션을 배포하는 관리형 클러스터를 등록할 수 있습니다.

일부 불안정한 네트워크 시나리오에서는 관리 클러스터가 일시적으로 Unavailable 상태가 될 수 있습

니다. 애플리케이션 배포를 용이하게 하는 데 배치 리소스를 사용하는 경우 사용할 수 없는 클러스터를 계속 포함하도록 배치 리소스에 대해 다음 허용 오차를 추가합니다. 다음 예제에서는 허용 오차가 있는 배치 리소스를 보여줍니다.

```
apiVersion: cluster.open-cluster-management.io/v1beta1
kind: Placement
metadata:
  name: placement
  namespace: ns1
spec:
  tolerations:
    - key: cluster.open-cluster-management.io/unreachable
      operator: Exists
    - key: cluster.open-cluster-management.io/unavailable
      operator: Exists
```

1.4. 푸시 및 가져오기 모델을 사용하여 ARGO CD 배포

푸시 모델을 사용하여 허브 클러스터의 Argo CD 서버는 관리 클러스터에 애플리케이션 리소스를 배포합니다. ***Pull** 모델* 의 경우 manifestWork 를 사용하여 ***Propagation** 컨트롤러에서* 관리 클러스터에 의해 애플리케이션 리소스를 전파합니다.

두 모델 모두에서 동일한 ApplicationSet CRD를 사용하여 애플리케이션을 관리 클러스터에 배포합니다.

필수 액세스: 클러스터 관리자

- 사전 요구 사항
- 아키텍처
- ***ApplicationSet*** 사용자 정의 리소스 생성
- ***MulticlusterApplicationSetReport***

1.4.1. 사전 요구 사항

Argo CD Pull 모델에 대한 다음 사전 요구 사항을 확인합니다.

중요:

- **openshift-gitops-ArgoCD-application-controller** 서비스 계정이 클러스터 관리자로 할당되지 *않은* 경우 **GitOps** 애플리케이션 컨트롤러에서 리소스를 배포하지 않을 수 있습니다. 애플리케이션 상태는 다음과 유사한 오류를 보낼 수 있습니다.

```
cannot create resource "services" in API group "" in the namespace
"mortgage",deployments.apps is forbidden: User
"system:serviceaccount:openshift-gitops:openshift-gitops-Argo CD-application-controller"
```

- 관리 클러스터에 **OpenShift Gitops Operator**를 설치한 후 동일한 관리 클러스터에서 **ClusterRoleBinding** 클러스터 관리자 권한을 생성해야 합니다.

- 관리 클러스터에 **ClusterRoleBinding** 클러스터 관리자 권한을 추가하려면 다음 예제 **YAML**을 참조하십시오.

```
kind: ClusterRoleBinding
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: argo-admin
subjects:
  - kind: ServiceAccount
    name: openshift-gitops-argocd-application-controller
    namespace: openshift-gitops
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: cluster-admin
```

- 클러스터 관리자가 아니며 이 문제를 해결해야 하는 경우 다음 단계를 완료합니다.

  1. **Argo CD** 애플리케이션이 배포될 각 관리형 클러스터에서 모든 네임스페이스를 생성합니다.

  2. 각 네임스페이스에 **managed-by** 레이블을 추가합니다. **Argo CD** 애플리케이션이 여러 네임스페이스에 배포된 경우 **Argo CD**에서 각 네임스페이스를 관리해야 합니다.

     managed-by 레이블을 사용하여 다음 예제를 참조하십시오.

```
apiVersion: v1
```

```
kind: Namespace
metadata:
  name: mortgage2
  labels:
    argocd.argoproj.io/managed-by: openshift-gitops
```

1. 애플리케이션의 리포지토리에 있는 모든 애플리케이션 대상 네임스페이스를 선언하고 네임스페이스에 **managed-by** 레이블을 포함해야 합니다. 네임스페이스를 선언하는 방법을 알아보려면 *추가 리소스* 를 참조하십시오.

Argo CD *Pull* 모델을 사용하려면 다음 요구 사항을 참조하십시오.

- **GitOps Operator**는 **hub** 클러스터 및 **openshift-gitops** 네임스페이스의 대상 관리 클러스터에 설치해야 합니다.
- 필수 허브 클러스터 **OpenShift Container Platform GitOps Operator**는 버전 **1.9.0** 이상이어야 합니다.
- 필요한 관리 클러스터 **OpenShift Container Platform GitOps Operator**는 **hub** 클러스터와 동일한 버전이어야 합니다.
- ***ApplicationSet** 컨트롤러가* 관리 클러스터의 **Argo CD** 애플리케이션 템플릿을 전파해야 합니다.
- 모든 관리 클러스터는 **hub** 클러스터의 **Argo CD** 서버 네임스페이스에 클러스터 시크릿이 있어야 합니다. 이 보안은 **ArgoCD** 애플리케이션 세트 컨트롤러에서 관리 클러스터의 **Argo CD** 애플리케이션 템플릿을 전파하는 데 필요합니다.

  클러스터 보안을 생성하려면 배치 리소스에 대한 참조가 포함된 **gitOpsCluster** 리소스를 생성합니다. **placement** 리소스는 **Pull** 모델을 지원하는 데 필요한 모든 관리 클러스터를 선택합니다. **GitOps** 클러스터 컨트롤러가 조정되면 **Argo CD** 서버 네임스페이스에서 관리 클러스터에 대한 클러스터 시크릿을 생성합니다.

1.4.2. 아키텍처

푸시 및 **Pull** 모델의 경우 **hub** 클러스터의 ***Argo CD ApplicationSet** 컨트롤러가* 조정되어 각 대상 관리 클러스터에 대한 애플리케이션 리소스를 생성합니다. 두 모델의 아키텍처에 대한 다음 정보를 참조하십시오.

1.4.2.1. 아키텍처 푸시 모델

- 푸시 모델을 사용하면 **OpenShift Container Platform GitOps**가 중앙 집중식 허브 클러스터에서 관리 클러스터에 *직접* 리소스를 적용합니다.
- **hub** 클러스터에서 실행 중인 **Argo CD** 애플리케이션은 **GitHub** 리포지토리와 통신하고 매니페스트를 관리 클러스터에 직접 배포합니다.
- 푸시 모델 구현에는 **hub** 클러스터에 **Argo CD** 애플리케이션만 포함되어 있으며 관리 클러스터에 대한 인증 정보가 있습니다. **hub** 클러스터의 **Argo CD** 애플리케이션은 애플리케이션을 관리 클러스터에 배포할 수 있습니다.
- 중요: 리소스 애플리케이션이 필요한 다수의 관리형 클러스터에서는 **OpenShift Container Platform GitOps** 컨트롤러 메모리 및 **CPU** 사용량에 대한 잠재적인 부담을 고려하십시오. 리소스 관리를 최적화하려면 리소스 할당량 또는 요청 구성 을 참조하십시오.
- 기본적으로 푸시 모델은 **ApplicationSet** 의 템플릿 섹션에 **apps.open-cluster-management.io/ocm-managed-cluster** 및 **apps.open-cluster-management.io/pull-to-ocm-managed-cluster** 주석을 추가하지 않는 한 애플리케이션을 배포하는 데 사용됩니다.

1.4.2.2. 아키텍처 가져오기 모델

- 풀 모델은 허브 클러스터의 컨트롤러에 대한 부담을 줄임으로써 푸시 모델에 비해 확장성 완화를 제공할 수 있지만 더 많은 요청 및 상태 보고가 필요합니다.
- **Pull** 모델을 사용하면 **OpenShift Container Platform GitOps** *는 중앙 집중식 허브 클러스터에서 관리 클러스터에 직접 리소스를 적용하지 않습니다*. **Argo CD** 애플리케이션은 **hub** 클러스터에서 관리 클러스터로 전파됩니다.
- 가져오기 모델 구현은 **OpenShift Cluster Manager** 등록, 배치 및 **manifestWork API**를 적용하여 허브 클러스터와 관리 클러스터 간에 보안 통신 채널을 사용하여 리소스를 배포할 수 있습니다.
- 각 관리형 클러스터는 리소스 매니페스트를 로컬로 배포하기 위해 **GitHub** 리포지토리와 개별적으로 통신하므로 각 관리 클러스터에 **GitOps Operator**를 설치하고 구성해야 합니다.
- **Argo CD** 서버는 각 대상 관리 클러스터에서 실행 중이어야 합니다. **Argo CD** 애플리케이션

리소스는 관리 클러스터에서 복제되며 로컬 **Argo CD** 서버에서 배포합니다. 관리 클러스터의 분산 **Argo CD** 애플리케이션은 허브 클러스터에 단일 **Argo CD ApplicationSet** 리소스를 사용하여 생성됩니다.

- 관리형 클러스터는 **ocm-managed-cluster** 주석의 값으로 결정됩니다.
- **Pull** 모델을 성공적으로 구현하려면 **Argo CD** 애플리케이션 컨트롤러에서 **ApplicationSet** 의 **template** 섹션에 있는 **argocd.argoproj.io/skip-reconcile** 주석을 사용하여 푸시 모델 애플리케이션 리소스를 *무시* 해야 합니다.
- **Pull** 모델의 경우 관리형 클러스터의 ***Argo CD 애플리케이션 컨트롤러가*** 애플리케이션을 배포하도록 조정됩니다.
- **hub** 클러스터의 **Pull** 모델 *리소스 동기화 컨트롤러는* 각 관리 클러스터의 **OpenShift Cluster Manager** 검색 **V2** 구성 요소를 주기적으로 쿼리하여 각 **Argo CD** 애플리케이션에 대한 리소스 목록 및 오류 메시지를 검색합니다.
- **hub** 클러스터의 *집계 컨트롤러는* 리소스 동기화 컨트롤러의 데이터와 **manifestWork** 의 상태 정보를 사용하여 클러스터 전체에서 **MulticlusterApplicationSetReport** 를 생성하고 업데이트합니다.
- 배포 상태는 **hub** 클러스터로 다시 수집되지만 모든 세부 정보가 전송되지는 않습니다. 개요를 제공하기 위해 추가 상태 업데이트가 주기적으로 스크랩됩니다. 상태 피드백은 실시간이 아니며 각 관리 클러스터 **GitOps Operator**는 **Git** 리포지토리와 통신해야 하므로 여러 요청이 발생합니다.

1.4.3. *ApplicationSet* 사용자 정의 리소스 생성

Argo CD ApplicationSet 리소스는 관리 클러스터 목록을 가져오는 데 사용되는 **generator** 필드에서 **Push** 또는 **Pull** 모델을 사용하여 관리 클러스터에 애플리케이션을 배포하는 데 사용됩니다.

1. **Pull** 모델의 경우 다음 예에 표시된 대로 애플리케이션의 대상을 기본 로컬 **Kubernetes** 서버로 설정합니다. 애플리케이션은 관리 클러스터의 애플리케이션 컨트롤러에서 로컬로 배포합니다.
2. 템플릿 주석과 함께 **Pull** 모델을 사용하는 다음 예제 **ApplicationSet YAML**에 표시된 대로 기본 푸시 모델을 재정의하는 데 필요한 주석을 추가합니다.

```
apiVersion: argoproj.io/v1alpha1
kind: `ApplicationSet`
metadata:
  name: guestbook-allclusters-app-set
  namespace: openshift-gitops
spec:
  generators:
  - clusterDecisionResource:
      configMapRef: ocm-placement-generator
      labelSelector:
        matchLabels:
          cluster.open-cluster-management.io/placement: aws-app-placement
      requeueAfterSeconds: 30
  template:
    metadata:
      annotations:
        apps.open-cluster-management.io/ocm-managed-cluster: '{{name}}' (1)
        apps.open-cluster-management.io/ocm-managed-cluster-app-namespace:
openshift-gitops
        argocd.argoproj.io/skip-reconcile: "true" (2)
      labels:
        apps.open-cluster-management.io/pull-to-ocm-managed-cluster: "true" (3)
      name: '{{name}}-guestbook-app'
    spec:
      destination:
        namespace: guestbook
        server: https://kubernetes.default.svc
      project: default
      sources: [
      {
        repoURL: https://github.com/argoproj/argocd-example-apps.git
        targetRevision: main
        path: guestbook
         }
      ]
      syncPolicy:
        automated: {}
        syncOptions:
        - CreateNamespace=true
```

(1) Pull 모델에는 apps.open-cluster-management.io/ocm-managed-cluster 가 필요합니다.

(2) 푸시 모델 리소스를 무시하려면 argocd.argoproj.io/skip-reconcile 이 필요합니다.

(3) apps.open-cluster-management.io/pull-to-ocm-managed-cluster: "true" 도 Pull 모델에 필요합니다.

1.4.4. *MulticlusterApplicationSetReport*

- **Pull** 모델의 경우 **MulticlusterApplicationSetReport** 은 관리되는 클러스터 전체에서 애플리케이션 상태를 집계합니다.

- 보고서에는 리소스 목록과 각 관리 클러스터의 애플리케이션 전체 상태가 포함됩니다.

- 각 **Argo CD ApplicationSet** 리소스에 대해 별도의 보고서 리소스가 생성됩니다. 보고서는 **ApplicationSet** 과 동일한 네임스페이스에 생성됩니다.

- 보고서에는 다음 항목이 포함됩니다.

  1. **Argo CD** 애플리케이션의 리소스 목록
  2. 각 **Argo CD** 애플리케이션의 전체 동기화 및 상태
  3. 전체 상태가 동기화 되지 않았거나 비정상인각 클러스터에 대한 오류 메시지
  4. 관리 클러스터의 모든 상태 요약 상태

- *리소스 동기화 컨트롤러와 집계 컨트롤러*는 모두 **10**초마다 실행하여 보고서를 생성합니다.

- 다음 예제 출력에 표시된 대로 두 컨트롤러는 **Propagation** 컨트롤러와 동일한 다중 클러스터 통합 **Pod**의 별도의 컨테이너에서 실행됩니다.

```
NAMESPACE                 NAME                                        READY   STATUS
open-cluster-management multicluster-integrations-7c46498d9-fqbq4  3/3     Running
```

다음은 **guestbook** 애플리케이션의 **MulticlusterApplicationSetReport YAML** 파일의 예입니다.

```
apiVersion: apps.open-cluster-management.io/v1alpha1
```

```
kind: MulticlusterApplicationSetReport
metadata:
  labels:
    apps.open-cluster-management.io/hosting-applicationset: openshift-gitops.guestbook-
allclusters-app-set
  name: guestbook-allclusters-app-set
  namespace: openshift-gitops
statuses:
  clusterConditions:
  - cluster: cluster1
    conditions:
    - message: 'Failed sync attempt: one or more objects failed to apply, reason: services is
forbidden: User "system:serviceaccount:openshift-gitops:openshift-gitops-Argo CD-
application-controller" cannot create resource "services" in API group "" in the namespace
"guestbook",deployments.apps is forbidden: User <name> cannot create resource
"deployments" in API group "apps" in the namespace "guestboo...'
      type: SyncError
    healthStatus: Missing
    syncStatus: OutOfSync
  - cluster: pcluster1
    healthStatus: Progressing
    syncStatus: Synced
  - cluster: pcluster2
    healthStatus: Progressing
    syncStatus: Synced
  summary:
    clusters: "3"
    healthy: "0"
    inProgress: "2"
    notHealthy: "3"
    notSynced: "1"
    synced: "2"
```

참고: 리소스를 배포하지 못하면 리소스가 리소스 목록에 포함되지 않습니다. 자세한 내용은 오류 메시지를 참조하십시오.

1.4.5. 추가 리소스

- OpenShift Container Platform 설명서에서 클러스터 구성으로 애플리케이션을 배포하여 OpenShift 클러스터 구성 을 참조하십시오.
- OpenShift Container Platform 설명서에서 Argo CD 인스턴스 설정을 참조하십시오.

1.5. OPENSHIFT CONTAINER PLATFORM GITOPS(ARGO CD)로 정책 정의 관리

더 이상 사용되지 않음: PlacementRule

Argo CD에 따라 OpenShift Container Platform GitOps를 사용하여 정책 정의를 관리할 수 있습니다. 이 워크플로를 허용하려면 Red Hat Advanced Cluster Management Hub 클러스터에서 정책을 생성할 수 있도록 OpenShift Container Platform GitOps 액세스 권한을 부여해야 합니다. 정책 및 배치를 생성, 읽기, 업데이트 및 삭제할 수 있는 액세스 권한을 사용하여 OpenShift Container Platform GitOps에 대한 ClusterRole 리소스를 생성하려면 다음 단계를 완료합니다.

1. 콘솔에서 ClusterRole 을 생성합니다. ClusterRole 은 다음 예와 유사할 수 있습니다.

```
kind: ClusterRole
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: openshift-gitops-policy-admin
rules:
  - verbs:
      - get
      - list
      - watch
      - create
      - update
      - patch
      - delete
    apiGroups:
      - policy.open-cluster-management.io
    resources:
      - policies
      - policysets
      - placementbindings
  - verbs:
      - get
      - list
      - watch
      - create
      - update
      - patch
      - delete
    apiGroups:
      - apps.open-cluster-management.io
    resources:
      - placementrules
  - verbs:
      - get
      - list
      - watch
      - create
      - update
      - patch
      - delete
    apiGroups:
      - cluster.open-cluster-management.io
    resources:
      - placements
```

```
    - placements/status
    - placementdecisions
    - placementdecisions/status
```

2. **OpenShift Container Platform GitOps** 서비스 계정 액세스 권한을 **openshift-gitops-policy-admin ClusterRole** 오브젝트에 부여할 **ClusterRoleBinding** 오브젝트를 생성합니다. **ClusterRoleBinding** 은 다음 예와 유사할 수 있습니다.

```
kind: ClusterRoleBinding
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: openshift-gitops-policy-admin
subjects:
  - kind: ServiceAccount
    name: openshift-gitops-argocd-application-controller
    namespace: openshift-gitops
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: openshift-gitops-policy-admin
```

Red Hat Advanced Cluster Management 정책 정의가 **OpenShift Container Platform GitOps**와 함께 배포되면 각 관리 클러스터 네임스페이스에 정책 사본이 생성됩니다. 이러한 복사본을 복제 정책이라고 합니다. **OpenShift Container Platform GitOps**가 복제된 정책을 반복적으로 삭제하지 못하거나 **ArgoCD** 애플리케이션이 동기화되지 않도록 **argocd.argoproj.io/compare-options: IgnoreExtraneous** 주석은 **Red Hat Advanced Cluster Management** 정책 프레임워크에서 각 복제 정책에 자동으로 설정됩니다.

Argo CD에서 오브젝트를 추적하는 데 사용하는 레이블 및 주석이 있습니다. **Argo CD**에 복제 정책이 표시되지 않는 경우 **Red Hat Advanced Cluster Management** 정책 정의에서 **spec.copyPolicyMetadata** 를 **false** 로 설정하여 **Argo CD** 추적 레이블 및 주석이 복제 정책으로 복사되지 않도록 할 수 있습니다.

1.5.1. OpenShift Container Platform GitOps와 정책 생성기 통합 (Argo CD)

Argo CD에 따라 **OpenShift Container Platform GitOps**를 사용하여 **GitOps**를 통해 정책 생성기를 사용하여 정책을 생성할 수 있습니다. 정책 생성기는 **OpenShift Container Platform GitOps** 컨테이너 이미지에 사전 설치되지 않으므로 일부 사용자 지정이 수행해야 합니다. 계속하려면 **OpenShift Container Platform GitOps Operator**가 **Red Hat Advanced Cluster Management Hub** 클러스터에 설치되어 있어야 하며 **hub** 클러스터에 로그인해야 합니다.

Kustomize를 실행할 때 **OpenShift Container Platform GitOps**가 정책 생성기에 액세스하려면 **Red Hat Advanced Cluster Management Application Subscription** 컨테이너 이미지에서 **OpenShift Container Platform GitOps** 컨테이너에 있는 **Policy Generator** 바이너리를 복사해야 합니다. 또한

Kustomize를 실행할 때 **--enable-alpha-plugins** 플래그를 제공하도록 **OpenShift Container Platform GitOps**를 구성해야 합니다. 다음 단계를 완료합니다.

1. 다음 명령을 사용하여 **OpenShift Container Platform GitOps argocd** 오브젝트 편집을 시작합니다.

```
oc -n openshift-gitops edit argocd openshift-gitops
```

2. 다음과 같은 추가 **YAML** 콘텐츠를 포함하도록 **OpenShift Container Platform GitOps argocd** 오브젝트를 수정합니다. **Red Hat Advanced Cluster Management**의 새로운 주요 버전이 릴리스되고 정책 생성기를 최신 버전으로 업데이트하려면 **Init Container**에서 사용하는 **registry.redhat.io/rhacm2/multicluster-operators-subscription-rhel9** 이미지를 최신 태그로 업데이트해야 합니다. 다음 예제를 보고 **< version >**을 **2.10** 또는 원하는 **Red Hat Advanced Cluster Management** 버전으로 바꿉니다.

```
apiVersion: argoproj.io/v1beta1
kind: ArgoCD
metadata:
  name: openshift-gitops
  namespace: openshift-gitops
spec:
  kustomizeBuildOptions: --enable-alpha-plugins
  repo:
    env:
    - name: KUSTOMIZE_PLUGIN_HOME
      value: /etc/kustomize/plugin
    initContainers:
    - args:
      - -c
      - cp /policy-generator/PolicyGenerator-not-fips-compliant /policy-generator-
tmp/PolicyGenerator
      command:
      - /bin/bash
      image: registry.redhat.io/rhacm2/multicluster-operators-subscription-
rhel9:v<version>
      name: policy-generator-install
      volumeMounts:
      - mountPath: /policy-generator-tmp
        name: policy-generator
    volumeMounts:
    - mountPath: /etc/kustomize/plugin/policy.open-cluster-
management.io/v1/policygenerator
      name: policy-generator
    volumes:
    - emptyDir: {}
      name: policy-generator
```

참고: 또는 **MulticlusterHub** 에 설정된 버전과 일치하도록 **ArgoCD** 매니페스트 및 템플릿이 포함된 **ConfigurationPolicy** 리소스를 생성할 수 있습니다.

```
image: '{{ (index (lookup "apps/v1" "Deployment" "open-cluster-management"
"multicluster-operators-hub-subscription").spec.template.spec.containers 0).image }}'
```

정책을 생성하기 전에 **Kustomize** 디렉터리 내부의 **Helm** 차트 처리를 활성화하려면 **spec.repo.env** 필드에서 환경 변수 **POLICY_GEN_ENABLE_HELM** 을 **"true"** 로 설정합니다.

```
env:
- name: POLICY_GEN_ENABLE_HELM
  value: "true"
```

3. **OpenShift Container Platform GitOps**에서 **Policy Generator**를 사용할 수 있으므로 **OpenShift Container Platform GitOps**에 **Red Hat Advanced Cluster Management Hub** 클러스터에 정책을 생성할 수 있는 액세스 권한이 부여되어야 합니다. 정책 및 배치를 생성, 읽기, 업데이트 및 삭제할 수 있는 액세스 권한으로 **openshift-gitops-policy-admin** 이라는 **ClusterRole** 리소스를 생성합니다. **ealier ClusterRole** 리소스 예제를 참조하십시오.

4. **OpenShift Container Platform GitOps** 서비스 계정 액세스 권한을 **openshift-gitops-policy-admin ClusterRole** 에 부여할 **ClusterRoleBinding** 오브젝트를 생성합니다. **ClusterRoleBinding** 은 다음 리소스와 유사할 수 있습니다.

```
kind: ClusterRoleBinding
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: openshift-gitops-policy-admin
subjects:
  - kind: ServiceAccount
    name: openshift-gitops-argocd-application-controller
    namespace: openshift-gitops
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: openshift-gitops-policy-admin
```

1.5.2. 추가 리소스

- Argo CD 문서를 참조하십시오.

1.6. GITOPS OPERATOR를 설치하기 위한 정책 생성

Red Hat Advanced Cluster Management 정책을 일반적으로 사용하는 것은 하나 이상의 관리형 **Red Hat OpenShift Container Platform** 클러스터에 **Operator**를 설치하는 것입니다. **Policy Generator**를 사

용하여 정책을 생성하고 생성된 정책을 사용하여 **OpenShift Container Platform GitOps Operator**를 설치하는 방법을 계속 읽으십시오.

1.6.1. OpenShift Container Platform GitOps를 설치하는 정책 생성

정책 생성기를 사용하여 **OpenShift Container Platform GitOps**를 설치하는 정책을 생성할 수 있습니다. **OpenShift Container Platform GitOps Operator**는 다음 예제에서 볼 수 있는 *모든 네임스페이스* 설치 모드를 제공합니다. 다음 예와 유사하게 **openshift-gitops-subscription.yaml** 이라는 서브스크립션 매니페스트 파일을 생성합니다.

```
apiVersion: operators.coreos.com/v1alpha1
kind: Subscription
metadata:
  name: openshift-gitops-operator
  namespace: openshift-operators
spec:
  channel: stable
  name: openshift-gitops-operator
  source: redhat-operators
  sourceNamespace: openshift-marketplace
```

특정 버전의 **Operator**에 고정하려면 **spec.startingCSV: openshift-gitops-operator.v<version>** 매개변수와 값을 추가합니다. **&lt ;version& gt;**을 선호하는 버전으로 바꿉니다.

PolicyGenerator 구성 파일이 필요합니다. **policy-generator-config.yaml** 이라는 구성 파일을 사용하여 모든 **OpenShift Container Platform** 관리 클러스터에 **OpenShift Container Platform GitOps**를 설치하는 정책을 생성합니다. 다음 예제를 참조하십시오.

```
apiVersion: policy.open-cluster-management.io/v1
kind: PolicyGenerator
metadata:
  name: install-openshift-gitops
policyDefaults:
  namespace: policies
  placement:
    clusterSelectors:
      vendor: "OpenShift"
  remediationAction: enforce
policies:
  - name: install-openshift-gitops
    manifests:
      - path: openshift-gitops-subscription.yaml
```

마지막 필수 파일은 **kustomization.yaml** 입니다. 이 파일은 다음 구성이 필요합니다.

```
generators:
  - policy-generator-config.yaml
```

생성된 정책은 **PlacementRule**(더 이상 사용되지 않음)를 사용하여 다음 파일과 유사할 수 있습니다.

```
apiVersion: apps.open-cluster-management.io/v1
kind: PlacementRule
metadata:
  name: placement-install-openshift-gitops
  namespace: policies
spec:
  clusterConditions:
    - status: "True"
      type: ManagedClusterConditionAvailable
  clusterSelector:
    matchExpressions:
      - key: vendor
        operator: In
        values:
          - OpenShift
---
apiVersion: policy.open-cluster-management.io/v1
kind: PlacementBinding
metadata:
  name: binding-install-openshift-gitops
  namespace: policies
placementRef:
  apiGroup: apps.open-cluster-management.io
  kind: PlacementRule
  name: placement-install-openshift-gitops
subjects:
  - apiGroup: policy.open-cluster-management.io
    kind: Policy
    name: install-openshift-gitops
---
apiVersion: policy.open-cluster-management.io/v1
kind: Policy
metadata:
  annotations:
    policy.open-cluster-management.io/categories: CM Configuration Management
    policy.open-cluster-management.io/controls: CM-2 Baseline Configuration
    policy.open-cluster-management.io/standards: NIST SP 800-53
    policy.open-cluster-management.io/description:
  name: install-openshift-gitops
  namespace: policies
spec:
  disabled: false
  policy-templates:
    - objectDefinition:
        apiVersion: policy.open-cluster-management.io/v1
        kind: ConfigurationPolicy
        metadata:
          name: install-openshift-gitops
```

```
    spec:
      object-templates:
        - complianceType: musthave
          objectDefinition:
            apiVersion: operators.coreos.com/v1alpha1
            kind: Subscription
            metadata:
              name: openshift-gitops-operator
              namespace: openshift-operators
            spec:
              channel: stable
              name: openshift-gitops-operator
              source: redhat-operators
              sourceNamespace: openshift-marketplace
      remediationAction: enforce
      severity: low
```

OpenShift Container Platform 설명서의 매니페스트에서 생성된 정책이 지원됩니다. OpenShift Container Platform 설명서의 모든 구성 지침은 정책 생성기를 사용하여 적용할 수 있습니다.

1.6.2. *OperatorGroups*에서 정책 종속 항목 사용

OperatorGroup 매니페스트를 사용하여 Operator를 설치할 때 Subscription 을 생성하기 전에 OperatorGroup 이 클러스터에 있어야 합니다. 정책 종속성 기능을 정책 생성기와 함께 사용하여 서브스크립션 정책을 적용하기 전에 OperatorGroup 정책을 준수하는지 확인합니다.

매니페스트를 원하는 순서대로 나열하여 정책 종속 항목을 설정합니다. 예를 들어 먼저 네임스페이스 정책을 생성하고 다음에 OperatorGroup 을 생성하고 서브스크립션 을 마지막으로 생성할 수 있습니다.

policyDefaults.orderManifests 매개변수를 활성화하고 정책 생성기 구성 매니페스트에서 policyDefaults.consolidateManifests 를 비활성화하여 매니페스트 간에 종속성을 자동으로 설정합니다.

1.6.3. 추가 리소스

- Compliance Operator를 설치하는 정책 생성을 참조하십시오.
- 자세한 내용은 GitOps를 사용하여 정책 배포를 참조하십시오.
- 자세한 내용은 OpenShift GitOps 이해 및 Operator 설명서를 참조하십시오.

- 클러스터에 Operator 추가 - CLI를 사용하여 OperatorHub에서 설치
- 자세한 내용은 Compliance Operator 설명서 를 참조하십시오.
- *모든 네임스페이스* 설치 모드를 참조하십시오.
- *네임스페이스가 지정된* 설치 모드를 참조하십시오.
- Pod를 배포하기 전에 Init Container를 사용하여 작업 수행을 참조하십시오.
- Argo CD 를 참조하십시오.
- OpenShift Container Platform에서 지원하는 YAML 입력의 다음 예제를 확인합니다.
  - 설치 후 클러스터 작업
  - 감사 로그 정책 구성
  - 타사 시스템으로 로그 전달 정보

1.7. ARGO CD 내보내기 모델에 대한 사용자 지정 서비스 계정 생성

hub 클러스터에서 managedserviceaccount 리소스를 생성하여 관리 클러스터에 서비스 계정을 생성합니다. clusterpermission 리소스를 사용하여 서비스 계정에 특정 권한을 부여합니다.

Argo CD 내보내기 모델에 사용할 사용자 정의화 서비스 계정을 생성하면 다음과 같은 이점이 있습니다.

- 애플리케이션 관리자 애드온은 각 관리 클러스터에서 실행됩니다. 기본적으로 Argo CD 컨트롤러는 서비스 계정 애플리케이션 관리자를 사용하여 이러한 리소스를 관리 클러스터로 내보냅

니다.

- 애플리케이션 서브스크립션 애드온에서 애플리케이션 관리자 서비스를 사용하여 관리형 클러스터에 애플리케이션을 배포하므로 애플리케이션 관리자 서비스 계정에는 많은 권한이 있습니다. 제한된 권한 집합을 원하는 경우 애플리케이션 관리자 서비스 계정을 사용하지 마십시오.

- **Argo CD** 내보내기 모델을 사용할 다른 서비스 계정을 지정할 수 있습니다. **Argo CD** 컨트롤러가 중앙 집중식 허브 클러스터에서 관리 클러스터로 리소스를 푸시하는 경우 기본 애플리케이션 관리자와 다른 서비스 계정을 사용할 수 있습니다. 다른 서비스 계정을 사용하면 이 서비스 계정에 부여된 권한을 제어할 수 있습니다.

- 서비스 계정이 관리 클러스터에 있어야 합니다. 연결된 권한으로 서비스 계정을 쉽게 생성하려면 중앙 허브 클러스터에서 **managedserviceaccount** 리소스 및 새 **clusterpermission** 리소스를 사용합니다.

다음 절차를 모두 완료한 후 관리 서비스 계정에 클러스터 권한을 부여할 수 있습니다. 클러스터 권한이 있으면 관리 서비스 계정에는 관리 클러스터에 애플리케이션 리소스를 배포하는 데 필요한 권한이 있습니다. 다음 절차를 완료하십시오.

1. 1.7.1절. “관리 서비스 계정 생성”
2. 1.7.2절. “클러스터 권한 생성”
3. 1.7.3절. “GitOpsCluster 리소스에서 관리형 서비스 계정 사용”
4. 1.7.4절. “Argo CD 애플리케이션 생성”
5. 1.7.5절. “정책을 사용하여 관리 서비스 계정 및 클러스터 권한 생성”

1.7.1. 관리 서비스 계정 생성

허브의 **managedserviceaccount** 사용자 지정 리소스는 관리 클러스터에서 **serviceaccounts** 를 편리하게 생성할 수 있는 방법을 제공합니다. **hub** 클러스터의 **<managed _cluster>** 네임스페이스에 **managed serviceccount** 사용자 정의 리소스가 생성되면 관리 클러스터에 **serviceccount** 가 생성됩니다.

관리 서비스 계정을 생성하려면 managedserviceaccount 애드온 활성화를 참조하십시오.

1.7.2. 클러스터 권한 생성

서비스 계정이 생성되면 연결된 권한이 없습니다. 새 서비스 계정에 권한을 부여하려면 **clusterpermission** 리소스를 사용합니다. **clusterpermission** 리소스는 허브의 관리 클러스터 네임스페이스에 생성됩니다. 역할, 관리 클러스터에서 클러스터 역할 리소스를 생성하고 **rolebinding** 또는 **clusterrolebinding** 리소스를 통해 서비스 계정에 바인딩할 수 있는 편리한 방법을 제공합니다.

1. < managed-sa-sample > 서비스 계정 권한을 < managed -sa-sample>의 **mortgage** 네임스페이스에 배포된 샘플 **mortgage** 애플리케이션에 부여하려면 다음 콘텐츠를 사용하여 **YAML**을 생성합니다.

```
apiVersion: rbac.open-cluster-management.io/v1alpha1
kind: ClusterPermission
metadata:
  name: <clusterpermission-msa-subject-sample>
  namespace: <managed cluster>
spec:
  roles:
  - namespace: default
    rules:
    - apiGroups: ["apps"]
      resources: ["deployments"]
      verbs: ["get", "list", "create", "update", "delete", "patch"]
    - apiGroups: [""]
      resources: ["configmaps", "secrets", "pods", "podtemplates",
"persistentvolumeclaims", "persistentvolumes"]
      verbs: ["get", "update", "list", "create", "delete", "patch"]
    - apiGroups: ["storage.k8s.io"]
      resources: ["*"]
      verbs: ["list"]
  - namespace: mortgage
    rules:
    - apiGroups: ["apps"]
      resources: ["deployments"]
      verbs: ["get", "list", "create", "update", "delete", "patch"]
    - apiGroups: [""]
      resources: ["configmaps", "secrets", "pods", "services", "namespace"]
      verbs: ["get", "update", "list", "create", "delete", "patch"]
  clusterRole:
    rules:
    - apiGroups: ["*"]
      resources: ["*"]
      verbs: ["get", "list"]
  roleBindings:
  - namespace: default
    roleRef:
      kind: Role
    subject:
```

```
    apiGroup: authentication.open-cluster-management.io
    kind: ManagedServiceAccount
    name: <managed-sa-sample>
- namespace: mortgage
  roleRef:
    kind: Role
  subject:
    apiGroup: authentication.open-cluster-management.io
    kind: ManagedServiceAccount
    name: <managed-sa-sample>
clusterRoleBinding:
  subject:
    apiGroup: authentication.open-cluster-management.io
    kind: ManagedServiceAccount
    name: <managed-sa-sample>
```

2. **YAML** 파일을 **cluster-permission.yaml** 이라는 파일에 저장합니다.

3. **oc apply -f cluster-permission.yaml** 을 실행합니다.

4. 샘플 **< clusterpermission >**은 **mortgage** 네임스페이스에서 **< clusterpermission-msa-subject-sample >**이라는 역할을 생성합니다. 아직 없는 경우 네임스페이스 **mortgage** 를 생성합니다.

5. **< managed cluster>**에서 생성된 리소스를 검토합니다.

샘플 **< clusterpermission>** 을 생성한 후 샘플 관리 클러스터에 다음 리소스가 생성됩니다.

- 기본 네임스페이스에서 **& lt;clusterpermission-msa-subject-sample>**이라는 하나의 역할입니다.
- 역할을 관리 서비스 계정에 바인딩하기 위해 기본 네임스페이스에서 **< clusterpermission-msa-subject-sample >**이라고 하는 하나의 **roleBinding.**
- **mortgage** 네임스페이스에서 **& lt;clusterpermission-msa-subject-sample>**이라는 하나의 역할입니다.
- 역할을 관리 서비스 계정에 바인딩하기 위해 **mortgage** 네임스페이스에서 **< clusterpermission-msa-subject-sample >**이라고 하는 하나의 **roleBinding.**

- < clusterpermission-msa-subject-sample>이라는 하나의 clusterRole.

- clusterRole을 관리 서비스 계정에 바인딩하기 위해 < clusterpermission-msa-subject-sample >이라는 하나의 clusterRoleBinding.

1.7.3. GitOpsCluster 리소스에서 관리형 서비스 계정 사용

GitOpsCluster 리소스는 배치를 사용하여 클러스터에 액세스하는 데 사용되는 정보가 포함된 Argo CD 클러스터 시크릿 생성을 포함하여 선택한 관리 클러스터를 Argo CD로 가져옵니다. 기본적으로 Argo CD 클러스터 시크릿은 애플리케이션 관리자 서비스 계정을 사용하여 관리 클러스터에 액세스합니다.

1. 관리 서비스 계정을 사용하도록 GitOpsCluster 리소스를 업데이트하려면 관리 서비스 계정의 이름으로 managedServiceAccountRef 속성을 추가합니다.

2. GitOpsCluster 사용자 정의 리소스를 생성하려면 다음 YAML을 Gitops.YAML으로 저장합니다.

```
---
apiVersion: apps.open-cluster-management.io/v1beta1
metadata:
  name: argo-acm-importer
  namespace: openshift-gitops
spec:
  managedServiceAccountRef: <managed-sa-sample>
  argoServer:
    cluster: notused
    argoNamespace: openshift-gitops
  placementRef:
    kind: Placement
    apiVersion: cluster.open-cluster-management.io/v1beta1
    name: all-openshift-clusters
    namespace: openshift-gitops
```

3. YAML 파일을 gitops.yaml 이라는 파일에 저장합니다.

4. oc apply -f gitops.yaml 을 실행합니다.

5. openshift-gitops 네임스페이스로 이동하여 <managed cluster- managed-sa-sample-cluster-secret>이라는 이름으로 새 Argo CD 클러스터 시크릿 이 있는지 확인합니다.

```
% oc get secrets -n openshift-gitops <managed cluster-managed-sa-sample-cluster-secret>
NAME                                TYPE    DATA   AGE
<managed cluster-managed-sa-sample-cluster-secret>   Opaque   3     4m2s
```

1.7.4. Argo CD 애플리케이션 생성

푸시 모델을 사용하여 Argo CD 콘솔에서 Argo CD 애플리케이션을 배포합니다. Argo CD 애플리케이션은 관리 서비스 계정 < managed-sa-sample>과 함께 배포됩니다.

1. Argo CD 콘솔에 로그인합니다.
2. 새 애플리케이션 생성을 클릭합니다.
3. 클러스터 URL을 선택합니다.
4. Argo CD 애플리케이션으로 이동하여 역할 및 클러스터 역할과 같이 < managed cluster>로 전파한 지정된 권한이 있는지 확인합니다.

1.7.5. 정책을 사용하여 관리 서비스 계정 및 클러스터 권한 생성

> When the GitOpsCluster resource is updated with the `managedServiceAccountRef`, each managed cluster in the placement of this GitOpsCluster needs to have the service account. If you have several managed clusters, it becomes tedious for you to create the managed service account and cluster permission for each managed cluster. You can simply this process by using a policy to create the managed service account and cluster permission for all your managed clusters

managedServiceAccount 및 clusterPermission 리소스를 hub 클러스터에 적용하면 이 정책의 배치가 로컬 클러스터에 바인딩됩니다. GitOpsCluster 리소스 배치에서 모든 관리 클러스터의 관리 클러스터 네임스페이스에 해당 리소스를 복제합니다.

정책을 사용하여 managedServiceAccount 및 clusterPermission 리소스를 생성하면 다음 속성이 포함됩니다.

- 정책에서 managedServiceAccount 및 clusterPermission 오브젝트 템플릿을 업데이트하면 모든 관리 클러스터에서 모든 managedServiceAccount 및 clusterPermission 리소스가 업

데이트됩니다.

- managedServiceAccount 및 clusterPermission 리소스로 직접 업데이트하면 정책에 의해 적용되므로 원래 상태로 되돌아갑니다.

- GitOpsCluster 배치에 대한 배치 결정이 변경되면 정책은 관리 클러스터 네임스페이스의 리소스 생성 및 삭제를 관리합니다.

1. YAML에서 관리 서비스 계정 및 클러스터 권한을 생성하는 정책을 생성하려면 다음 콘텐츠를 사용하여 YAML을 생성합니다.

```
apiVersion: policy.open-cluster-management.io/v1
kind: Policy
metadata:
  name: policy-gitops
  namespace: openshift-gitops
  annotations:
    policy.open-cluster-management.io/standards: NIST-CSF
    policy.open-cluster-management.io/categories: PR.PT Protective Technology
    policy.open-cluster-management.io/controls: PR.PT-3 Least Functionality
spec:
  remediationAction: enforce
  disabled: false
  policy-templates:

    - objectDefinition:
        apiVersion: policy.open-cluster-management.io/v1
        kind: ConfigurationPolicy
        metadata:
          name: policy-gitops-sub
        spec:
          pruneObjectBehavior: None
          remediationAction: enforce
          severity: low
          object-templates-raw: |
            {{ range $placedec := (lookup "cluster.open-cluster-management.io/v1beta1"
"PlacementDecision" "openshift-gitops" "" "cluster.open-cluster-
management.io/placement=aws-app-placement").items }}
            {{ range $clustdec := $placedec.status.decisions }}
            - complianceType: musthave
              objectDefinition:
                apiVersion: authentication.open-cluster-management.io/v1alpha1
                kind: ManagedServiceAccount
                metadata:
                  name: <managed-sa-sample>
                  namespace: {{ $clustdec.clusterName }}
                spec:
                  rotation: {}
            - complianceType: musthave
```

```
          objectDefinition:
            apiVersion: rbac.open-cluster-management.io/v1alpha1
            kind: ClusterPermission
            metadata:
              name: <clusterpermission-msa-subject-sample>
              namespace: {{ $clustdec.clusterName }}
            spec:
              roles:
              - namespace: default
                rules:
                - apiGroups: ["apps"]
                  resources: ["deployments"]
                  verbs: ["get", "list", "create", "update", "delete"]
                - apiGroups: [""]
                  resources: ["configmaps", "secrets", "pods", "podtemplates",
"persistentvolumeclaims", "persistentvolumes"]
                  verbs: ["get", "update", "list", "create", "delete"]
                - apiGroups: ["storage.k8s.io"]
                  resources: ["*"]
                  verbs: ["list"]
              - namespace: mortgage
                rules:
                - apiGroups: ["apps"]
                  resources: ["deployments"]
                  verbs: ["get", "list", "create", "update", "delete"]
                - apiGroups: [""]
                  resources: ["configmaps", "secrets", "pods", "services", "namespace"]
                  verbs: ["get", "update", "list", "create", "delete"]
              clusterRole:
                rules:
                - apiGroups: ["*"]
                  resources: ["*"]
                  verbs: ["get", "list"]
              roleBindings:
              - namespace: default
                roleRef:
                  kind: Role
                subject:
                  apiGroup: authentication.open-cluster-management.io
                  kind: ManagedServiceAccount
                  name: <managed-sa-sample>
              - namespace: mortgage
                roleRef:
                  kind: Role
                subject:
                  apiGroup: authentication.open-cluster-management.io
                  kind: ManagedServiceAccount
                  name: <managed-sa-sample>
              clusterRoleBinding:
                subject:
                  apiGroup: authentication.open-cluster-management.io
                  kind: ManagedServiceAccount
                  name: <managed-sa-sample>
        {{ end }}
        {{ end }}
---
```

```
apiVersion: policy.open-cluster-management.io/v1
kind: PlacementBinding
metadata:
  name: binding-policy-gitops
  namespace: openshift-gitops
placementRef:
  name: lc-app-placement
  kind: Placement
  apiGroup: cluster.open-cluster-management.io
subjects:
  - name: policy-gitops
    kind: Policy
    apiGroup: policy.open-cluster-management.io
---
apiVersion: cluster.open-cluster-management.io/v1beta1
kind: Placement
metadata:
  name: lc-app-placement
  namespace: openshift-gitops
spec:
  numberOfClusters: 1
  predicates:
  - requiredClusterSelector:
      labelSelector:
        matchLabels:
          name: local-cluster
```

1. YAML 파일을 policy.yaml 이라는 파일에 저장합니다.

2. oc apply -f policy.yaml 을 실행합니다.

3. 정책의 오브젝트 템플릿에서 GitOpsCluster 관련 배치 결정을 반복하고 다음 managedServiceAccount 및 clusterPermission 템플릿을 적용합니다.