Red Hat Advanced Cluster Management for Kubernetes 2.8

Access control

Access control

Last Updated: 2024-10-18

Red Hat Advanced Cluster Management for Kubernetes 2.8 Access control

Access control

Legal Notice

Abstract

Ensure users have access to resources that are required to perform specific roles.

Table of Contents

CHAPTER 1. ACCESS CONTROL

Access control might need to manually be created and managed. You must configure *authentication* service requirements for Red Hat Advanced Cluster Management for Kubernetes to onboard workloads to Identity and Access Management (IAM). For more information see, *Understanding authentication* in Understanding authentication in the OpenShift Container Platform documentation.

Role-based access control and authentication identifies the user associated roles and cluster credentials. See the following documentation for information about access and credentials.

Required access: Cluster administrator

- Role-based access control
- Implementing role-based access control

1.1. ROLE-BASED ACCESS CONTROL

Red Hat Advanced Cluster Management for Kubernetes supports role-based access control (RBAC). Your role determines the actions that you can perform. RBAC is based on the authorization mechanisms in Kubernetes, similar to Red Hat OpenShift Container Platform. For more information about RBAC, see the OpenShift *RBAC* overview in the OpenShift Container Platform documentation.

Note: Action buttons are disabled from the console if the user-role access is impermissible.

1.1.1. Overview of roles

Some product resources are cluster-wide and some are namespace-scoped. You must apply cluster role bindings and namespace role bindings to your users for consistent access controls. View the table list of the following role definitions that are supported in Red Hat Advanced Cluster Management for Kubernetes:

Table 1.1. Role definition table

Role	Definition
cluster-admin	This is an OpenShift Container Platform default role. A user with cluster binding to the **cluster-admin** role is an OpenShift Container Platform super user, who has all access.
open-cluster-management:cluster-manager-admin	A user with cluster binding to the **open-cluster-management:cluster-manager-admin** role is a Red Hat Advanced Cluster Management for Kubernetes super user, who has all access. This role allows the user to create a **ManagedCluster** resource.

open-cluster-management:admin:<managed_cluster_name>	A user with cluster binding to the **open-cluster-management:admin:<managed_cluster_name>** role has administrator access to the **ManagedCluster** resource named, **<managed_cluster_name>**. When a user has a managed cluster, this role is automatically created.
open-cluster-management:view:<managed_cluster_name>	A user with cluster binding to the **open-cluster-management:view:<managed_cluster_name>** role has view access to the **ManagedCluster** resource named, **<managed_cluster_name>**.
open-cluster-management:managedclusterset:admin:<managed_clusterset_name>	A user with cluster binding to the **open-cluster-management:managedclusterset:admin:<managed_clusterset_name>** role has administrator access to **ManagedCluster** resource named **<managed_clusterset_name>**. The user also has administrator access to **managedcluster.cluster.open-cluster-management.io**, **clusterclaim.hive.openshift.io**, **clusterdeployment.hive.openshift.io**, and **clusterpool.hive.openshift.io** resources, which has the managed cluster set label: **cluster.open-cluster-management.io/clusterset=<managed_clusterset_name>**. A role binding is automatically generated when you are using a cluster set. See Creating a *ManagedClusterSet* to learn how to manage the resource.
open-cluster-management:managedclusterset:view:<managed_clusterset_name>	A user with cluster binding to the **open-cluster-management:managedclusterset:view:<managed_clusterset_name>** role has view access to the **ManagedCluster** resource named, <managed_clusterset_name>`. The user also has view access to **managedcluster.cluster.open-cluster-management.io**, **clusterclaim.hive.openshift.io**, **clusterdeployment.hive.openshift.io**, and **clusterpool.hive.openshift.io** resources, which has the managed cluster set labels: **cluster.open-cluster-management.io**, **clusterset=<managed_clusterset_name>**. For more details on how to manage managed cluster set resources, see Creating a *ManagedClusterSet*.

open-cluster-management:subscription-admin	A user with the **open-cluster-management:subscription-admin** role can create Git subscriptions that deploy resources to multiple namespaces. The resources are specified in Kubernetes resource YAML files in the subscribed Git repository. **Note:** When a non-subscription-admin user creates a subscription, all resources are deployed into the subscription namespace regardless of specified namespaces in the resources. For more information, see the Application lifecycle RBAC section.
admin, edit, view	Admin, edit, and view are OpenShift Container Platform default roles. A user with a namespace-scoped binding to these roles has access to **open-cluster-management** resources in a specific namespace, while cluster-wide binding to the same roles gives access to all of the **open-cluster-management** resources cluster-wide.
open-cluster-management:managedclusterset:bind:<managed_clusterset_name>	A user with the **open-cluster-management:managedclusterset:bind:<managed_clusterset_name>** role has view access to the managed cluster resource called **<managed_clusterset_name>**. The user can bind **<managed_clusterset_name>** to a namespace. The user also has view access to **managedcluster.cluster.open-cluster-management.io**, **clusterclaim.hive.openshift.io**, **clusterdeployment.hive.openshift.io**, and **clusterpool.hive.openshift.io** resources, which have the following managed cluster set label: **cluster.open-cluster-management.io/clusterset=<managed_clusterset_name>**. See Creating a *ManagedClusterSet* to learn how to manage the resource.

Important:

- Any user can create projects from OpenShift Container Platform, which gives administrator role permissions for the namespace.
- If a user does not have role access to a cluster, the cluster name is not displayed. The cluster name might be displayed with the following symbol: **-**.

See Implementing role-based access control for more details.

1.2. IMPLEMENTING ROLE-BASED ACCESS CONTROL

Red Hat Advanced Cluster Management for Kubernetes RBAC is validated at the console level and at the API level. Actions in the console can be enabled or disabled based on user access role permissions.

The multicluster engine operator is a prerequisite and the cluster lifecycle function of Red Hat Advanced Cluster Management. To manage RBAC for clusters with the multicluster engine operator, use the RBAC guidance from the cluster lifecycle multicluster engine for Kubernetes operator Role-based access control documentation.

View the following sections for more information on RBAC for specific lifecycles for Red Hat Advanced Cluster Management:

- Application lifecycle RBAC
  - Console and API RBAC table for application lifecycle
- Governance lifecycle RBAC
  - Console and API RBAC table for governance lifecycle
- Observability RBAC
  - Console and API RBAC table for observability lifecycle

1.2.1. Application lifecycle RBAC

When you create an application, the ***subscription*** namespace is created and the configuration map is created in the ***subscription*** namespace. You must also have access to the ***channel*** namespace. When you want to apply a subscription, you must be a subscription administrator. For more information on managing applications, see Creating an allow and deny list as subscription administrator .

View the following application lifecycle RBAC operations:

- Create and administer applications on all managed clusters with a user named **username**. You must create a cluster role binding and bind it to **username**. Run the following command:

  ```
  oc create clusterrolebinding <role-binding-name> --clusterrole=open-cluster-management:cluster-manager-admin --user=<username>
  ```

  This role is a super user, which has access to all resources and actions. You can create the namespace for the application and all application resources in the namespace with this role.

- Create applications that deploy resources to multiple namespaces. You must create a cluster role binding to the **open-cluster-management:subscription-admin** cluster role, and bind it to a user named **username**. Run the following command:

  ```
  oc create clusterrolebinding <role-binding-name> --clusterrole=open-cluster-management:subscription-admin --user=<username>
  ```

- Create and administer an application named **application-name** in the **cluster-name** managed cluster, with the **username** user. You must create a cluster role binding to the **open-cluster-management:admin:<cluster-name>** cluster role and bind it to **username** by entering the following command:

  ```
  oc create clusterrolebinding <role-binding-name> --clusterrole=open-cluster-management:admin:<cluster-name> --user=<username>
  ```

  This role has read and write access to all **application** resources on the managed cluster, **cluster-name**. Repeat this if access for other managed clusters is required.

- Create a namespace role binding to the **application** namespace using the **admin** role and bind it to **username** by entering the following command:

  ```
  oc create rolebinding <role-binding-name> -n <application-namespace> --clusterrole=admin
  --user=<username>
  ```

  This role has read and write access to all **application** resources in the **application** namspace. Repeat this if access for other applications is required or if the application deploys to multiple namespaces.

- You can create applications that deploy resources to multiple namespaces. Create a cluster role binding to the **open-cluster-management:subscription-admin** cluster role and bind it to **username** by entering the following command:

  ```
  oc create clusterrolebinding <role-binding-name> --clusterrole=open-cluster-
  management:subscription-admin --user=<username>
  ```

- To view an application on a managed cluster named **cluster-name** with the user named **username**, create a cluster role binding to the **open-cluster-management:view:** cluster role and bind it to **username**. Enter the following command:

  ```
  oc create clusterrolebinding <role-binding-name> --clusterrole=open-cluster-
  management:view:<cluster-name> --user=<username>
  ```

  This role has read access to all **application** resources on the managed cluster, **cluster-name**. Repeat this if access for other managed clusters is required.

- Create a namespace role binding to the **application** namespace using the **view** role and bind it to **username**. Enter the following command:

  ```
  oc create rolebinding <role-binding-name> -n <application-namespace> --clusterrole=view --
  user=<username>
  ```

  This role has read access to all **application** resources in the **application** namspace. Repeat this if access for other applications is required.

1.2.1.1. Console and API RBAC table for application lifecycle

View the following console and API RBAC tables for Application lifecycle:

Table 1.2. Console RBAC table for application lifecycle

Resource	Admin	Edit	View
Application	create, read, update, delete	create, read, update, delete	read
Channel	create, read, update, delete	create, read, update, delete	read
Subscription	create, read, update, delete	create, read, update, delete	read

Resource	Admin	Edit	View
Placement rule	create, read, update, delete	create, read, update, delete	read

Table 1.3. API RBAC table for application lifecycle

API	Admin	Edit	View
applications.app.k8s.io	create, read, update, delete	create, read, update, delete	read
channels.apps.open-cluster-management.io	create, read, update, delete	create, read, update, delete	read
deployables.apps.open-cluster-management.io	create, read, update, delete	create, read, update, delete	read
helmreleases.apps.open-cluster-management.io	create, read, update, delete	create, read, update, delete	read
placements.apps.open-cluster-management.io	create, read, update, delete	create, read, update, delete	read
placementrules.apps.open-cluster-management.io (Deprecated)	create, read, update, delete	create, read, update, delete	read
subscriptions.apps.open-cluster-management.io	create, read, update, delete	create, read, update, delete	read
configmaps	create, read, update, delete	create, read, update, delete	read
secrets	create, read, update, delete	create, read, update, delete	read
namespaces	create, read, update, delete	create, read, update, delete	read

1.2.2. Governance lifecycle RBAC

When a policy is created, the policy is created in the cluster. Roles for the governance lifecycle are namespace-scoped. A user must also have access to the managed cluster.

To perform governance lifecycle operations, users must have access to the namespace where the policy is created, along with access to the managed cluster where the policy is applied.

View the following operations:

- To create a policy in the **policy** namespace and apply it in a managed cluster named **cluster-name**, create a namespace role binding to the **policy** namespace using the **open-cluster-management:admin:<cluster-name>** cluster role. Run the following command:

```
oc create rolebinding <role-binding-name> -n <policy-namespace> --clusterrole=open-cluster-management:admin:<cluster-name> --user=<username>
```

- To view a policy in a managed cluster, create a cluster role binding to **open-cluster-management:view:<cluster-name>** cluster role and bind it to the **view** role with the following command:

```
oc create clusterrolebinding <role-binding-name> --clusterrole=open-cluster-management:view:<cluster-name> --user=<username>
```

1.2.2.1. Console and API RBAC table for governance lifecycle

View the following console and API RBAC tables for governance lifecycle:

Table 1.4. Console RBAC table for governance lifecycle

Resource	Admin	Edit	View
Policies	create, read, update, delete	read, update	read
PlacementBindings	create, read, update, delete	read, update	read
Placements	create, read, update, delete	read, update	read
PlacementRules (deprecated)	create, read, update, delete	read, update	read
PolicyAutomations	create, read, update, delete	read, update	read

Table 1.5. API RBAC table for governance lifecycle

API	Admin	Edit	View
policies.policy.open-cluster-management.io	create, read, update, delete	read, update	read
placementbindings.policy.open-cluster-management.io	create, read, update, delete	read, update	read
policyautomations.policy.open-cluster-management.io	create, read, update, delete	read, update	read

1.2.3. Observability RBAC

To view the observability metrics for a managed cluster, you must have **view** access to that managed cluster on the hub cluster. View the following list of observability features:

- Access managed cluster metrics.
  Users are denied access to managed cluster metrics, if they are not assigned to the **view** role for the managed cluster on the hub cluster. Run the following command to verify if a user has the authority to create a **managedClusterView** role in the managed cluster namespace:

  ```
  oc auth can-i create ManagedClusterView -n <managedClusterName> --as=<user>
  ```

  As a cluster administrator, create a **managedClusterView** role in the managed cluster namespace. Run the following command:

  ```
  oc create role create-managedclusterview --verb=create --resource=managedclusterviews -n <managedClusterName>
  ```

  Then apply and bind the role to a user by creating a role bind. Run the following command:

  ```
  oc create rolebinding user-create-managedclusterview-binding --role=create-managedclusterview --user=<user>  -n <managedClusterName>
  ```

- Search for resources.
  To verify if a user has access to resource types, use the following command:

  ```
  oc auth can-i list <resource-type> -n <namespace> --as=<rbac-user>
  ```

  Note: <resource-type> must be plural.

- To view observability data in Grafana, you must have a **RoleBinding** resource in the same namespace of the managed cluster.
  View the following **RoleBinding** example:

  ```
  kind: RoleBinding
  apiVersion: rbac.authorization.k8s.io/v1
  metadata:
  ```

```
  name: <replace-with-name-of-rolebinding>
  namespace: <replace-with-name-of-managedcluster-namespace>
subjects:
 - kind: <replace with User|Group|ServiceAccount>
   apiGroup: rbac.authorization.k8s.io
   name: <replace with name of User|Group|ServiceAccount>
roleRef:
 apiGroup: rbac.authorization.k8s.io
 kind: ClusterRole
 name: view
```

See Role binding policy for more information. See Customizing observability to configure observability.

1.2.3.1. Console and API RBAC table for observability lifecycle

To manage components of observability, view the following API RBAC table:

Table 1.6. API RBAC table for observability

API	Admin	Edit	View
multiclusterobservabilities.observability.open-cluster-management.io	create, read, update, and delete	read, update	read
searchcustomizations.search.open-cluster-management.io	create, get, list, watch, update, delete, patch	-	-
policyreports.wgpolicyk8s.io	get, list, watch	get, list, watch	get, list, watch

Continue to learn about securing your cluster, see Risk and compliance.