Authorize actions in clusters using role-based access control

On this page

- If you want to use the Google Cloud CLI for this task, install and then initialize the gcloud CLI. If you previously installed the gcloud CLI, get the latest version by running `gcloud components update`.

  > ★ **Note:** For existing gcloud CLI installations, make sure to set the **`compute/region`** and **`compute/zone`** properties. By setting default locations, you can avoid errors in gcloud CLI like the following: **`One of [--zone, --region] must be supplied: Please specify location`**.

- Read Best practices for GKE RBAC for guidelines to improve the design of your RBAC policies.

Interaction with Identity and Access Management

★ **Note:** Many failures that appear to be due to authorization are actually caused because the cluster is unable to *authenticate* the client. For example, there are special requirements for authenticating from Compute Engine instances, which are described in Cluster access for kubectl.

In almost all cases, Kubernetes RBAC can be used instead of IAM. GKE users require at minimum, the `container.clusters.get` IAM permission in the project that contains the cluster. This permission is included in the `container.clusterViewer` role, and in other more highly privileged roles. The `container.clusters.get` permission is required for users to *authenticate* to the clusters in the project, but does not *authorize* them to perform any actions inside those clusters. Authorization may then be provided by either IAM or Kubernetes RBAC.

Define and assign permissions

of each role as *granting* access to resources.

Define permissions using Roles or ClusterRoles

You define permissions within a Role or ClusterRole object. A Role defines access to resources within a single Namespace, while a ClusterRole defines access to resources in the entire cluster.

Roles and ClusterRoles have the same syntax. Each has a `rules` section, where you define the resources the rule applies to and allowed operations for the Role. For example, the following Role grants read access (`get` , `watch` , and `list`) to all pods in the `accounting` Namespace:

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

- Non-resource REST Endpoints such as `/healthz`
- Namespaced resources *across all Namespaces* (for example, all Pods across the entire cluster, regardless of Namespace)

Assign Roles using RoleBindings or ClusterRoleBindings

After creating a Role or ClusterRole, you assign it to a user or group of users by creating a RoleBinding or ClusterRoleBinding. Users and groups are called `subjects`, and can be any of the following:

Subject type	Value for `kind`	Value for `name`
Google Cloud user account	`User`	Google Cloud registered email address
Kubernetes service account	`ServiceAccount`	The name of a Kubernetes ServiceAccount object in the cluster

```
  name: johndoe
# IAM service account
- kind: User
  name: test-account@test-project.iam.gserviceaccount.com
# Google Group
- kind: Group
  name: accounting-group@example.com
roleRef:
  kind: Role
  name: pod-reader
  apiGroup: rbac.authorization.k8s.io
```

Verify API access using `kubectl`

`kubectl` provides the `auth can-i` subcommand for quickly querying the API authorization layer. As a platform administrator, you might need to impersonate users to determine what actions they can perform. You can use the `auth can-i` and pass an

permissions, the API server logs an `RBAC DENY` error, along with additional information such as the user's implicit and explicit group membership. If you are using Google Groups for RBAC, `google groups` appears in the log message.

Limitations

The following sections describe interactions that might not seem obvious when working with Kubernetes RBAC and IAM.

Default discovery roles

Clusters are created with a set of default ClusterRoles and ClusterRoleBindings. Requests made with valid credentials are placed in the `system:authenticated` group, whereas all other requests fall into `system:unauthenticated`.

The `system:basic-user` ClusterRole lets users make `SelfSubjectAccessReviews` to

```
Error from server (Forbidden): error when creating ... "role-name" is forbidden:
```

For example, suppose the VM has `cloud-platform` scope but does not have `userinfo-email` scope. When the VM gets an access token, Google Cloud associates that token with the `cloud-platform` scope. When the Kubernetes API server asks Google Cloud for the identity associated with the access token, it receives the service account's unique ID, not the service account's email.

To authenticate successfully, either create a new VM with the `userinfo-email` scope or create a new role binding that uses the unique ID.

To create a new VM instance with the `userinfo-email` scope, run the following command:

```
gcloud compute instances create INSTANCE_NAME \
    --service-account SERVICE_ACCOUNT_EMAIL \
```

```
name: projects/project-name/serviceAccounts/my-iam-account@somedomain.
oauth2ClientId: '123456789012345678901'
projectId: project-name
uniqueId: '123456789012345678901'
```

2. Create a role binding using the `uniqueId` of the service account:

```
kubectl create clusterrolebinding CLUSTERROLEBINDING_NAME \
    --clusterrole cluster-admin \
    --user UNIQUE_ID
```

Permission to create or update roles and role bindings

In Kubernetes, you can only create or update a role or a role binding with specific permissions if you meet the following conditions:

```
Error from server (Forbidden): clusterroles.rbac.authorization.k8s.io "allowed-to
user "caller@example.com" (groups=["system:authenticated"]) is attempting to gran
{APIGroups:[""], Resources:["pods"], Verbs:["list" "get"]}
```

To mitigate this limitation, grant the caller the permissions in the role using RBAC instead of IAM.

You can alternatively use either RBAC or IAM to grant the caller the `escalate` verb, the `bind` verb, or both. However, GKE does not recommend this approach, because the caller can then grant *any* permission to any role.

What's next

- Learn how to create IAM policies.
- Learn how to configure Google Groups for RBAC