Tencent Kubernetes Engine

Access Management

Product Documentation

Contents

Access Management Overview

Last updated : 2019-09-18 17:53:40

If you use Tencent Kubernetes Engine (TKE), and have multiple users managing and sharing your Tencent Cloud account password, you may encounter the following issues:

- Your password is shared by multiple users, leading to high risk of compromise.
- You cannot restrict the access permissions of others, exposing the system to faulty operations that lead to security risks.

To resolve the problems described above, you can use different sub-accounts to implement the management of different projects by different people. By default, sub-accounts do not have permission to use TKE. To do so, you need to create a policy that permits sub-accounts to have all the permissions they need.

Overview

Cloud Access Management (CAM) by Tencent Cloud is a permission and user management system designed for secure and precise products management and access. By using CAM, you can create, manage, and terminate users (groups), and control what actions users and roles can perform and what resources they can access by identity and policy management.

When you use CAM, you can associate a policy with a user or a user group. Policies can allow or forbid users to use the specified resources to complete the specified tasks. For more basic information about CAM policies, see Element Reference. For more information about using CAM policies, see Policies.
If you do not need to manage the access permission to CAM-related resources for sub-accounts, you can skip this chapter. This will not affect your understanding and usage of other parts in this document.

Getting Started

A CAM policy allows or prohibit the use of one or more TKE operations, or must forbid the use of one or more TKE operations. At the same time, it is also necessary to specify the resources that can be

operated (you can specify all resources, or some operations can specify some resources). Policies also can contain conditions for operating resources.

Some TKE APIs do not support resource-level permissions, meaning that, when calling these APIs, you cannot specify specific resources for the operations. Instead, you must specify all resources for the operations.

TKE Resource-level Permission API List

Last updated : 2020-04-26 16:18:04

With resource-level permissions, you can specify the resources that a user can operate on. TKE (formerly CCS) supports some resource-level permissions, where for certain TKE operations, you can control the operations that the user is allow to perform (based on the conditions that must be met) or the resources that the user can use.
The following table describes the types of resources that can be authorized in TKE.

Resource Type	Resource Description Method in the Authorization Policy
Cluster resources	`qcs::ccs:$region::cluster/*`

The following table describes the TKE (Tencent Kubernetes Engines) API operations that currently support resource-level permissions. You can use the wildcard (*) when specifying a resource path.

> **Notes:**
> Only the TKE API operations listed here support resource-level permissions. You can still authorize a user to perform a TKE API operation that does not support resource-level permissions, but you must specify the resource element in the policy statement with the asterisk (*).

API Operation	Resource Path

DescribeClusterService	Cluster resource
	`qcs::ccs:region:account:cluster/*`
	`qcs::ccs:region:account:cluster/$clusterId`

DescribeClusterServiceInfo	Cluster resource
	`qcs::ccs:region:account:cluster/*`
	`qcs::ccs:region:account:cluster/$clusterId`

API Operation	Resource Path

CreateClusterService	Cluster resource
	`qcs::ccs:region:account:cluster/*`
	`qcs::ccs:region:account:cluster/$clusterId`
	CLB resource
	`qcs::clb:$region:$account:clb/*`
	CBS resource
	`qcs::cvm:$region:$account:volume/*`
	`qcs::cvm:$region:$account:volume/$diskId`

ModifyClusterService	Cluster resource
	`qcs::ccs:region:account:cluster/*`
	`qcs::ccs:region:account:cluster/$clusterId`
	CLB resource
	`qcs::clb:$region:$account:clb/*`
	CBS resource
	`qcs::cvm:$region:$account:volume/*`
	`qcs::cvm:$region:$account:volume/$diskId`

DeleteClusterService	Cluster resource
	`qcs::ccs:region:account:cluster/*`
	`qcs::ccs:region:account:cluster/$clusterId`

ModifyServiceDescription	Cluster resource
	`qcs::ccs:region:account:cluster/*`
	`qcs::ccs:region:account:cluster/$clusterId`

DescribeServiceEvent	Cluster resource
	`qcs::ccs:region:account:cluster/*`
	`qcs::ccs:region:account:cluster/$clusterId`

ResumeClusterService	Cluster resource
	`qcs::ccs:region:account:cluster/*`
	`qcs::ccs:region:account:cluster/$clusterId`

PauseClusterService	Cluster resource
	`qcs::ccs:region:account:cluster/*`
	`qcs::ccs:region:account:cluster/$clusterId`

RollBackClusterService	Cluster resource
	`qcs::ccs:region:account:cluster/*`
	`qcs::ccs:region:account:cluster/$clusterId`

API Operation	Resource Path

ModifyClusterServiceImage	Cluster resource
	`qcs::ccs:region:account:cluster/*`
	`qcs::ccs:region:account:cluster/$clusterId`

RedeployClusterService	Cluster resource
	`qcs::ccs:region:account:cluster/*`
	`qcs::ccs:region:account:cluster/$clusterId`

DescribeServiceInstance	Cluster resource
	`qcs::ccs:region:account:cluster/*`
	`qcs::ccs:region:account:cluster/$clusterId`

ModifyServiceReplicas	Cluster resource
	`qcs::ccs:region:account:cluster/*`
	`qcs::ccs:region:account:cluster/$clusterId`

DeleteInstances	Cluster resource
	`qcs::ccs:region:account:cluster/*`
	`qcs::ccs:region:account:cluster/$clusterId`

DescribeClusterNameSpaces	Cluster resource
	`qcs::ccs:region:account:cluster/*`
	`qcs::ccs:region:account:cluster/$clusterId`

CreateClusterNamespace	Cluster resource
	`qcs::ccs:region:account:cluster/*`
	`qcs::ccs:region:account:cluster/$clusterId`

DeleteClusterNamespace	Cluster resource
	`qcs::ccs:region:account:cluster/*`
	`qcs::ccs:region:account:cluster/$clusterId`

DescribeCluster	Cluster resource
	`qcs::ccs:region:account:cluster/*`
	`qcs::ccs:region:account:cluster/$clusterId`

CreateCluster	CVM resource `qcs::cvm:$region:$account:instance/*`

DeleteCluster	Cluster resource
	`qcs::ccs:region:account:cluster/*`
	`qcs::ccs:region:account:cluster/$clusterId`

API Operation	Resource Path

DescribeClusterInstances	Cluster resource
	`qcs::ccs:region:account:cluster/*`
	`qcs::ccs:region:account:cluster/$clusterId`

AddClusterInstances	Cluster resource
	`qcs::ccs:region:account:cluster/*`
	`qcs::ccs:region:account:cluster/$clusterId`
	CVM resource
	`qcs::cvm:$region:$account:instance/*`

DeleteClusterInstances	Cluster resource
	`qcs::ccs:region:account:cluster/*`
	`qcs::ccs:region:account:cluster/$clusterId`
	CVM resource
	`qcs::cvm:$region:$account:instance/*`
	`qcs::cvm:$region:$account:instance/$instanceId`

AddClusterInstancesFromExistedCvm	Cluster resource
	`qcs::ccs:region:account:cluster/*`
	`qcs::ccs:region:account:cluster/$clusterId`
	CVM resource
	`qcs::cvm:$region:$account:instance/*`
	`qcs::cvm:$region:$account:instance/$instanceId`

TKE Image Registry Resource-level Permission Settings

Last updated : 2020-01-02 11:52:26

Overview of the TKE Image Service Permissions

The address format for TKE image is as follows: `ccr.ccs.tencentyun.com/${namespace}/${name}:${tag}` . The following fields are required for configuring the permissions of an image repository:

- `${namespace}` : The namespace to which the image repository belongs.
- `${name}` : The name of the image repository.

> Note:
> Do not include slashes (/) in the namespace `${namespace}` and the image name `${name}` .
> The `${tag}` field currently is only for authenticating the permissions for deleting. For more information, see Image Tag Permissions.
>
> `${namespace}` and `${name}` fields allow you to develop detailed permission schemes for managers to flexibly manage access permissions.
> For example:

- Permit collaborator A to pull images
- Forbid collaborator A from deleting images
- Forbid collaborator B from pulling images in namespace ns1

If you do not need to manage image repository permissions in detail, you can use Presetting Policy Authorization.
If you need to manage image repository permissions in detail, use Customizing Policy Authorization.
The TKE image service utilizes Cloud Access Management (CAM) to manage access permissions. You can learn more about how to use CAM here:

- User management
- Policy management
- Authorization management

Preset Policy Authorization

To simplify TKE image service permission management, the TKE image service has two preset policies:

- Image repository (CCR) full read/write permission
  The preset policy configures all the permissions of the TKE image service. If the collaborator is associated with the preset policy, they will have the same image repository permissions as the administrator. For more information, see Permissions List.
- Image repository read-only permission
  This preset policy includes only the read-only permission for the TKE image service. If a collaborator is **only** associated with this policy in the TKE image service, the following operations will be prohibited:
- Pushing an image using `docker push`
- Creating an image repository namespace
- Deleting an image repository namespace
- Creating an image repository
- Deleting an image repository
- Deleting an image tag

For information about how to associate a preset policy with a collaborator, see the following CAM documents: Preset Policy Overview and Associating a User with a Preset Policy.

Custom Policy Authorization

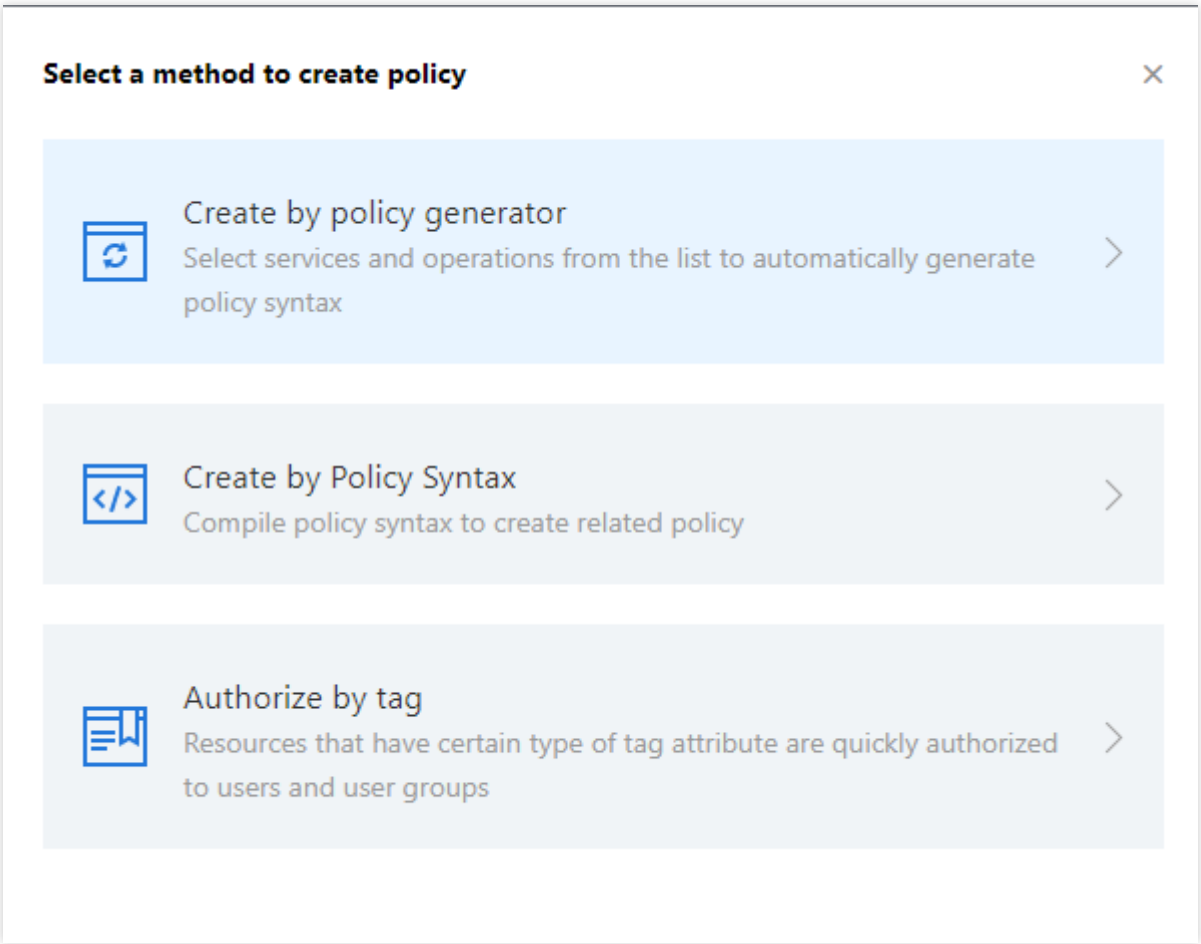
With a custom policy, the manager can associate different permissions with different collaborators. Take the following factors into account when assigning permissions:

- **resource**: Which Image Registries are associated with this permission policy. For example, all Image Registries are described as `qcs::ccr:::repo/*` . For more information, see CAM Resource Description Method.
- **action**: What operations, such as deleting and creating, this permission policy allows the collaborators to perform on the **resource**. The operations are usually described using APIs.
- **effect**: Whether this permission policy allows collaborators to perform such operations.

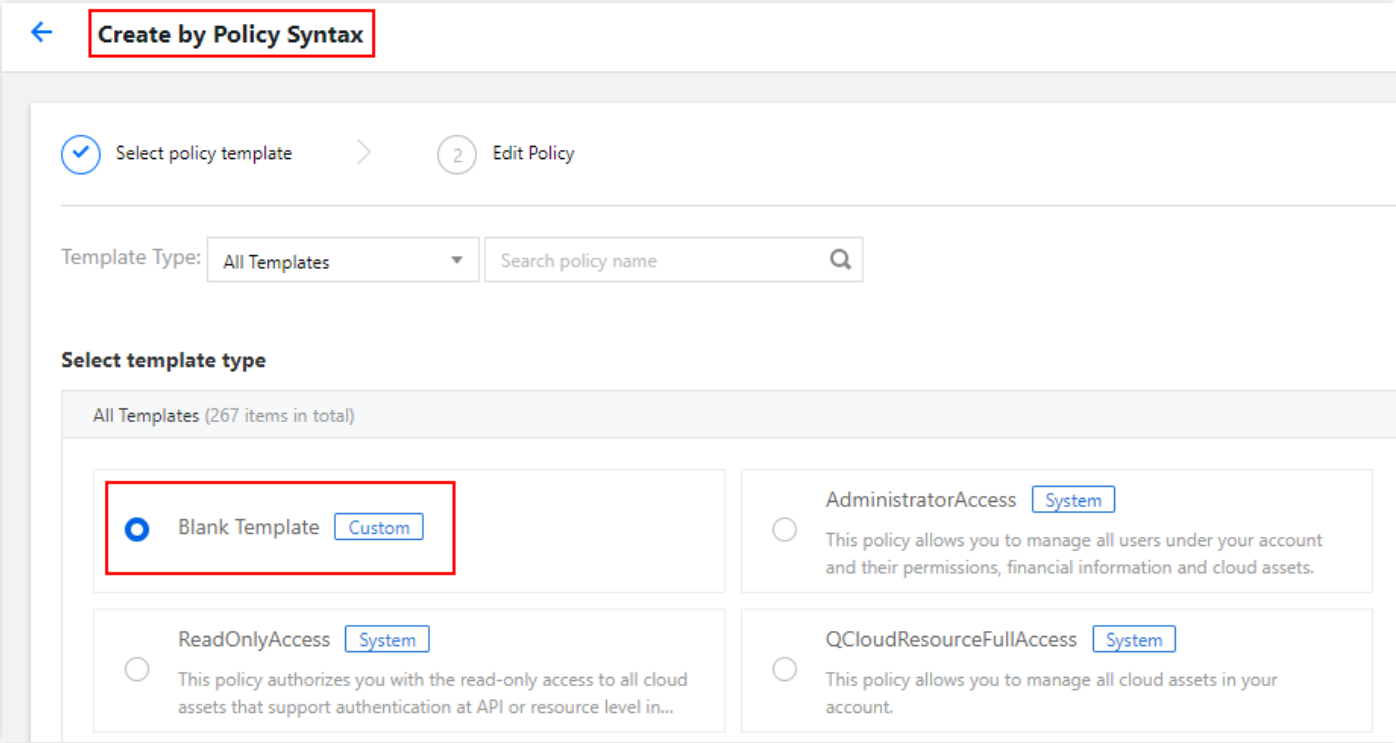
When you have planned the permission settings, you can assign the permissions. The following example shows how to permit collaborators to create an image repository:

1. Create a custom policy (see the CAM document).

2. Log in to the Tencent Cloud Console using your developer account.
3. Go to the CAM custom policy management page and click **Create a custom policy** to open the **Select a policy creation method** dialog box. This is shown in the following figure:



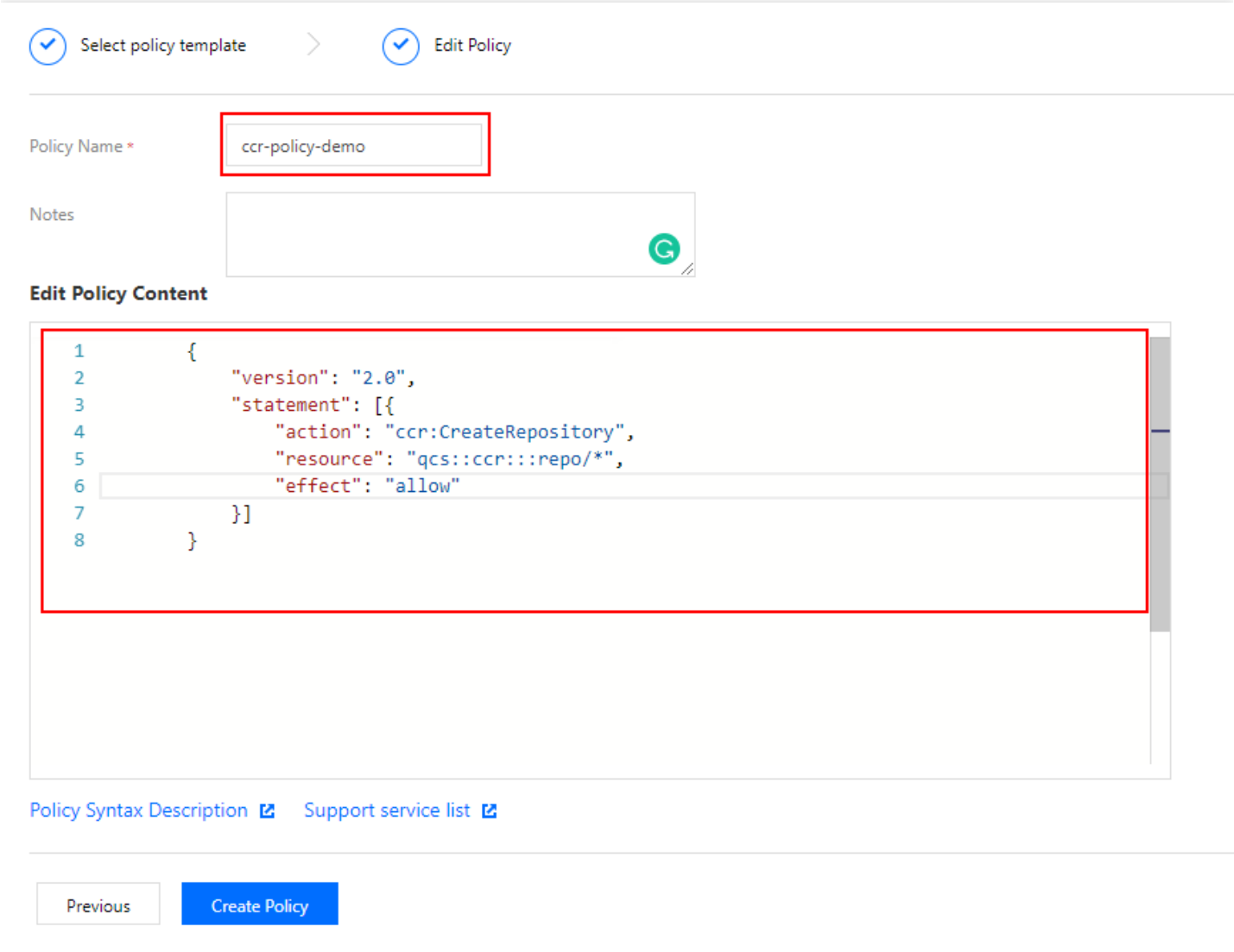
4. Select **Create by Policy Syntax**>**Blank Template**.



5. Click **Next Step** to enter the **Edit Policy** page.
6. Set the policy name, and enter the following content into the **Edit Policy Content** editing box.

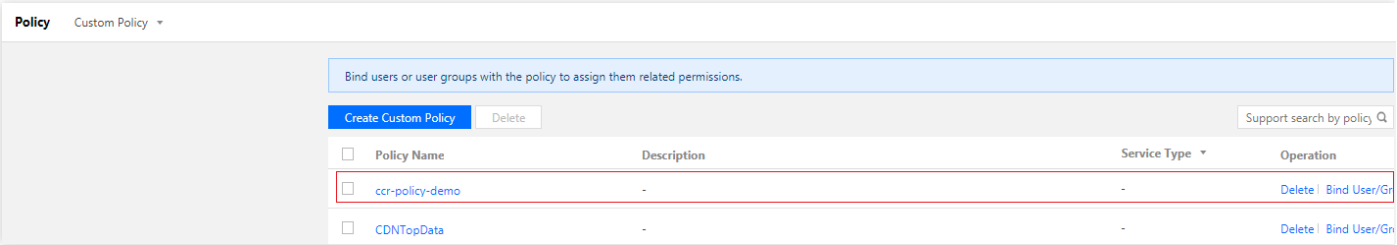
```
{
"version": "2.0",
"statement": [{
"action": "ccr:CreateRepository",
"resource": "qcs::ccr:::repo/*",
"effect": "allow"
}]
}
```

For example, set the policy name to `ccr-policy-demo` , as shown in the following figure:



> At the **end** of "resource", use * to indicate that an image repository can be created under any namespace.

6. Click **Create Policy** to complete the policy creation process.



7. Associate a custom policy. After the policy (`ccr-policy-demo`) is created in step 1, you can associate it with any collaborator. For more information, see the CAM Documentation. After the policy has been associated, the collaborators have **create image repository permissions in any namespace**.
   _resource `qcs::ccr:::repo/*` Format description:

- `qcs::ccr:::` is a fixed format, indicating the developer's TKE image repository service.
- `repo` is a fixed prefix, representing the resource type, which is an image repository here.
- `*` after the slash (`/`) means matching all image repositories.

For a detailed description of resource, see CAM Resource Description Method.

Authorizing by Resource

You can authorize multiple resources at the same time. For example, **to allow deletion of image repositories in namespace foo and bar**, you can create the following custom policy:

```
{
"version": "2.0",
"statement": [{
"action": [
"ccr:BatchDeleteRepository",
"ccr:DeleteRepository"
],
"resource": [
"qcs::ccr:::repo/foo/*",
"qcs::ccr:::repo/bar/*"
],
"effect": "allow"
}]
}
```

- `foo/*` in `qcs::ccr:::repo/foo/*` means all images in the image repository namespace `foo` .
- `bar/*` in `qcs::ccr:::repo/bar/*` means all images in the image repository namespace `bar` .

Authorizing by Action (API)

You can configure multiple `actions` for a resource for a centralized management of resource permissions. For example, to **permit the creation, deletion and pushing of image repository**

in the namespace foo, you can create the following custom policies:

```
{
"version": "2.0",
"statement": [{
"action": [
"ccr:CreateRepository",
"ccr:BatchDeleteRepository",
"ccr:DeleteRepository",
"ccr:push"
],
"resource": "qcs::ccr:::repo/foo/*",
"effect": "allow"
}]
}
```

Permission List

Docker Client Permissions

resource : `qcs::ccr:::repo/${namespace}/${name}`

action:

- `ccr:pull` : Use the Docker command line to pull an image
- `ccr:push` : Use the Docker command line to push an image

Namespace Permissions

resource: `qcs::ccr:::repo/${namespace}`

action:

- `ccr:CreateCCRNamespace` Create an image repository namespace
- `ccr:DeleteUserNamespace` Delete an image repository namespace

Function Guide: **TKE** > Left sidebar **Image Repositories** > **My Images** > **Namespaces**.

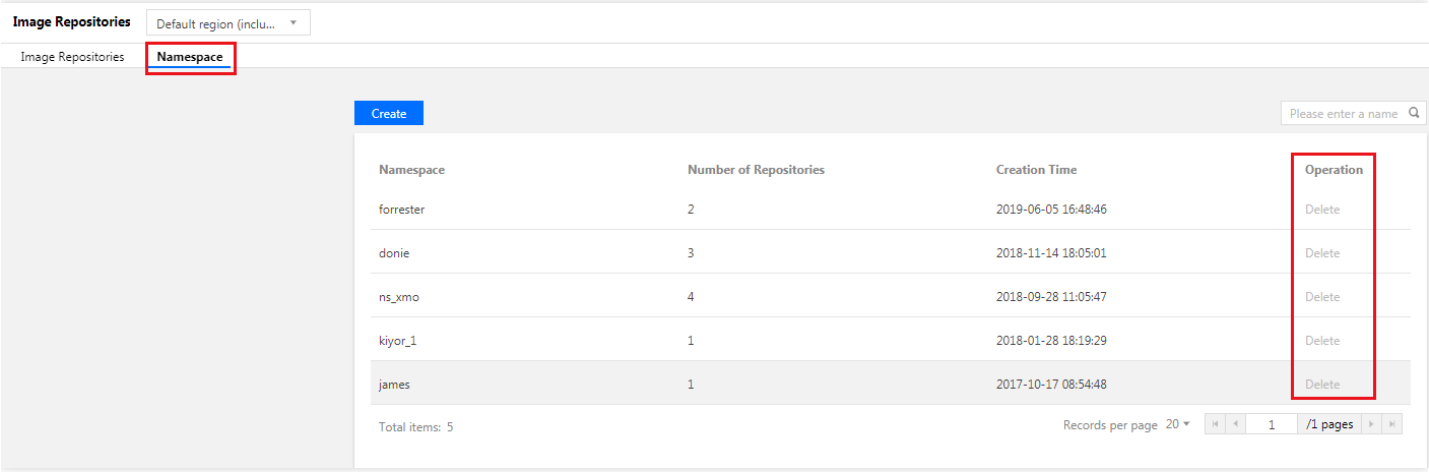


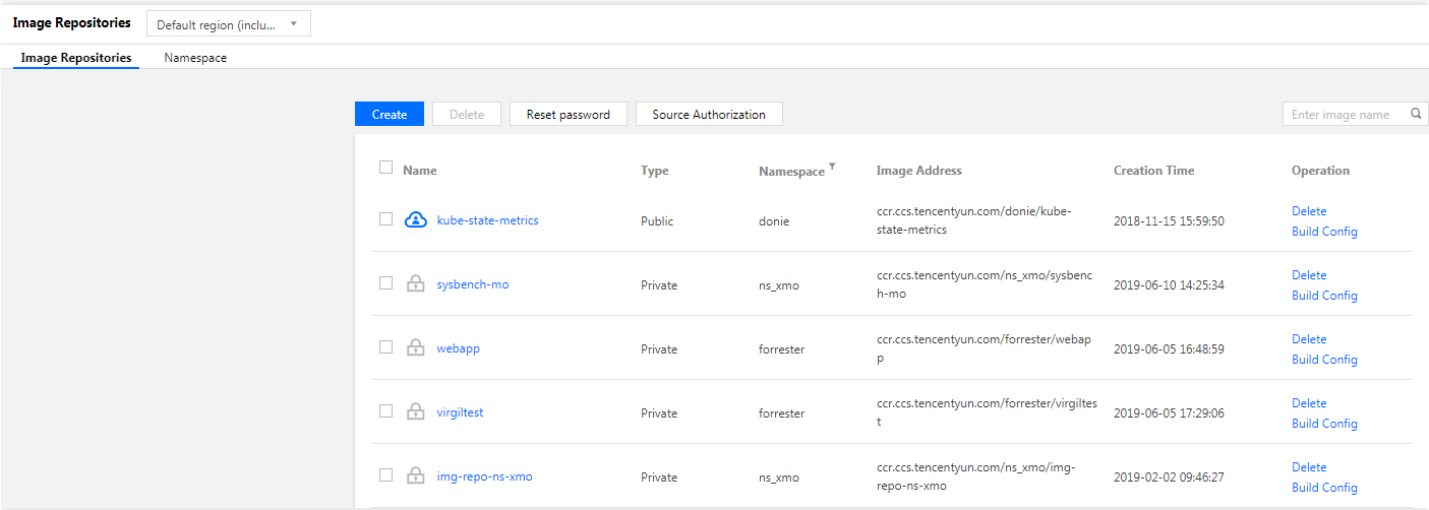
Image Repository Permissions

resurce: `qcs::ccr:::repo/${namespace}/${name}`

action:

- `ccr:CreateRepository` Create an image repository
- `ccr:DeleteRepository` Delete an image repository
- `ccr:BatchDeleteRepository` Batch delete image repositories
- `ccr:GetUserRepositoryList` View the list of image repositories

Function Guide: **TKE** > Left sidebar **Image Repositories** > **My Images** > **My Images**.



> Note:
> If you want to prevent a collaborator from deleting certain images, configure multiple actions.

For example, to prohibit deleting any image repository:

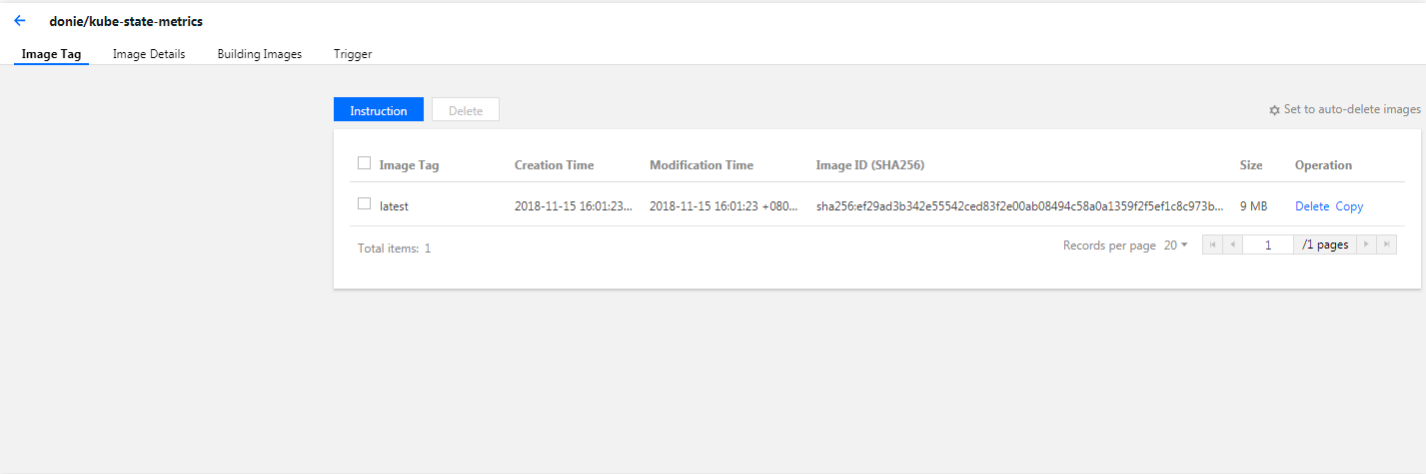
```
{
"version": "2.0",
"statement": [{
"action": [
"ccr:BatchDeleteRepository",
"ccr:DeleteRepository"
],
"resource": "qcs::ccr:::repo/*",
"effect": "deny"
}]
}
```

Image Tag Permissions

resource: `qcs::ccr:::repo/${namespace}/${name}:${tag}`

action : `ccr:DeleteTag` Delete image tag permissions

Function Guide: **TKE** > Left sidebar **Image Repositories** > **My Images** > **My Images** > Click an image name > **Image Tag** page.



Usage Examples
Configuring a Sub-account's Administrative Permissions to a Single TKE Cluster

Last updated : 2019-07-19 17:54:25

Operation Scenario

You can grant a user the permissions to view and use specific resources in the TKE console by using a CAM policy. The examples in this document guide you through the process of configuring a single cluster in the console.

Directions

Configuring Full Read/write Permission for a Single Cluster

1. Log in to the CAM console.
2. In the left sidebar, click Policies to go to the policy management page.
3. Click **Create a custom policy** and select the "Create by policy syntax" method.
4. Select the "Blank template" type and click **Next**.
5. Enter a custom policy name and replace "Edit policy content" with the following.

```
{
”version”: ”2.0”,
”statement”: [
{
”action”: [
”ccs:*”
],
”resource”: [
”qcs::ccs:sh::cluster/cls-XXXXXXX”, // Replace with the cluster in the specified region for wh
ich you want to grant permissions
”qcs::cvm:sh::instance/*”
],
”effect”: ”allow”
},
{
```

```
"action": [
"cvm:*"
],
"resource": "*",
"effect": "allow"
},
{
"action": [
"vpc:*"
],
"resource": "*",
"effect": "allow"
},
{
"action": [
"clb:*"
],
"resource": "*",
"effect": "allow"
},
{
"action": [
"monitor:*",
"cam:ListUsersForGroup",
"cam:ListGroups",
"cam:GetGroup",
"cam:GetRole"
],
"resource": "*",
"effect": "allow"
}
]
}
```

6. In "Edit policy content", change `qcs::ccs:sh::cluster/cls-XXXXXXX` to the cluster in the specified region for which you want to grant permissions. See the figure below:
   For example, if you need to grant full read/write permission for the cls-69z7ek9l cluster in

Guangzhou, change `qcs::ccs:sh::cluster/cls-XXXXXXX` to `"qcs::ccs:gz::cluster/cls-69z7ek9l"`.

Edit Policy Content

```
1  {
2      "version": "2.0",
3      "statement": [
4          {
5              "action": [
6                  "ccs:*"
7              ],
8              "resource": [
9                  "qcs::ccs:gz::cluster/cls-69z7ek9l", // Replace with the cluster in the spe
10                 "qcs::cvm:sh::instance/*"
11             ],
12             "effect": "allow"
13         },
14         {
15             "action": [
16                 "cvm:*"
```

> Replace with the ID of the cluster ID in the specified region for which you want to grant permissions. If you want to allow sub-accounts to scale the cluster, you also need to configure the user payment permission for the sub-accounts.

7. Click **Create a policy** to complete the configuration of full read/write permission for a single cluster.

Configuring Read-only Permission for a Single Cluster

1. Log in to the CAM console.
2. In the left sidebar, click Policies to go to the policy management page.
3. Click **Create a custom policy** and select the "Create by policy syntax" method.
4. Select the "Blank template" type and click **Next**.
5. Enter a custom policy name and replace "Edit policy content" with the following.

```
{
"version": "2.0",
"statement": [
{
"action": [
"ccs:Describe*",
"ccs:Check*"
],
"resource": "qcs::ccs:gz::cluster/cls-1xxxxxx", // Replace with the cluster in the specified r
```

```
egion for which you want to grant permissions
”effect”: ”allow”
},
{
”action”: [
”cvm:Describe*”,
”cvm:Inquiry*”
],
”resource”: ”*”,
”effect”: ”allow”
},
{
”action”: [
”vpc:Describe*”,
”vpc:Inquiry*”,
”vpc:Get*”
],
”resource”: ”*”,
”effect”: ”allow”
},
{
”action”: [
”clb:Describe*”
],
”resource”: ”*”,
”effect”: ”allow”
},
{
”effect”: ”allow”,
”action”: [
”monitor:*”,
”cam:ListUsersForGroup”,
”cam:ListGroups”,
”cam:GetGroup”,
”cam:GetRole”
],
”resource”: ”*”
}
]
}
```

6. In "Edit policy content", change `qcs::ccs:gz::cluster/cls-1xxxxxx` to the cluster in the specified region for which you want to grant permissions. See the figure below:
   For example, if you need to grant ready-only permission for the cls-19a7dz9c cluster in Beijing,

change `qcs::ccs:gz::cluster/cls-1xxxxxx` to `qcs::ccs:bj::cluster/cls-19a7dz9c` .

Edit Policy Content

```
1  {
2      "version": "2.0",
3      "statement": [
4          {
5              "action": [
6                  "ccs:Describe*",
7                  "ccs:Check*"
8              ],
9              "resource": "qcs::ccs:bj::cluster/cls-19a7dz9c"  // Replace with the cluster in
10             "effect": "allow"
11         },
12         {
13             "action": [
14                 "cvm:Describe*",
15                 "cvm:Inquiry*"
16             ],
```

Replace with the ID of the cluster ID in the specified region for which you want to grant permissions.

7. Click **Create a policy** to complete the configuration of read-only permission for a single cluster.

Configuring a Sub-account's Full Read/write or Read-only Permission to TKE

Last updated : 2020-02-24 16:43:58

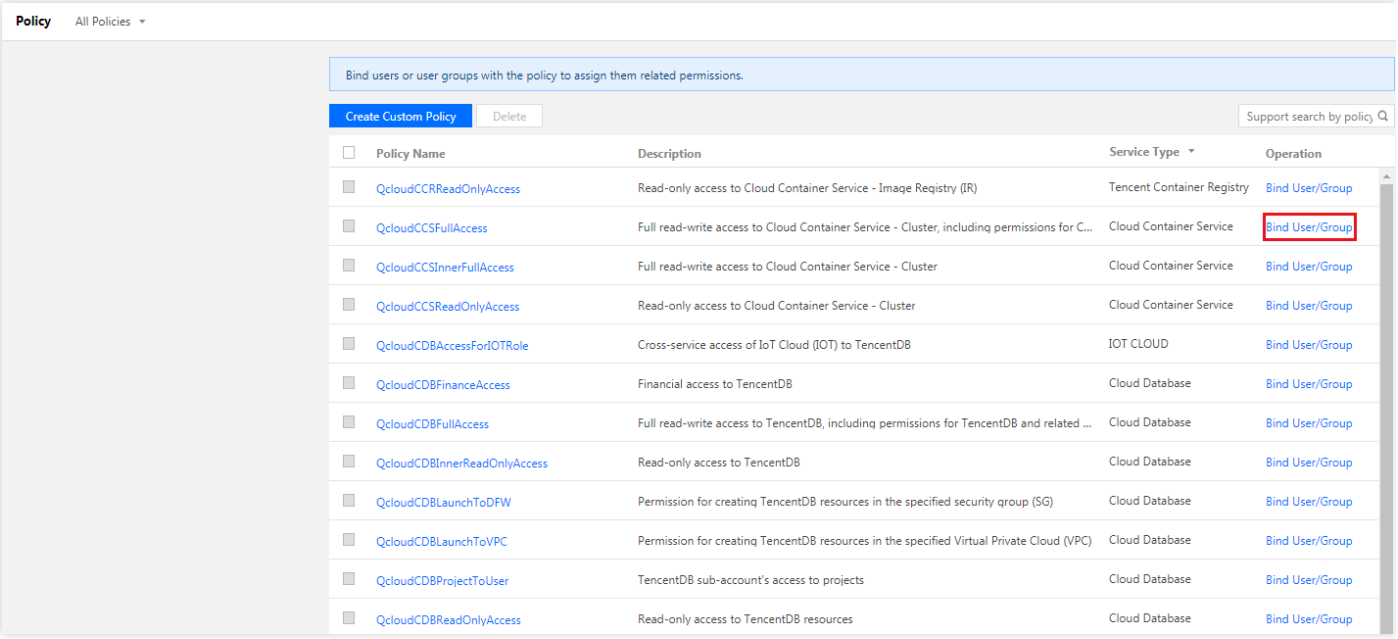
Operation Scenario

You can grant a user the permissions to view and use specific resources in the TKE console by using a CAM policy. The examples in this document guide you through the process of configuring certain permissions in the console.

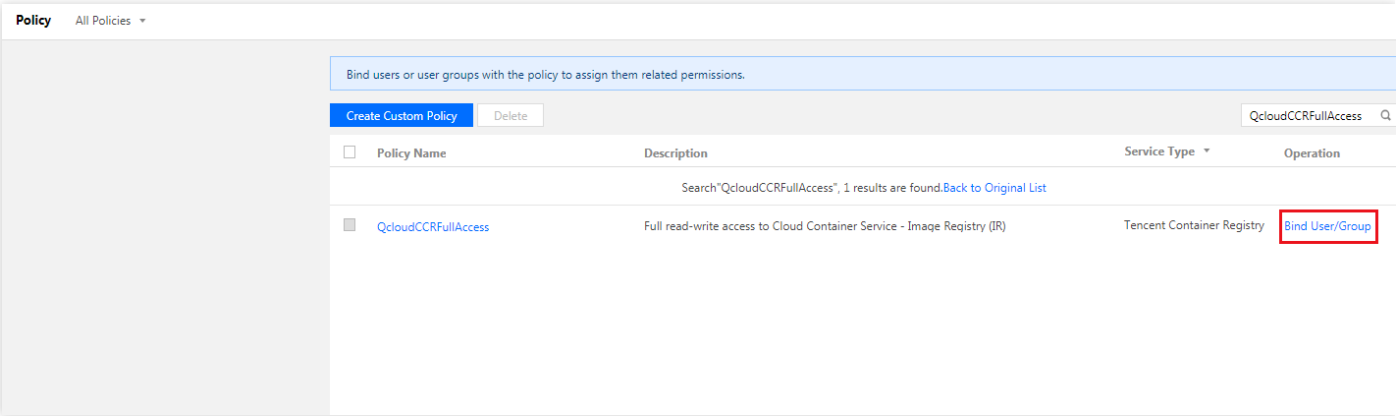
Steps

Configuring Full Read/write Permission

1. Log in to the CAM console.
2. In the left navigation pane, click Policies to go to the policy management page.
3. On the "Policy management" page, click **Associate a user/group** in the row of **QcloudCCSFullAccess** policy. See the figure below:



4. In the **Associate a user/user** window that pops up, select the account that needs full read/write permission for the TKE service, and click **OK** to grant full read/write permission for the TKE service to the sub-accounts.
5. On the "Policy management" page, click **Associate a user/group** in the row of **QcloudCCRFullAccess** policy. See the figure below:

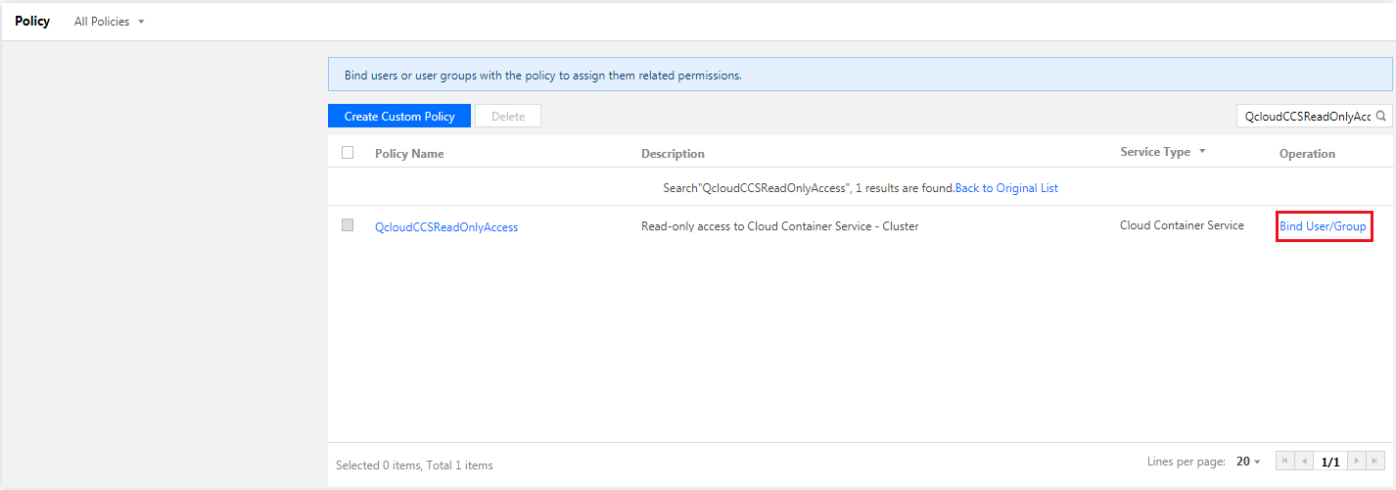


6. In the **Associate a user/group** window that pops up, select the account that needs full read/write permission for Image Registry, and click **OK** to grant full read/write permission for Image Registry to the sub-accounts.

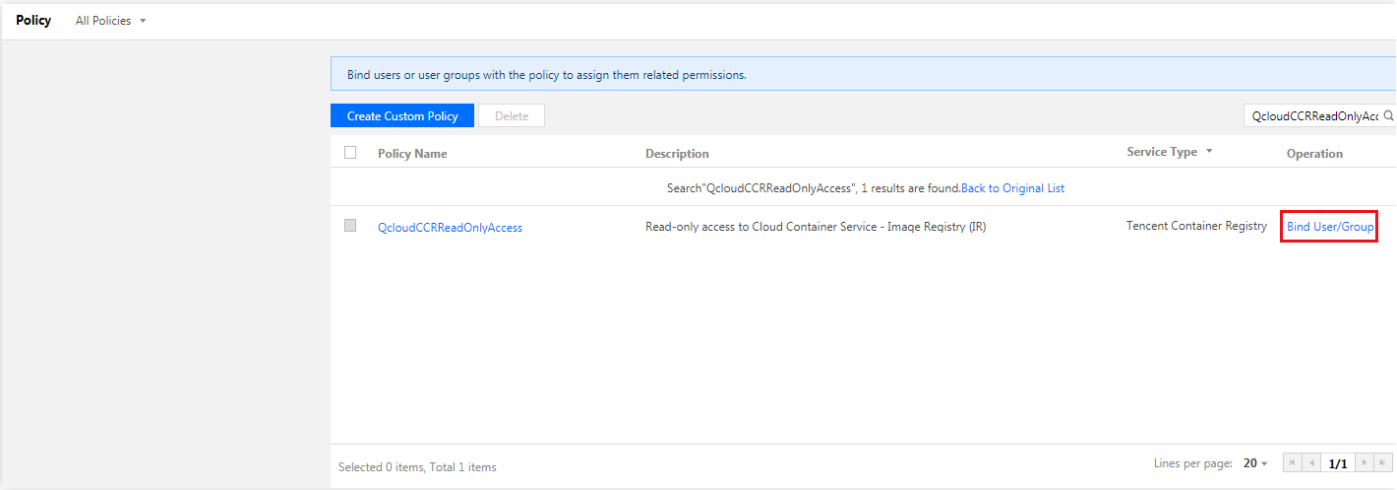
> If you want to use the trigger and automatic building features of Image Registry, you also need to configure additional permissions for TKE - continuous integration (CCB).

Configuring Read-only Permission

1. Log in to the CAM console.
2. In the left navigation pane, click Policies to go to the policy management page.
3. On the "Policy management" page, click **Associate a user/group** in the row of **QcloudCCSReadOnlyAccess** policy. See the figure below:



4. In the **Associate a user/user** window that pops up, select the account that needs read-only permission for the TKE service, and click **OK** to grant read-only permission for the TKE service to the sub-accounts.
5. On the "Policy management" page, click **Associate a user/group** in the row of **QcloudCCRReadOnlyAccess** policy. See the figure below:



6. In the **Associate a user/group** window that pops up, select the account that needs read-only permission for Image Registry, and click **OK** to grant read-only permission for Image Registry to the sub-accounts.

> If you want to use the trigger and automatic building features of Image Registry, you also need to configure additional permissions for TKE - continuous integration (CCB).