Tencent Kubernetes Engine
Permission Management
Product Documentation

Contents

Permission Management

Overview

Last updated：2022-06-10 16:48:46

If you have multiple users managing the TKE service, and they all share your Tencent Cloud account access key, you may face the following problems:

- The risk of your key being compromised is high since multiple users are sharing it.
- Your users might introduce security risks from maloperations due to the lack of user access control.

To solve these problems, create sub-accounts for other users and these users use sub-accounts to log in and manage their services. By default, sub-accounts do not have permission to use TKE. You need to create a policy to grant the required permissions to sub-accounts.

Overview

Cloud Access Management (CAM) is a web-based Tencent Cloud service that helps you securely manage and control access permissions of your Tencent Cloud resources. Using CAM, you can create, manage, and terminate users (groups), and control the Tencent Cloud resources that can be used by the specified user through identity and policy management.

When using CAM, you can associate a policy with a user or user group to allow or forbid them to use specified resources to complete specified tasks. For more information on CAM policies, see Element Reference. For more information on how to use CAM policies, see Policy.
You can skip this section if you don't need to manage permissions to CAM resources for sub-accounts. This will not affect your understanding and use of the other sections of the document.

Getting Started

A CAM policy must authorize or deny the use of one or more TKE operations. At the same time, it must specify the resources that can be used for the operations (which can be all resources or partial resources for certain operations). A policy can also include the conditions set for the manipulated resources.

Some TKE APIs do not support resource-level permissions. This means that you cannot specify certain resources when performing such API operations, and these operations are performed on all the resources.

Description of Role Permissions Related to Service Authorization

Last updated：2022-05-09 11:40:29

When you use Tencent Kubernetes Engine (TKE), you need to authorize services to use relevant cloud resources. Each scenario usually contains policies that are defined for different roles in advance. The main roles involved are `TKE_QCSRole` and `IPAMDofTKE_QCSRole` . This document introduces the details of each authorization policy, and the authorization scenarios and authorization steps for each role.

> Note：
> The sample role in this document does not contain the authorization policy related to container image repositories. For more information about TKE image related permissions, see TKE Image Registry Resource-level Permission Settings.

TKE_QCSRole

After TKE is activated, Tencent Cloud grants your account the permissions of the role `TKE_QCSRole` , which is associated with multiple preset policies by default. To obtain relevant permissions, you need to perform the corresponding preset policy authorization operations in specific authorization scenarios. After these operations are completed, the corresponding policy will appear in the role's list of authorized policies. The preset policies associated with `TKE_QCSRole` by default include:

The default associated preset policies

- `QcloudAccessForTKERole` : The permission for TKE to access cloud resources
- `QcloudAccessForTKERoleInOpsManagement` : The permission for Ops management, including the log service

Other associated preset policies

- `QcloudAccessForTKERoleInCreatingCFSStorageclass` : The permission for TKE to operate on Cloud File Storage (CFS), including adding/deleting/querying CFS systems, and querying the mount targets of a file system.
- `QcloudCVMFinanceAccess` : CVM finance permission

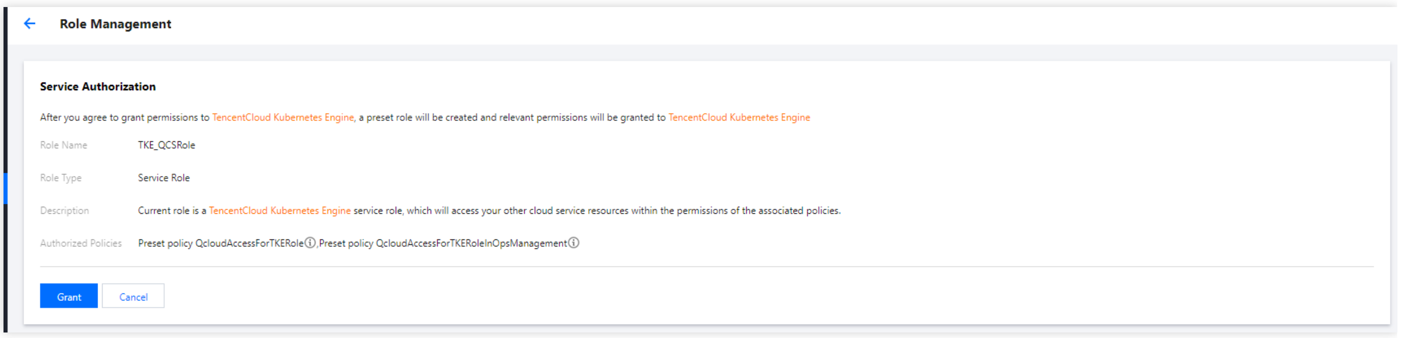
Preset policy QcloudAccessForTKERole

Authorization scenario

When you log in to the TKE console for the first time after registering and logging in to a Tencent Cloud account, you need to go to the "Cloud Access Management" page to grant the current account TKE permissions for operating on CVMs, CLBs, CBS, and other cloud resources.

Authorization steps

1. Log in to the TKE console and click **Cluster** in the left sidebar to pop up the **Service authorization** window.
2. Click **Go to Cloud Access Management** to enter the **Role management** page.
3. Click **Grant** to complete authentication.



Service Authorization

After you agree to grant permissions to TencentCloud Kubernetes Engine, a preset role will be created and relevant permissions will be granted to TencentCloud Kubernetes Engine

Role Name	TKE_QCSRole
Role Type	Service Role
Description	Current role is a TencentCloud Kubernetes Engine service role, which will access your other cloud service resources within the permissions of the associated policies.
Authorized Policies	Preset policy QcloudAccessForTKERole,Preset policy QcloudAccessForTKERoleInOpsManagement

Permission content

- CVM

Permission Name	Permission Description
`cvm:DescribeInstances`	Querying the list of server instances
`cvm:*Cbs*`	CBS-related permissions

- Tag

Permission Name	Permission Description
`tag:*`	All features related to tags

- CLB

Permission Name	Permission Description
`clb:*`	All features related to CLB

- TKE

Permission Name	Permission Description
`ccs:DescribeCluster`	Querying a cluster list
`ccs:DescribeClusterInstances`	Querying cluster node information

Preset policy QcloudAccessForTKERoleInOpsManagement

Authorization scenario

This policy is associated with `TKE_QCSRole` by default. After TKE is activated and `TKE_QCSRole` is granted, you have the permissions of various Ops-related features, including log features.

Authorization steps

This policy and the preset policy QcloudAccessForTKERole are authorized at the same time, so no extra operation is needed.

Permission content

Log service

Permission Name	Permission Description
`cls:listTopic`	Displaying the list of log topics under a specified logset
`cls:getTopic`	Viewing log topic information
`cls:createTopic`	Creating a log topic
`cls:modifyTopic`	Modifying a log topic
`cls:deleteTopic`	Deleting a log topic
`cls:listLogset`	Displaying the logset list
`cls:getLogset`	Viewing logset information
`cls:createLogset`	Creating a logset
`cls:modifyLogset`	Modifying a logset
`cls:deleteLogset`	Deleting a logset
`cls:listMachineGroup`	Displaying the server group list

Permission Name	Permission Description
`cls:getMachineGroup`	Viewing server group information
`cls:createMachineGroup`	Creating a server group
`cls:modifyMachineGroup`	Modifying a server group
`cls:deleteMachineGroup`	Deleting a server group
`cls:getMachineStatus`	Viewing server group status
`cls:pushLog`	Uploading logs
`cls:searchLog`	Querying logs
`cls:downloadLog`	Downloading logs
`cls:getCursor`	Getting the cursor based on time
`cls:getIndex`	Viewing indexes
`cls:modifyIndex`	Modifying indexes
`cls:agentHeartBeat`	Heartbeat
`cls:getConfig`	Getting the pusher configuration information

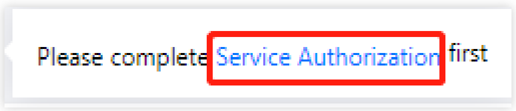
Preset policy QcloudAccessForTKERoleInCreatingCFSStorageclass

Authorization scenario

The Tencent Cloud CFS add-on can help you use file storage in TKE clusters. When using this add-on for the first time, you need to authorize relevant resources, such as file systems in CFS, via TKE.

Authorization steps

1. Log in to the TKE console and click **Cluster** in the left sidebar.
2. On the "Cluster management" page, select the region and ID of the target cluster to go to the cluster details page.
3. Select **Add-on management** and click **Create**.
4. On the **Add-on management** page, if the add-on is selected as "CFS" for the first time, click **Service Authorization** at the bottom of the page.


Please complete Service Authorization first

5. In the "Service authorization" window that pops up, click **Cloud Access Management**.
6. On the "Role management" page, click **Grant** to complete authentication.

Permission content

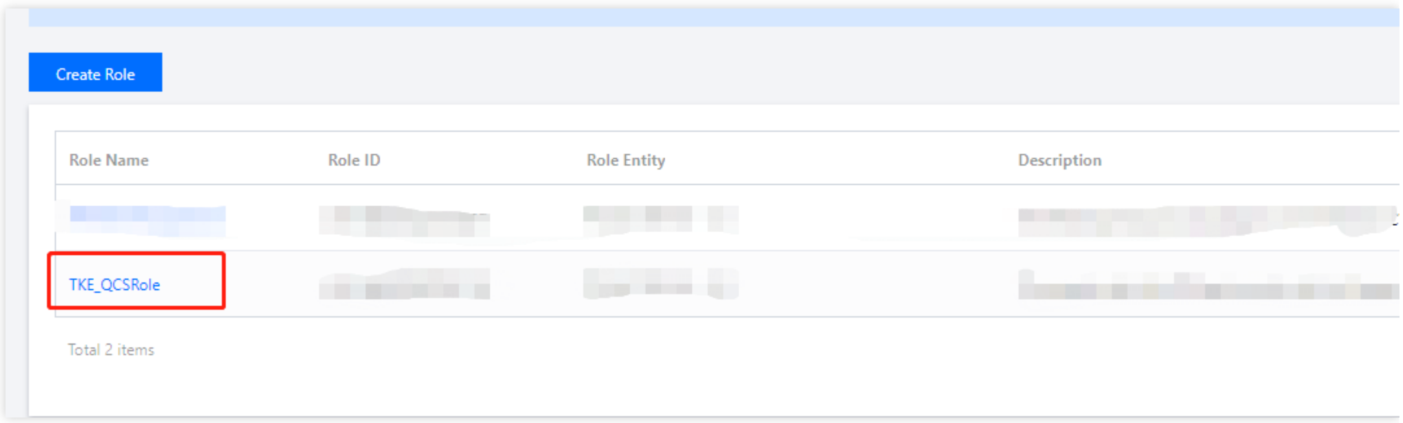
File storage

Permission Name	Permission Description
cfs:CreateCfsFileSystem	Creating a file system
cfs:DescribeCfsFileSystems	Querying a file system
cfs:DescribeMountTargets	Querying mount targets of a file system
cfs:DeleteCfsFileSystem	Deletes a file system

Preset policy QcloudCVMFinanceAccess

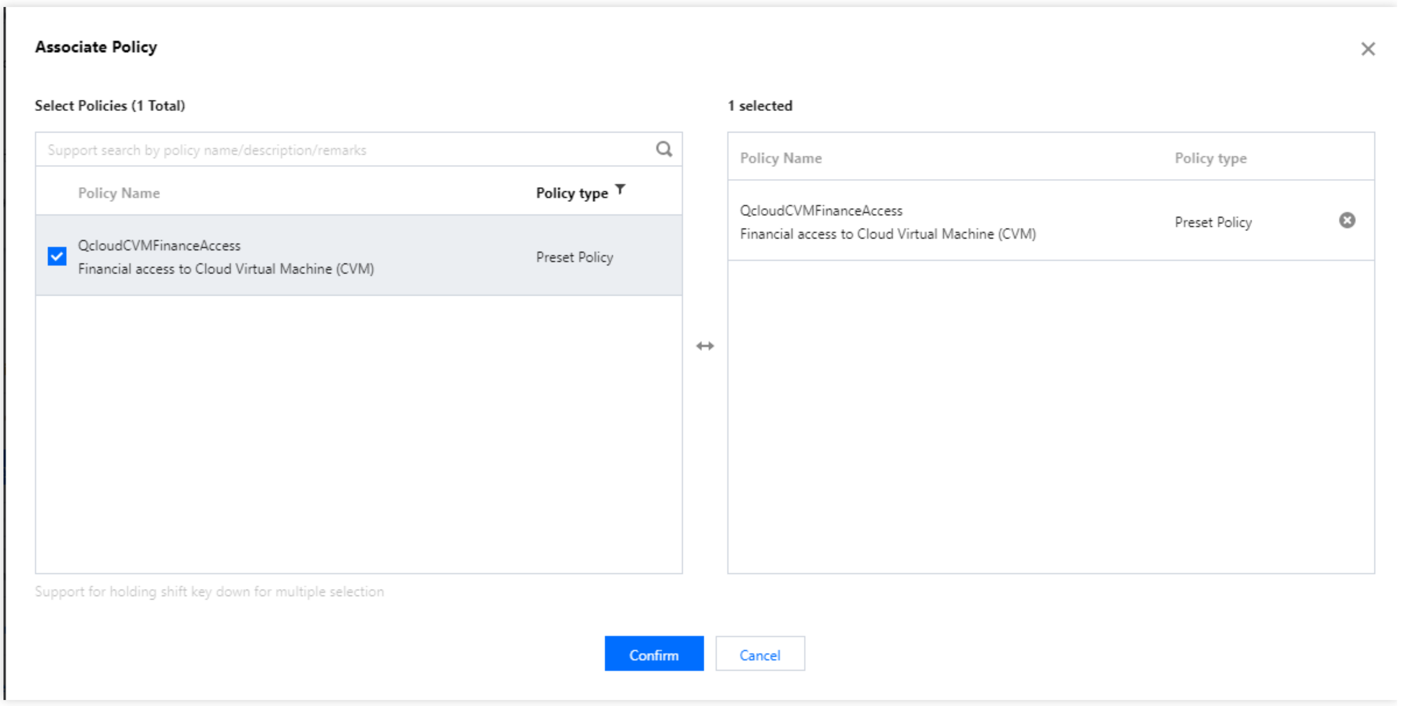
Authorization steps

1. Log in to the CAM console, and select **Roles** in the left sidebar.
2. On the role list page, click **TKE_QCSRole** to enter the role management page.



3. Select **Associate policy** on the **TKE_QCSRole** page, and confirm the operation in the "Risk tips" pop-up window.

4. In the "Associate policy" window that pops up, find the policy `QcloudCVMFinanceAccess` and select it.



Associate Policy

Select Policies (1 Total)

Support search by policy name/description/remarks

	Policy Name	Policy type
✓	QcloudCVMFinanceAccess Financial access to Cloud Virtual Machine (CVM)	Preset Policy

1 selected

Policy Name	Policy type
QcloudCVMFinanceAccess Financial access to Cloud Virtual Machine (CVM)	Preset Policy

Support for holding shift key down for multiple selection

5. Click **Confirm** to complete the process.

Permission content

Permission Name	Permission Description
`finance:*`	CVM finance permission

IPAMDofTKE_QCSRole

`IPAMDofTKE_QCSRole` is the TKE IPAMD support service role. After the permissions of this role are granted, you need to associate preset policies in the authorization scenarios described in this document. After these operations are completed, the following policies will appear in the list of authorized policies of the role:

`QcloudAccessForIPAMDofTKERole` : The permission for TKE IPAMD to access cloud resources

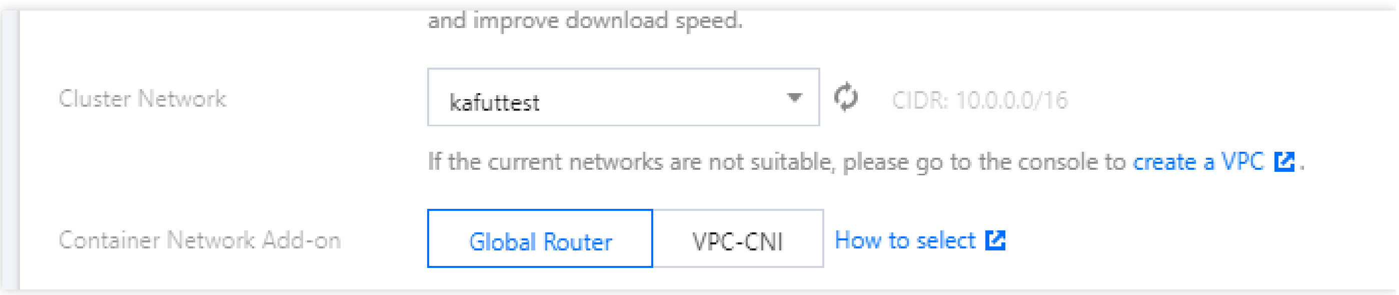
Preset policy QcloudAccessForIPAMDofTKERole

Authorization scenario

When using the VPC-CNI network mode to create a cluster for the first time, you need to grant permission for TKE IPAMD to access cloud resources, so that you can use the VPC-CNI network mode normally.

Authorization steps

1. Log in to the TKE console and click **Cluster** in the left sidebar.
2. On the "Cluster Management" page, click **Create** or **Create with a template** above the cluster list.
3. On the "Create cluster" page, select **VPC-CNI** for **Container network add-on** in "Cluster information" section, and click "Service Authorization".


and improve download speed.

Cluster Network: kafuttest — CIDR: 10.0.0.0/16

If the current networks are not suitable, please go to the console to create a VPC.

Container Network Add-on: Global Router | VPC-CNI — How to select

4. In the displayed "Service authorization" window, click **Go to Cloud Access Management**.
5. On the "Role management" page, click **Grant** to complete authentication.

Permission content

- CVM

Permission Name	Permission Description
`cvm:DescribeInstances`	Viewing the list of instances

- Tag

Permission Name	Permission Description
`tag:GetResourcesByTags`	Querying the resource list by tag
`tag:ModifyResourceTags`	Batch modifying tags associated with a resource
`tag:GetResourceTagsByResourceIds`	Querying tags associated with a resource

- VPC

Permission Name	Permission Description
`vpc:DescribeSubnet`	Querying the list of subnets
`vpc:CreateNetworkInterface`	Creating an ENI
`vpc:DescribeNetworkInterfaces`	Querying the list of ENIs
`vpc:AttachNetworkInterfac` e	Binding an ENI with a CVM

Permission Name	Permission Description
`vpc:DetachNetworkInterface`	Unbinding an ENI from a CVM
`vpc:DeleteNetworkInterface`	Deleting an ENI
`vpc:AssignPrivateIpAddresses`	Applying for private IP addresses for an ENI
`vpc:UnassignPrivateIpAddresses`	Returning the private IP addresses of an ENI
`vpc:MigratePrivateIpAddress`	Migrating the private IP addresses of an ENI
`vpc:DescribeSubnetEx`	Querying the list of subnets
`vpc:DescribeVpcEx`	Querying peering connection
`vpc:DescribeNetworkInterfaceLimit`	Querying the ENI quota
`vpc:DescribeVpcPrivateIpAddresses`	Querying the private IP address of a VPC

Controlling TKE cluster-level permissions

Using TKE Preset Policy Authorization

Last updated：2020-09-04 10:53:28

This document describes the preset policies offered by Tencent Kubernetes Engine (TKE). It shows you how to associate sub-accounts with preset policies to grant specific permissions. You can use this document as a reference to configure preset policies that are in line with your particular business needs.

TKE Preset Policies

You can grant relevant permissions to sub-accounts by using the following preset policies:

Policy	Description
`QcloudTKEFullAccess`	This policy grants full read/write access permissions for TKE, including permissions for TKE and related CVMs, CLBs, VPCs, monitors, and user groups.
`QcloudTKEInnerFullAccess`	This policy grants full access permission for TKE. However, since TKE involves the use of many products, we recommend configuring `QcloudTKEFullAccess` .
`QcloudTKEReadOnlyAccess`	This policy grants read-only permission for TKE.

The following preset policies are used to grant permissions for specific TKE services. We do not recommend associating the following preset policies with a sub-account:

Policy	Description
`QcloudAccessForCODINGRoleInAccessTKE`	This policy grants the relevant TKE permissions for the Coding service.
`QcloudAccessForIPAMDofTKERole`	This policy grants the relevant ENI permissions for the TKE service.
`QcloudAccessForIPAMDRoleInQcloudAllocateEIP`	This policy grants the relevant EIP permissions for the TKE service.
`QcloudAccessForTKERole`	This policy grants the relevant CVM, Tag, CLB, and CLS permissions for the TKE service.

`QcloudAccessForTKERoleInCreatingCFSStorageclass`	This policy grants the relevant CFS permissions for the TKE service.
`QcloudAccessForTKERoleInOpsManagement`	This policy associates the TKE service role (TKE_QCSRole) so that TKE can access other Tencent Cloud services, including CLS.

Associating Sub-accounts with Preset Policies

When setting user permissions during the creation of a sub-account, you can associate preset policies with the sub-account by direct association or association via group.

Direct association

By directly associating your sub-account with a policy, the sub-account obtains the permissions contained in the policy. The direct association process is outlined below:

1. Log in to the CAM console and select **Users** -> **User List** on the left sidebar.
2. On the **User List** page, find the target sub-account and click **Grant Permission** in the **Operation** column.
3. On the **Associate Policies** page, select the policies that you want to associate.
4. Click **OK**.

Association via group

By adding your sub-account to a user group, the sub-account automatically obtains the permissions that are associated with the user group. To disassociate the sub-account from the policies of the group, you simply need to remove the sub-account from the user group. The group association process is outlined below:

1. Log in to the CAM console and select **Users** -> **User List** on the left sidebar.
2. On the **User List** page, find the target sub-account and choose **More** -> **Add to Group** in the **Operation** column.
3. On the **Add to Group** page, select the target user group.
4. Click **OK**.

Logging in to the sub-account for verification

Log in to the TKE console to verify that the features corresponding to the associated policies can be used and/or accessed properly. If so, this indicates that the sub-account was successfully authorized.

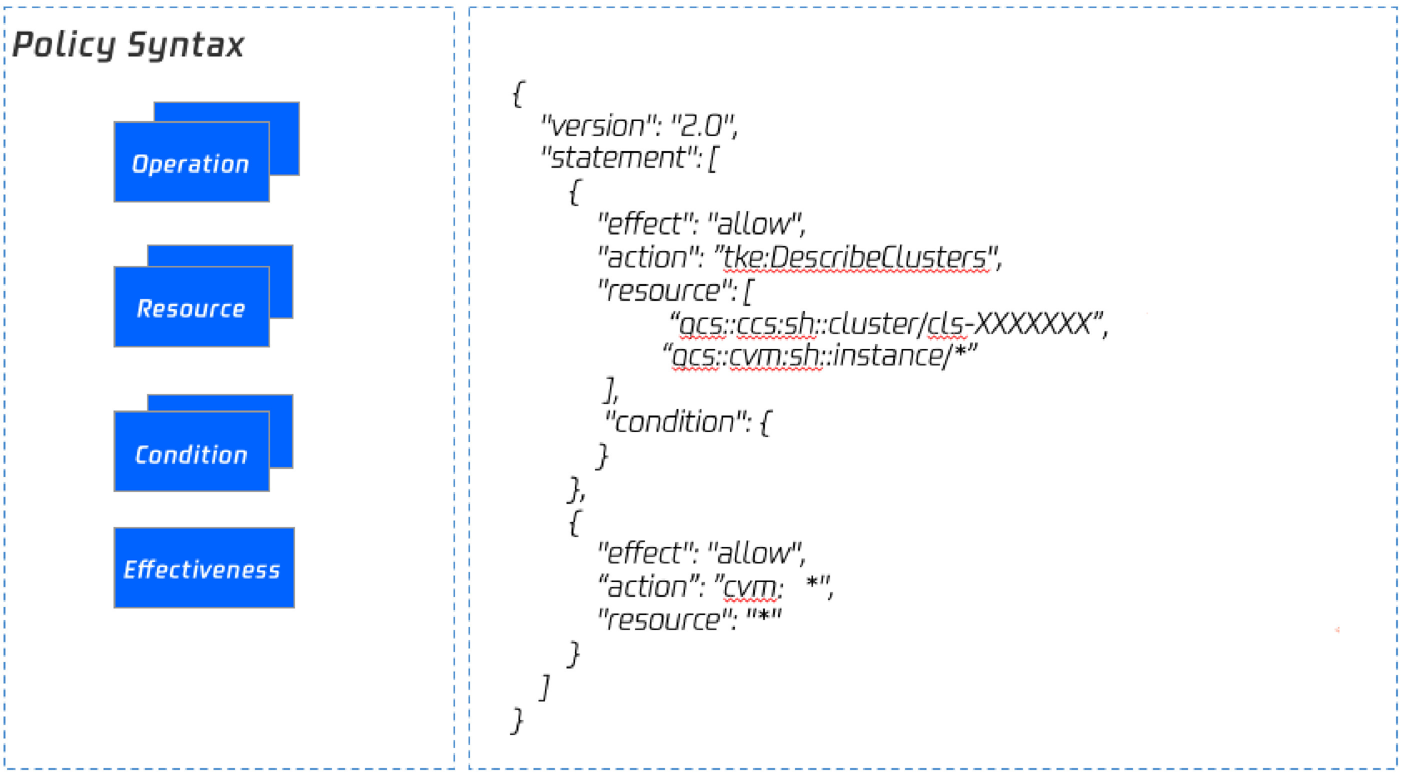
Authorizing by using custom policies

Last updated：2020-10-10 15:04:42

This document describes how to configure custom policies in Tencent Kubernetes Engine (TKE) and grant sub-accounts specific permissions. Reference this document to create custom policies that best fit your business requirements.

Policy Syntax Description

The following figure shows the structure of the policy syntax.



```
{
  "version": "2.0",
  "statement": [
    {
      "effect": "allow",
      "action": "tke:DescribeClusters",
      "resource": [
            "qcs::ccs:sh::cluster/cls-XXXXXXX",
            "qcs::cvm:sh::instance/*"
      ],
       "condition": {
      }
    },
    {
      "effect": "allow",
      "action": "cvm:  *",
      "resource": "*"
    }
  ]
}
```

- **action**: indicates an API.
- **resource**: indicates a resource.

Note：

You can define the policy syntax on your own, or create a custom policy by using the policy generator in CAM. You can configure a custom policy based on the following example.

- Configuring a Sub-account's Administrative Permissions for a Single TKE Cluster

- Use tags to grant full permissions for a batch of clusters to a sub-account

Configuring TKE API Permissions

This section describes multiple features, their sub-features, corresponding Tencent Cloud APIs, APIs for indirect calls, resource levels for permission control, and Action fields of clusters and node modules.

Cluster modules

The following table describes the mappings between features and APIs.

Feature	Sub-feature	Corresponding Tencent Cloud API	API for Indirect Calls
Creating an empty cluster	• Selecting a Kubernetes version • Selecting a runtime component • Selecting a VPC • Setting a container network • Selecting a custom image • Setting IPVS	tke:CreateCluster	cam:GetRole account:DescribeUserData account:DescribeWhiteList tag:GetTagKeys cvm:GetVmConfigQuota vpc:DescribeVpcEx cvm:DescribeImages
Using an existing CVM to create a managed cluster	• Creating an empty cluster to include features • Using an existing CVM as a node • Mounting a security		cvm:DescribeInstances vpc:DescribeSubnetEx cvm:DescribeSecurityGroups vpc:DescribeVpcEx cvm:DescribeImages cvm:ResetInstance cvm:DescribeKeyPairs

	group • Mounting a data disk • Enabling automatic adjustment		
Using an existing CVM to create a self-deployed cluster	• Creating an empty cluster to include features • Using an existing CVM as a node • Using an existing CVM as Control Plane & ETCD • Mounting a security group • Mounting a data disk • Enabling automatic adjustment		cvm:DescribeInstances vpc:DescribeSubnetEx cvm:DescribeSecurityGroups vpc:DescribeVpcEx cvm:DescribeImages cvm:ResetInstance cvm:DescribeKeyPairs
Automatically creating a CVM to create a managed cluster	• Creating an empty cluster to include features • Purchasing a CVM as a node • Mounting a security group • Mounting a data disk • Enabling automatic		cvm:DescribeSecurityGroups cvm:DescribeKeyPairs cvm:RunInstances vpc:DescribeSubnetEx vpc:DescribeVpcEx cvm:DescribeImages

	adjustment		
Automatically creating a CVM to create a self-deployed cluster	• Creating an empty cluster to include features • Purchasing a CVM as a node • Purchasing a CVM as Control Plane & ETCD • Mounting a security group • Mounting a data disk • Enabling automatic adjustment		cvm:DescribeSecurityGroups cvm:DescribeKeyPairs cvm:RunInstances vpc:DescribeSubnetEx vpc:DescribeVpcEx cvm:DescribeImages
Querying a cluster list	-	tke:DescribeClusters	-
Displaying cluster credentials	-	tke:DescribeClusterSecurity	-
Enabling/Disabling the private network/Internet access URL of a cluster	• Creating an Internet access port for a managed cluster • Creating a cluster access port	tke:CreateClusterEndpointVip tke:CreateClusterEndpoint tke:ModifyClusterEndpointSP tke:DescribeClusterEndpointVipStatus tke:DescribeClusterEndpointStatus tke:DeleteClusterEndpointVip tke:DeleteClusterEndpoint	-

	• Modifying security policies for the Internet access port of a managed cluster • Querying the Internet access port enabling status of a managed cluster • Deleting the Internet access port of a managed cluster • Deleting a cluster access port		
Deleting a cluster	-	tke:DeleteCluster	tke:DescribeClusterInstances tke:DescribeInstancesVersion tke:DescribeClusterStatus

Node modules

The following table describes the mappings between features and APIs.

Feature	Sub-feature	Corresponding Tencent API	API for Indirect Calls	Resource Level for Permission Control
Adding an existing node	• Adding an existing node to a cluster	tke:AddExistedInstances	cvm:DescribeInstances vpc:DescribeSubnetEx cvm:DescribeSecurityGroups vpc:DescribeVpcEx cvm:DescribeImages cvm:ResetInstance	• Cluster-level permissions are required for adding

	• Resetting a data disk • Setting a security group		cvm:DescribeKeyPairs cvm:ModifyInstancesAttribute tke:DescribeClusters	an existing node. • CVM-level permissions are required for obtaining a CVM list.
Creating a node	• Creating a node and adding it to a cluster • Resetting a data disk • Setting a security group	tke:CreateClusterInstances	cvm:DescribeSecurityGroups cvm:DescribeKeyPairs cvm:RunInstances vpc:DescribeSubnetEx vpc:DescribeVpcEx cvm:DescribeImages tke:DescribeClusters	Cluster-level permissions are required for creating a node.
Node list	Viewing a cluster node list	tke:DescribeClusterInstances	cvm:DescribeInstances tke:DescribeClusters	• Cluster-level permissions are required for viewing a node list. • CVM-level permissions are required for obtaining a CVM list.
Deleting a node	-	tke:DeleteClusterInstances	cvm:TerminateInstances tke:DescribeClusters	• Cluster-level permissions are required for viewing a node list. • CVM-level permissions are required for

				obtaining a CVM list. • The termination policy of a node is required for terminating the node.

Usage Examples

Using Labels to Configure Sub-accounts with Full Read/Write Permissions for Batch Clusters

Last updated：2023-02-02 17:05:22

Overview

You can grant a user permissions to view and use specific resources in the TKE console by using a Cloud Access Management (CAM) policy. This document describes how to grant a sub-account the permissions for a cluster with the specified tag in the console.

Directions

1. Log in to the CAM console. Click **Create custom policy** in the upper-left corner.
2. On the **Select Policy Creation Method** page that pops up, select **Authorize by Tag**.
3. In the service and action addition area of the Visual Policy Generator, enter the following information, and edit an authorization statement.
    - **Service** (required): select TKE.
    - **Action** (required): select the actions you want to authorize.
4. In the tag selection area, select the tags to authorize. Authorized sub-accounts will have the full read/write permissions for the resources with the specified tag key and tag value.
5. Click **Next**. On the **Bind User**/**Group**/**Role** page displayed, enter the policy name and description. The policy name is `policygen` by default, which is generated automatically in the console. The suffix number is generated based on the creation date. This is customizable.
6. Authorize users/groups/roles. Authorized sub-accounts will have the full read/write permissions for the resources with the specified tag key and tag value.

- **Authorized User**: select the target sub-accounts as required.
- **Authorized User Group**: select the user group to which the target sub-accounts belongs.
- **Authorized Role**: select the role to which the target sub-accounts belong.

7. Click **Complete**.

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

Last updated：2021-06-08 11:21:00

Overview

You can grant a user the permissions to view and use specific resources in the TKE console by using a CAM policy. This document describes how to configure the CAM policy of a single cluster in the console.

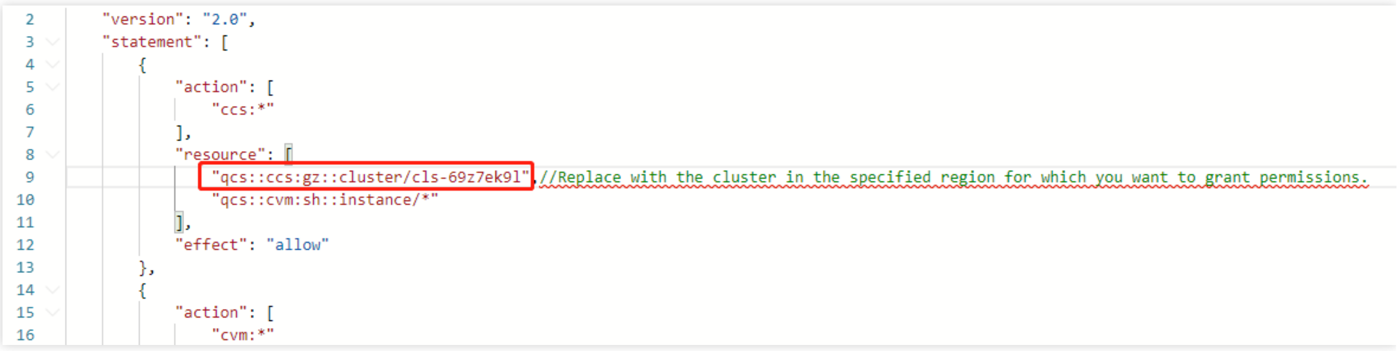
Directions

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 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 content.

```
json
{
"version": "2.0",
"statement": [
{
"action": [
"ccs:*"
],
"resource": [
"qcs::ccs:sh::cluster/cls-XXXXXXX",
"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", modify `qcs::ccs:sh::cluster/cls-XXXXXXX` to the cluster in the specified region for which you want to grant permissions, as shown below:
   For example, if you need to grant full read/write permission for the cls-69z7ek9l cluster in Guangzhou, modify `qcs::ccs:sh::cluster/cls-XXXXXXX` to `"qcs::ccs:gz::cluster/cls-69z7ek9l"` .



```
    "version": "2.0",
    "statement": [
        {
            "action": [
                "ccs:*"
            ],
            "resource": [
                "qcs::ccs:gz::cluster/cls-69z7ek9l",//Replace with the cluster in the specified region for which you want to grant permissions.
                "qcs::cvm:sh::instance/*"
            ],
            "effect": "allow"
        },
        {
            "action": [
                "cvm:*"
```

Note：

Replace with the ID of the cluster 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.

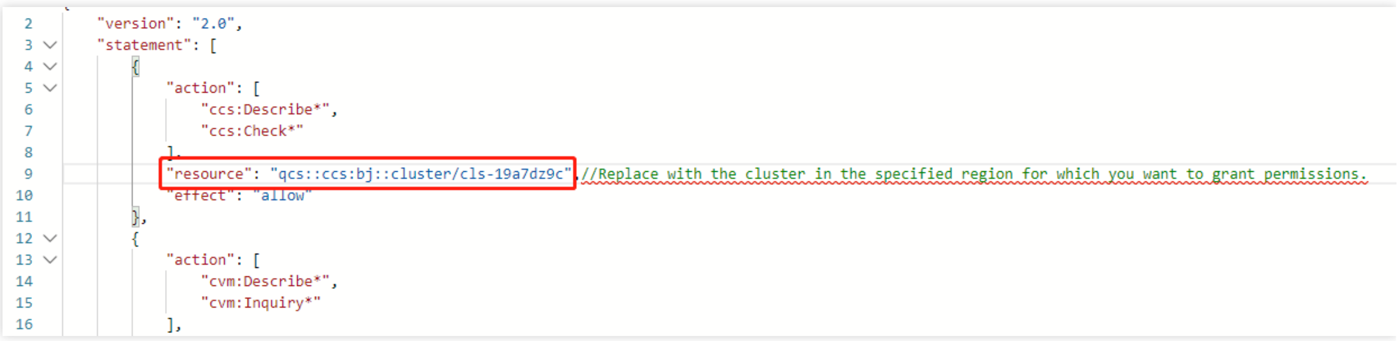
Configuring read-only permission for a single cluster

1. Log in to the CAM console.
2. In the left sidebar, click Policies to go to the policy management page.
3. Click **Create 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 content.

```
json
{
"version": "2.0",
"statement": [
{
"action": [
"ccs:Describe*",
"ccs:Check*"
],
"resource": "qcs::ccs:gz::cluster/cls-1xxxxxx",
"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", modify `qcs::ccs:gz::cluster/cls-1xxxxxx` to the cluster in the specified region for which you want to grant permissions, as shown below:
   For example, if you need to grant ready-only permission for the cls-19a7dz9c cluster in Beijing, modify `qcs::ccs:gz::cluster/cls-1xxxxxx` to `qcs::ccs:bj::cluster/cls-19a7dz9c` .



```
    "version": "2.0",
    "statement": [
        {
            "action": [
                "ccs:Describe*",
                "ccs:Check*"
            ],
            "resource": "qcs::ccs:bj::cluster/cls-19a7dz9c",//Replace with the cluster in the specified region for which you want to grant permissions.
            "effect": "allow"
        },
        {
            "action": [
                "cvm:Describe*",
                "cvm:Inquiry*"
            ],
```

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：2023-02-02 17:05:22

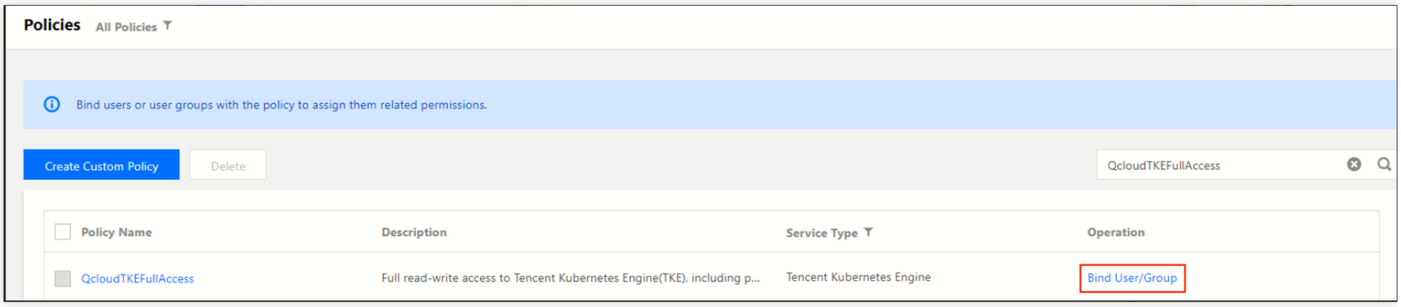
Overview

You can grant a user the permissions to view and use specific resources in the TKE console by using a CAM policy. This document describes how to configure certain permission policies in the console.

Directions

Configuring Full Read/write Permission

1. Log in to the CAM console and select **Policies** in the left sidebar.
2. On the **Policies** page, click **Bind User**/**Group**/**Role** in the **Operation** column of the **QcloudTKEFullAccess** policy.


Policies
All Policies
Bind users or user groups with the policy to assign them related permissions.
Create Custom Policy
Delete
QcloudTKEFullAccess

Policy Name	Description	Service Type	Operation
QcloudTKEFullAccess	Full read-write access to Tencent Kubernetes Engine(TKE), including p...	Tencent Kubernetes Engine	Bind User/Group

3. In the **Bind User**/**Group**/**Role** window that pops up, select the accounts that need 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.
4. On the **Policies** page, click **Bind User**/**Group**/**Role** in the **Operation** column of the **QcloudCCRFullAccess** policy.
5. In the **Bind User**/**Group**/**Role** window that pops up, select the accounts that need full read/write permission for Image Registry, and click **OK** to grant full read/write permission for Image Registry to the sub-accounts.

> Note：
> 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 and select **Policies** in the left sidebar.
2. On the **Policies** page, click **Bind User**/**Group**/**Role** in the **Operation** column of the **QcloudTKEReadOnlyAccess** policy.
3. In the **Bind User**/**Group**/**Role** window that pops up, select the accounts that need the read-only permission for the TKE service, and click **OK** to grant the read-only permission for the TKE service to the sub-accounts.
4. On the **Policies** page, click **Bind User**/**Group**/**Role** in the **Operation** column of the **QcloudCCRReadOnlyAccess* policy.
5. In the **Bind User**/**Group**/**Role** window that pops up, select the accounts that need the read-only permission for Image Registry, and click **OK** to grant the read-only permission for Image Registry to the sub-accounts.

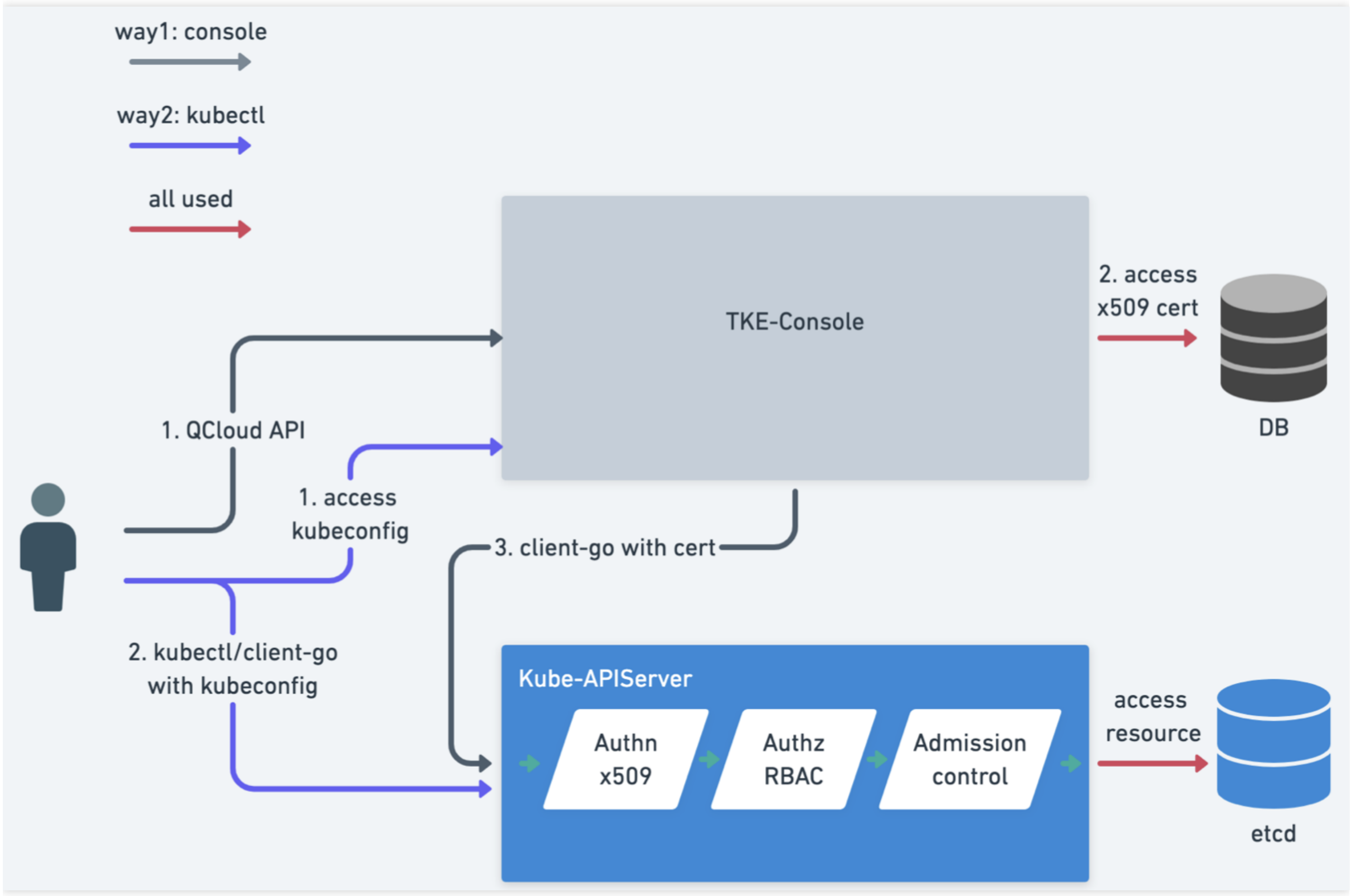
> Note：
> If you want to use the trigger and automatic building features of Image Registry, you also need to configure additional permissions for TKE - continuous integration (CCB).

TKE Kubernetes Object-level Permission Control

Overview

Last updated：2020-12-24 10:56:17

TKE supports the Kubernetes RBAC authorization method, allowing you to perform fine-grained access control for sub-accounts. With this authorization method, you can access resources in a cluster through the TKE console and kubectl. For more information, see the following figure.



Glossary

RBAC

Role-Based Access Control (RBAC) associates users and permissions with roles to indirectly grant permissions to users.

In Kubernetes, RBAC is implemented through the `rbac.authorization.k8s.io` API group, which allows cluster administrators to dynamically configure policies through Kubernetes APIs.

Role

A Role is used to define access permissions for resources in a single namespace.

ClusterRole

A ClusterRole is used to define access permissions for resources in an entire cluster.

RoleBinding

RoleBinding grants the permissions defined in a role to a user or group of users in order to grant authorization for a namespace.

ClusterRoleBinding

ClusterRoleBinding grants the permissions defined in a role to a user or group of users in order to grant cluster-wide authorization.

For more information, see the Kubernetes official documentation.

Solutions for TKE Kubernetes Object-Level Permission Control

Verification method

Kubernetes API servers support various verification policies, such as x509 certificates, bearer tokens, and basic auth. Of these, only individual bearer token verification policies support verification based on the tokens of specified known-token csv files, such as bearer tokens, serviceaccount tokens, OIDC tokens, and webhook token servers.

With implementation complexity and different usage scenarios in mind, TKE has chosen to use x509 certificates as the verification method. This verification method offers the following advantages:

- This method is easier for users to understand.
- No complex changes need to be made to existing clusters.
- You can sort by users and groups, which facilitates subsequent scaling.

TKE implements the following features based on x509 certificate verification:

- Each sub-account has a unique client certificate used for accessing Kubernetes API servers.
- When a sub-account accesses Kubernetes resources on the console, the backend uses the sub-account’s client certificate to access the Kubernetes API server by default.
- A sub-account can update its unique client certificate to prevent credential disclosure.

- A root account or an account that has `tke:admin` permission for a cluster can view and update the certificates of other sub-accounts.

Authorization method

Kubernetes supports two main authorization methods: RBAC and Webhook Server. In order to provide a consistent experience for users who are familiar with and work with native Kubernetes, TKE has chosen RBAC as its authorization method. This authorization method provides preset Roles and ClusterRoles. You only need to create the corresponding RoleBinding or ClusterRoleBinding to implement authorizations for a cluster or namespace. This authorization method has the following advantages:

- It is friendly to users who have a basic knowledge of Kubernetes.
- It allows you to reuse Kubernetes RBAC capabilities and supports various verb-based permission controls for namespaces, API groups, and resources.
- It supports custom policies.
- It allows you to manage custom extended API resources.

Features of TKE Kubernetes Object-level Permission Control

By using the authorization management feature provided by TKE, you can perform more fine-grained permission control. For example, you can configuring sets of permissions such as assigning read-only permissions to a sub-account or assigning read/write permissions to only a certain namespace under a sub-account. For more information on configuring more fine-grained sets of permissions for sub-accounts, see the following documents:

- Using a preset identity for authorization
- Using custom policies for authorization

Comparison of Authorization Modes

Last updated：2020-09-18 10:44:42

Tencent Kubernetes Engine (TKE) currently supports both new and old authorization methods. However, old authorization methods cannot be used to perform Kubernetes-level authorization. We recommend you upgrade the authorization method for your cluster so that you can perform fine-grained permission control for the Kubernetes resources in the cluster.

Comparison of New and Old Authorization Methods

Item	Old Authorization Method	New Authorization Method
Kubeconfig	admin token	x509 certificate unique to each sub-account
Access to cluster resources on the console	No fine-grained permissions, and sub-accounts are granted full read/write permission	Incorporates Kubernetes RBAC resource control

Upgrading the Authorization Method for Existing Clusters

Upgrading the authorization method

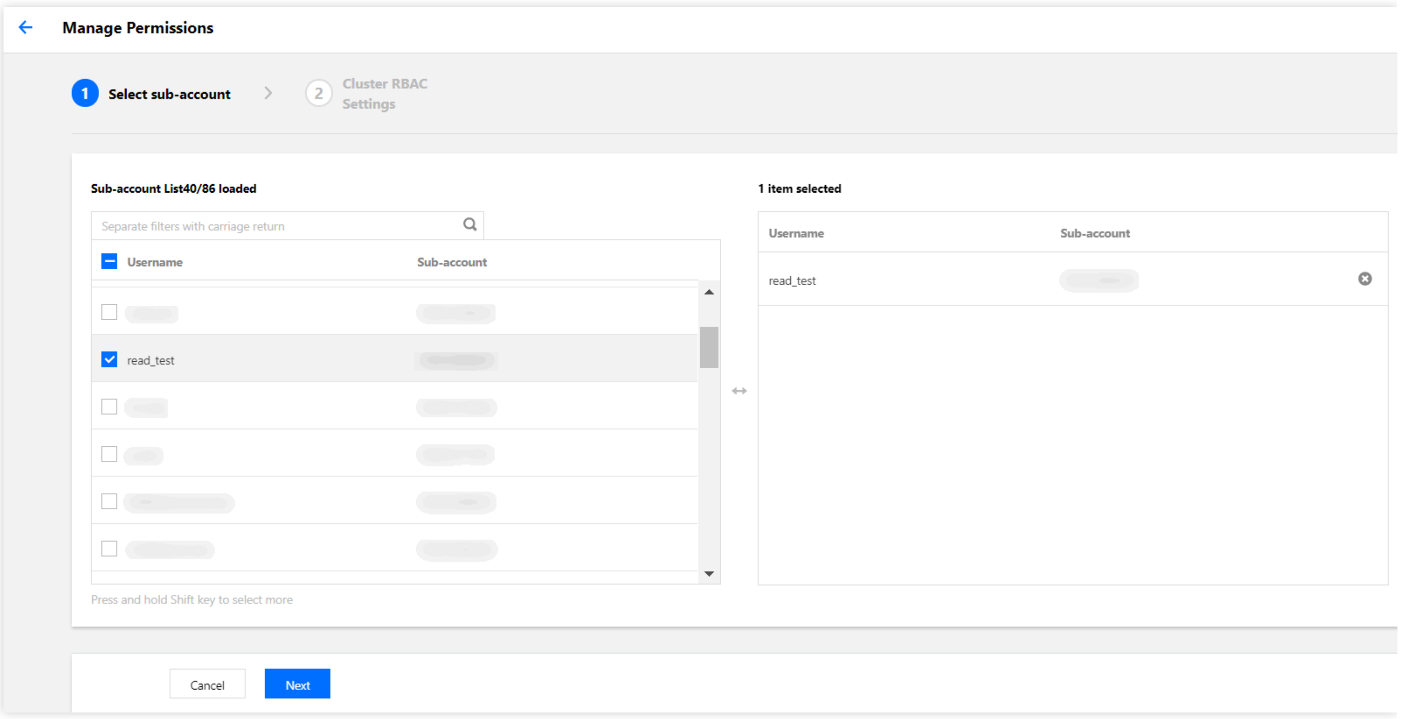
To upgrade a cluster that uses an old authorization method, perform the following steps:

1. Log in to the TKE console and click **Cluster** on the left sidebar.
2. On the **Cluster Management** page, click the ID of the target cluster.
3. On the cluster details page, select **Authorization Management** -> **ClusterRole** on the left sidebar.
4. On the **ClusterRole** page, click **RBAC Policy Generator**.
5. On the **Switch Permission Management Mode** page, click **Switch Permission Management Mode** to upgrade the authorization method.
   To ensure that the new authorization method is compatible with the old one, TKE will perform the following operations during the upgrade:
   vi. Creating the default preset administrator ClusterRole: `tke:admin` .
   vii. Obtaining the sub-account list.
   viii. Generating the x509 client certificates for Kubernetes API server authentication for each sub-account.
   ix. Binding the `tke:admin` role to each sub-account to ensure compatibility with existing features.
   x. Completing the upgrade.

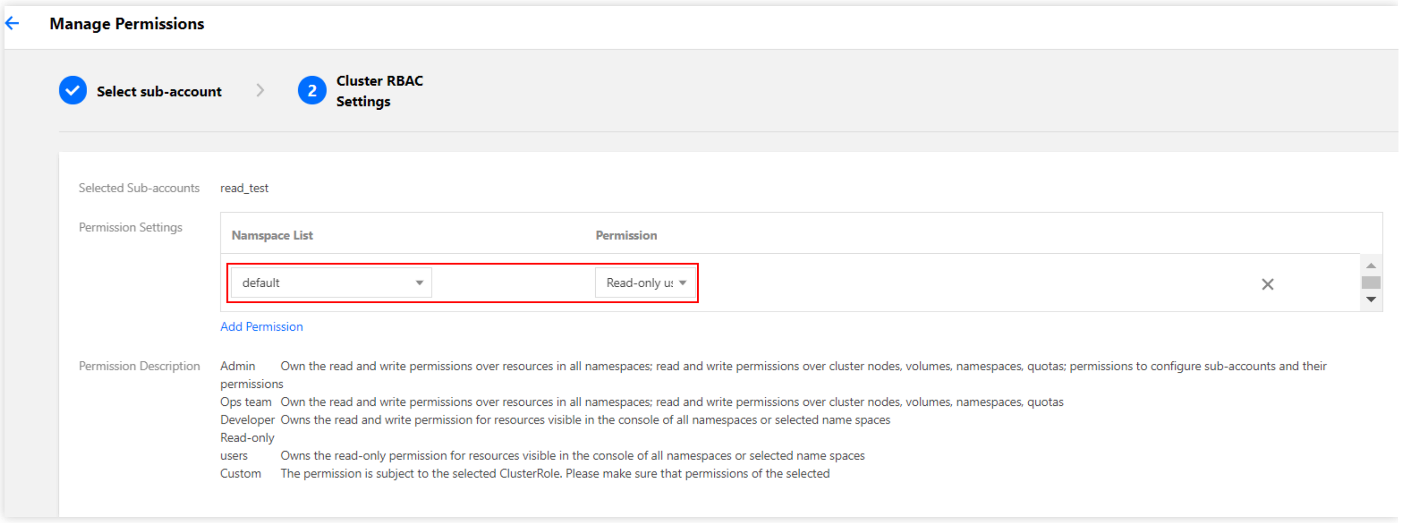
Repossessing sub-account permissions

After the authorization method of a cluster is upgraded, the cluster administrator (often the root account administrator or OPS person who created the cluster) can repossess the cluster permissions granted to sub-accounts as required. The steps are as follows:

1. Select an item under the cluster’s **Authorization Management** page, and click **RBAC Policy Generator** on the corresponding management page.
2. When you select a sub-account on the **Administration Permissions** page, select the sub-account whose permissions you want to repossess and then click **Next**, as shown in the following figure.



3. When you set the cluster RBAC, you can also set permissions. For example, select **Read-only Users** as the **Permission Setting** for the **default** namespace, as shown in the following figure.



Selected Sub-accounts read_test

Permission Settings

Namspace List	Permission
default	Read-only u:

Permission Description

Admin	Own the read and write permissions over resources in all namespaces; read and write permissions over cluster nodes, volumes, namespaces, quotas; permissions to configure sub-accounts and their permissions
Ops team	Own the read and write permissions over resources in all namespaces; read and write permissions over cluster nodes, volumes, namespaces, quotas
Developer	Owns the read and write permission for resources visible in the console of all namespaces or selected name spaces
Read-only users	Owns the read-only permission for resources visible in the console of all namespaces or selected name spaces
Custom	The permission is subject to the selected ClusterRole. Please make sure that permissions of the selected

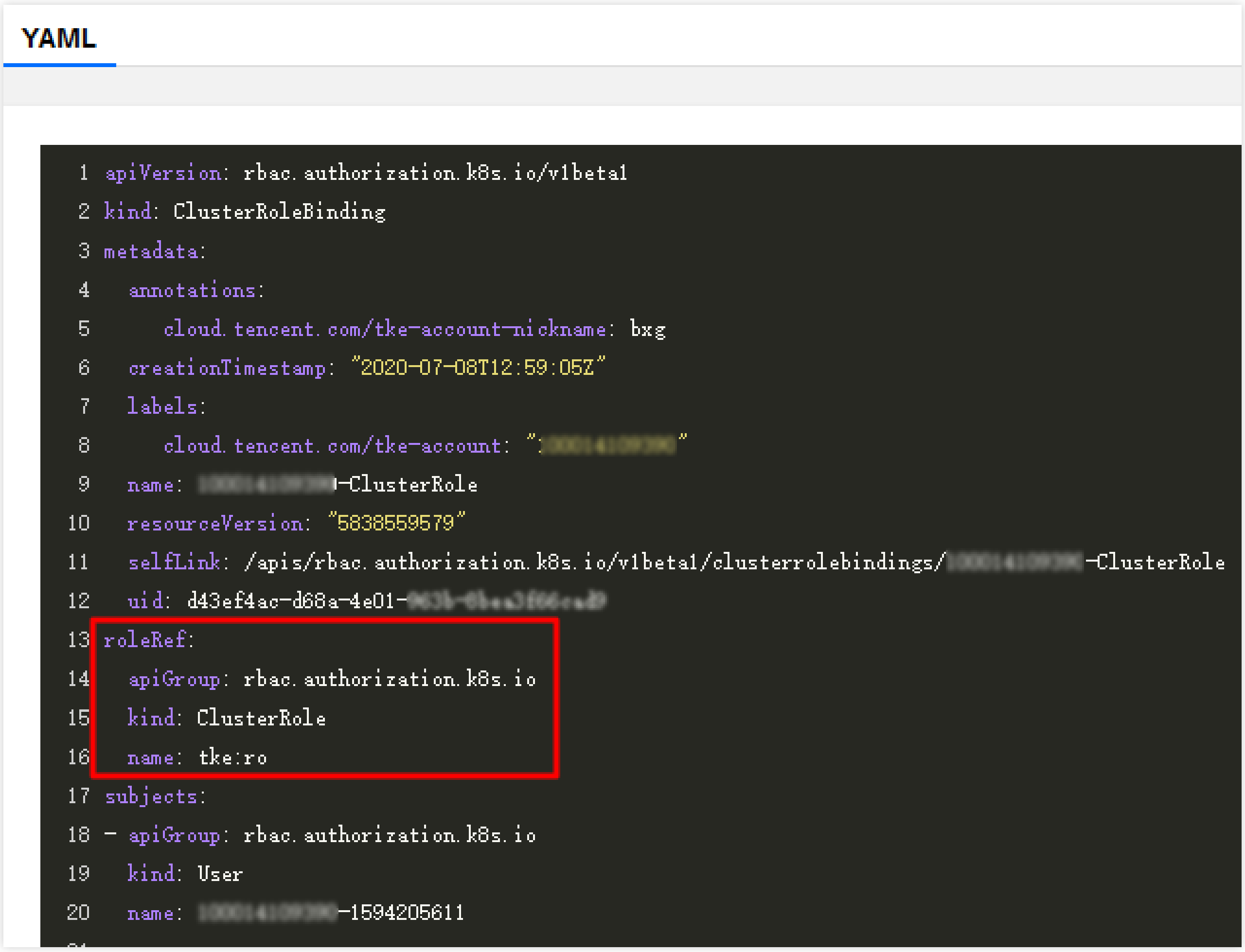
4. Click **Done** to complete the repossession.

Verifying permissions of sub-accounts

After you repossess the permissions of a sub-account, you can verify the current permissions as follows:

1. On the cluster details page, select **Authorization Management** -> **ClusterRoleBinding** on the left sidebar to enter the **ClusterRoleBinding** page.
2. Select the sub-account whose permissions have been repossessed to go to the YAML file page.
   The sub-account has `tke:admin` permission by default. After permissions are repossessed, you can view the

change in the YAML file, as shown in the following figure.



```yaml
apiVersion: rbac.authorization.k8s.io/v1beta1
kind: ClusterRoleBinding
metadata:
  annotations:
     cloud.tencent.com/tke-account-nickname: bxg
  creationTimestamp: "2020-07-08T12:59:05Z"
  labels:
     cloud.tencent.com/tke-account: "[redacted]"
  name: [redacted]-ClusterRole
  resourceVersion: "5838559579"
  selfLink: /apis/rbac.authorization.k8s.io/v1beta1/clusterrolebindings/[redacted]-ClusterRole
  uid: d43ef4ac-d68a-4e01-[redacted]
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: tke:ro
subjects:
- apiGroup: rbac.authorization.k8s.io
  kind: User
  name: [redacted]-1594205611
```

New Authorization Method FAQ

For a cluster that is created using the new authorization method, who has admin permission?

The cluster creator and the root account always have `tke:admin` ClusterRole permission.

Can I control the permissions of the current account?

TKE currently does not allow you to perform permission operations on the current account. You can perform these operations by using kubectl.

Can I directly perform operations on ClusterRoleBindings and ClusterRoles?

Please do not directly modify or delete ClusterRoleBindings and ClusterRoles.

How can I create a client certificate?

When you access cluster resources on the console through a sub-account, TKE will obtain the client certificate of the sub-account. If no certificate is obtained, TKE will create a client certificate for the sub-account.

After a sub-account is deleted on the CAM console, will TKE automatically repossess the relevant permissions?

TKE will automatically repossess the permissions, so you do not need to perform any additional operations.

How can I grant “authorization management” permission to another account?

You can use the default admin role `tke:admin` to grant “authorization management” permission.

Using Preset Identity Authorization

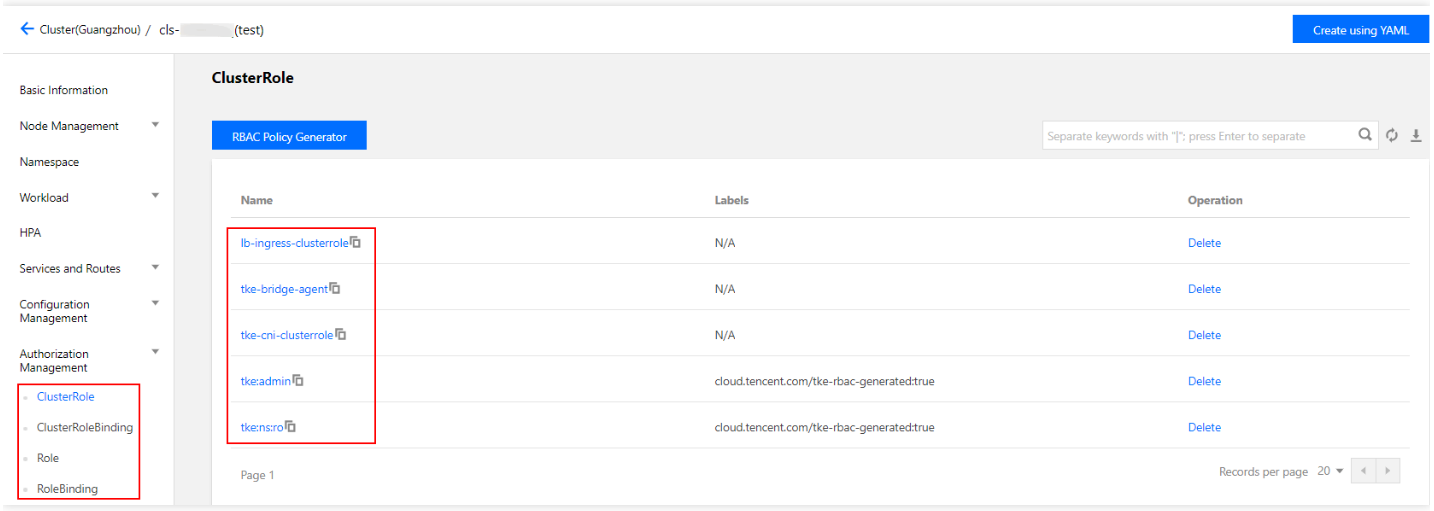
Last updated：2022-12-06 11:23:25

Description of Preset Roles

The Tencent Kubernetes Engine (TKE) console provides fine-grained permission control for Kubernetes resources based on Kubernetes' native Role-Based Access Control (RBAC) authorization policies. It also provides the preset roles `Role` and `ClusterRole` , which are described below:

Role

The TKE console provides an access management page for which the **root account** and **cluster creator** by default have administrator permissions and can manage sub-accounts that have the DescribeCluster Action permission for a given cluster. See the following figure for more information.



ClusterRole

Name	Labels	Operation
lb-ingress-clusterrole	N/A	Delete
tke-bridge-agent	N/A	Delete
tke-cni-clusterrole	N/A	Delete
tke:admin	cloud.tencent.com/tke-rbac-generated:true	Delete
tke:ns:ro	cloud.tencent.com/tke-rbac-generated:true	Delete

Page 1 Records per page 20

ClusterRole

- **For all namespaces**:
- **Administrators (tke:admin)**: have read/write permission for the resources in all namespaces, read/write permission for cluster nodes, storage volumes, namespaces, and quotas, and read/write permission for sub-account configurations.
- **OPS personnel (tke:ops)**: have read/write permission for the resources visible on the console in all namespaces and read/write permission for cluster nodes, storage volumes, namespaces, and quotas.
- **Developers (tke:dev)**: have read/write permission for the resources visible on the console in all namespaces.
- **Restricted personnel (tke:ro)**: have read-only permission for the resources visible on the console in all namespaces.

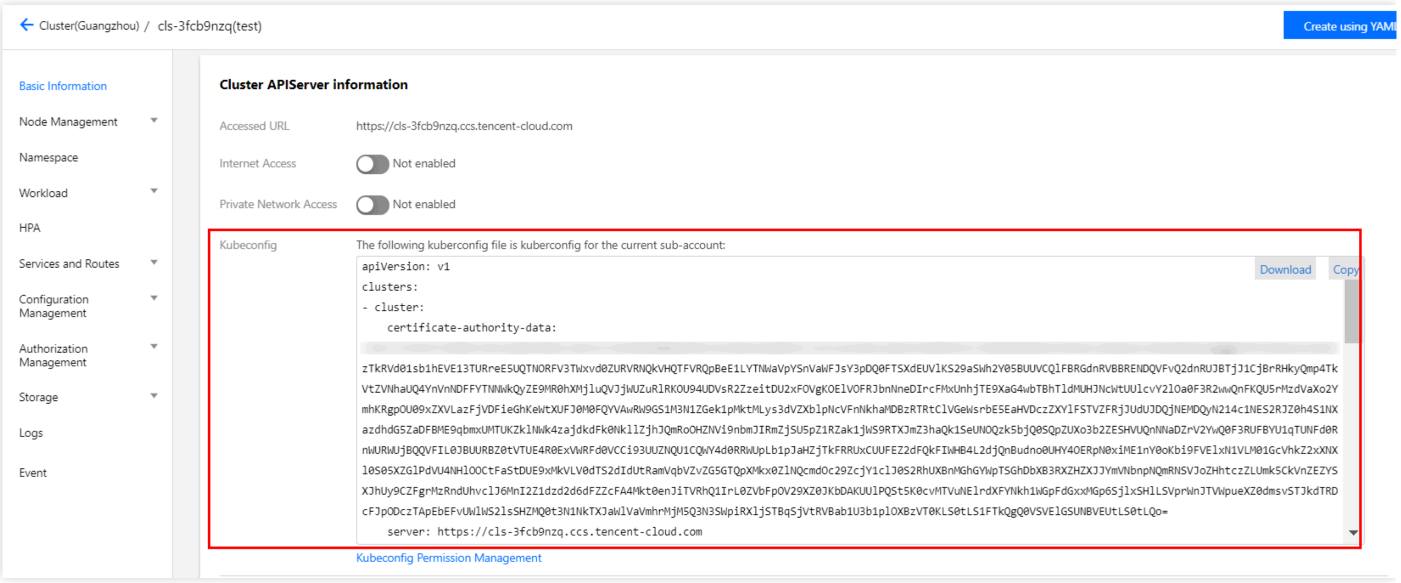
- **Custom:** user-defined ClusterRole.
- **For a specified namespace**:
  - **Developers (tke:ns:dev)**: have read/write permission for the resources visible on the console in a specified namespace.
  - **Read-only users (tke:ns:ro)**: have read-only permission for the resources visible on the console in a specified namespace.
- All the preset ClusterRole policies contain the fixed label: `cloud.tencent.com/tke-rbac-generated: "true"` .
- All the preset ClusterRoleBinding policies contain the fixed annotation: `cloud.tencent.com/tke-account-nickname: yournickname` and the label: `cloud.tencent.com/tke-account: "yourUIN"` .

Directions

Obtaining credentials

TKE will create independent credentials for each sub-account by default. You only need to access the cluster details page or call the Tencent Cloud API DescribeClusterKubeconfig to obtain the credential file `Kubeconfig` of the current account. The procedure for obtaining the file on the console is as follows:

1. Log in to the TKE console and click **Cluster** on the left sidebar.
2. On the **Cluster Management** page, click the ID of the target cluster.
3. On the cluster details page, select **Basic Information** on the left sidebar. Then, you can view and download the Kubeconfig file in the **Cluster APIServer information** section, as shown in the following figure.



Cluster APIServer information

Accessed URL https://cls-3fcb9nzq.ccs.tencent-cloud.com

Internet Access Not enabled

Private Network Access Not enabled

Kubeconfig The following kuberconfig file is kuberconfig for the current sub-account:

```
apiVersion: v1
clusters:
- cluster:
    certificate-authority-data:
[redacted]
zTkRVd01sb1hEVE13TURreE5UQTNORFV3TWxvd0ZURVRNQkVHQTFVRQpBeE1LYTNWaVpYSnVaWFJsY3pDQ0FTSXdEUVlKS29aSWh2Y05BUUVCQlFBRGdnRVBBRENDQVFvQ2dnRUJBTjJ1CjBrRHkyQmp4Tk
VtZVNhaUQ4YnVnNDFFYTNNWkQyZE9MR0hXMjluQVJjWUZuRlRKOU94UDVsR2ZzeitDU2xFOVgKOElVOFRJbnNneDIrcFMxUnhjTE9XaG4wbTBhTldMUHJNcWtUUlcvY2lOa0F3R2wwQnFKQU5rMzdVaXo2Y
mhKRgpOU09xZXVLazFjVDFieGhKeWtXUFJ0M0FQYVAwRW9GS1M3N1ZGek1pMktMLys3dVZXblpNcVFnNkhhMDBzRTRtClVGeWsrbE5EaHVDczZXYlFSTVZFRjJUdUJDQjNEMDQyN214c1NES2RJZ0h4S1NX
azdhdG5ZaDFBME9qbmxUMTUKZklNWk4zajdkdFk0NkllZjhJQmRoOHZNVi9nbmJIRmZjSU5pZ1RZak1jWS9RTXJmZ3haQk1SeUNOQzk5bjQ0SQpZUXo3b2ZESHVUQnNNaDZrV2YwQ0F3RUFBYU1qTUNFd0R
nWURWUjBQQVFIL0JBUURBZ0tVTUE4R0ExVWRFd0VCCi93UUZNQU1CQWY4d0RRWUpLb1pJaHZjTkFRRUxCUUFEZ2dFQkFIWHB4L2djQnBudno0UHY4OERpN0xiME1nY0oKbi9FVElxN1VLM01GcVhkZ2xXNX
l0S05XZGlPdVU4NHlOOCtFaStDUE9xMkVLV0dTS2dIdUtRamVqbVZvZG5GTQpXMkx0ZlNQcmdOc29ZcjY1clJ0S2RhUXBnMGhGYWpTSGhDbXB3RXZHZXJJYmVNbnpNQmRNSVJoZHhtczZLUmk5CkVnZEZYS
XJhUy9CZFgrMzRndUhvclJ6MnI2Z1dzd2d6dFZZcFA4Mkt0enJiTVRhQ1IrL0ZVbFpOV29XZ0JKbDAKUUlPQSt5K0cvMTVuNElrdXFYNkh1WGpFdGxxMGp6SjlxSHlLSVprWnJTVWpueXZ0dmsvSTJkdTRD
cFJpODczTApEbEFvUWlWS2lsSHZMQ0t3N1NkTXJaWlVaVmhrMjM5Q3N3SWpiRXljSTBqSjVtRVBab1U3b1plOXBzVT0KLS0tLS1FTkQgQ0VSVElGSUNBVEUtLS0tLQo=
    server: https://cls-3fcb9nzq.ccs.tencent-cloud.com
```

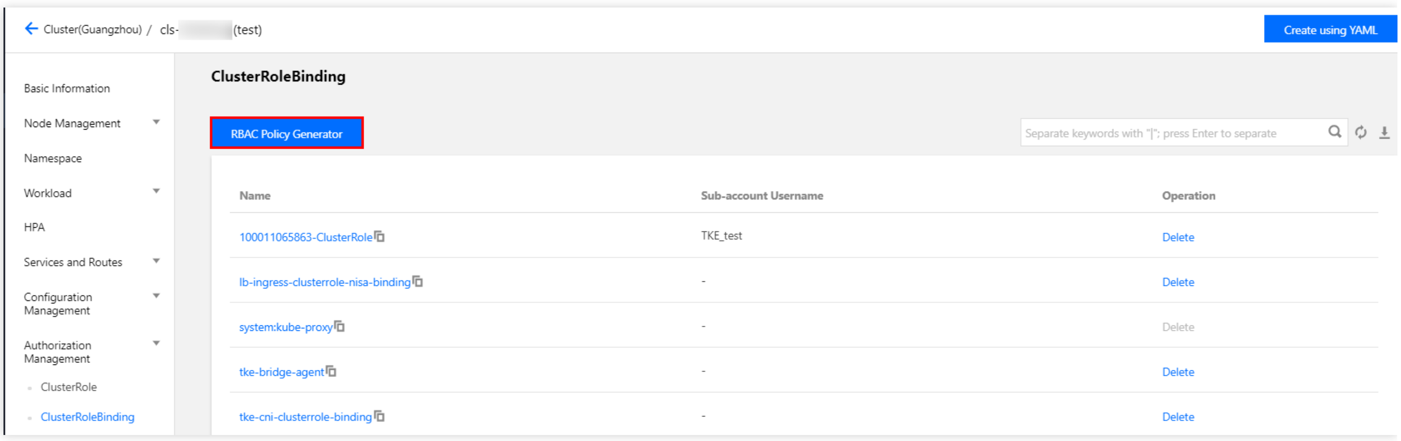
Managing credentials

Cluster administrators can access the credential management page to view and update the cluster credentials of all accounts. For more information, see Updating the TKE cluster access credentials of sub-accounts.

Authorization

> Note：
> Please contact cluster administrators (root accounts, cluster creators, or users with the admin role) for authorization.

1. On the **Cluster Management** page, click the ID of the target cluster.
2. On the cluster details page, select **Authorization Management** -> **ClusterRoleBinding** on the left sidebar.
3. On the **ClusterRoleBinding** page, click **RBAC Policy Generator**, as shown in the following figure.



Name	Sub-account Username	Operation
100011065863-ClusterRole	TKE_test	Delete
lb-ingress-clusterrole-nisa-binding	-	Delete
system:kube-proxy	-	Delete
tke-bridge-agent	-	Delete
tke-cni-clusterrole-binding	-	Delete

4. When you select a sub-account on the **Administration Permissions** page, select the target sub-account and click **Next**.
5. When you set the cluster RBAC, set the permissions as follows:

- **Namespace List**: specify the namespaces for which the permissions apply.
- **Permissions**: please reference the descriptions provided on the page and set permissions as needed.

> Note：
> You can also click **Add Permission** to set custom permissions.

Authentication

Log in to your sub-account and verify that the sub-account has the permissions in question. If so, this indicates that the authorization was successful.

Custom Policy Authorization

Last updated：2021-08-17 15:54:03

This document describes how to grant specified permissions to a sub-account by customizing ClusterRoles and Roles in Kubernetes to fit your specific business requirements.

Policy Syntax Description

You can write your own policy syntax or use the Cloud Access Management (CAM) policy generator to create custom policies. An example YAML is shown below:

Role: for a namespace

```
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
name: testRole
namespace: default
rules:
- apiGroups:
- ""
resources:
- pods
verbs:
- create
- delete
- deletecollection
- get
- list
- patch
- update
- watch
```

ClusterRole: for a cluster

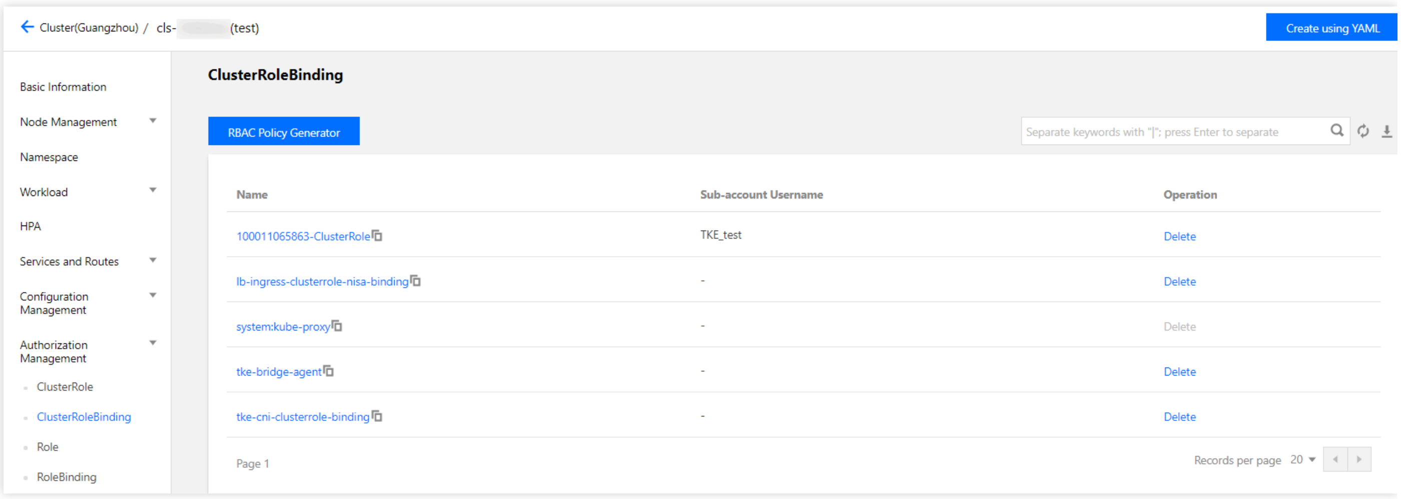
```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
name: testClusterRole
rules:
- apiGroups:
```

```
- ""
resources:
- pods
verbs:
- create
- delete
- deletecollection
- get
- list
- patch
- update
- watch
```

Directions

Note：
This section describes how to bind a custom ClusterRole policy to a sub-account. This operation is basically the same as that for binding a Role policy. Following the directions below, you can bind policies to fit your specific business requirements.

1. Log in to the TKE console and click **Cluster** on the left sidebar.
2. On the **Cluster Management** page, click the ID of the target cluster.
3. On the cluster details page, select **Authorization Management** -> **ClusterRole** on the left sidebar, as shown in the following figure.



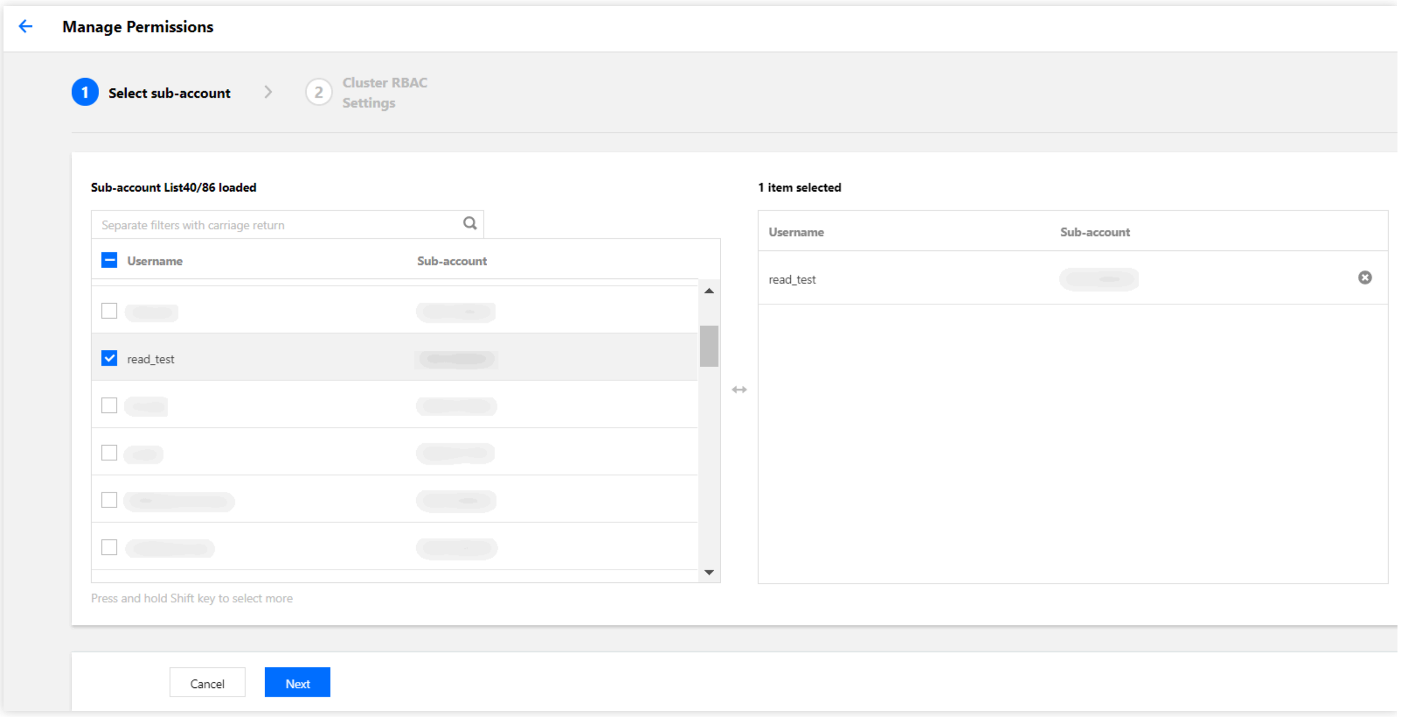
ClusterRoleBinding

Name	Sub-account Username	Operation
100011065863-ClusterRole	TKE_test	Delete
lb-ingress-clusterrole-nisa-binding	-	Delete
system:kube-proxy	-	Delete
tke-bridge-agent	-	Delete
tke-cni-clusterrole-binding	-	Delete

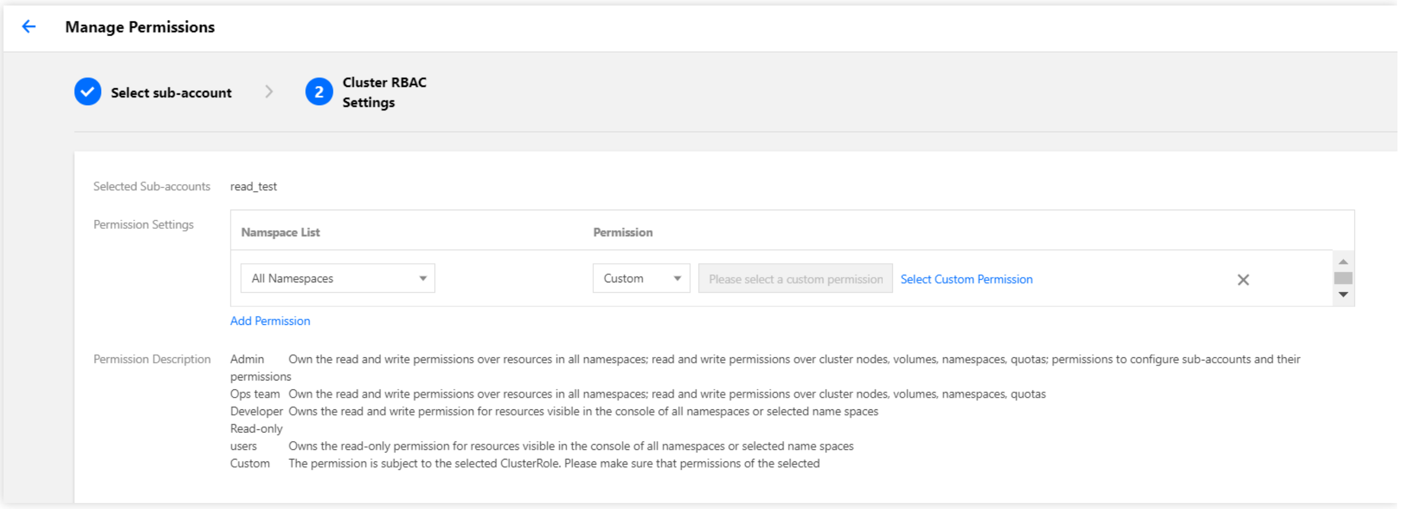
Page 1　　Records per page 20

4. On the **ClusterRole** page, select **Create using YAML** in the upper-right corner.

5. On the editing page, enter the YAML content of the custom policy and then click **Complete** to create the ClusterRole policy.
   For this step, the ClusterRole: for a cluster YAML is used as an example. After the policy is created, you can view the custom permission `testClusterRole` on the **ClusterRole** page.
6. On the **ClusterRoleBinding** page, click **RBAC Policy Generator**.
7. When you select a sub-account on the **Administration Permissions** page, select the target sub-account and click **Next**, as shown in the following figure.



8. On the **Cluster RBAC Setting** page, set the permissions as instructed, as shown in the following figure.



Selected Sub-accounts read_test

Permission Settings

Permission Description
Admin Own the read and write permissions over resources in all namespaces; read and write permissions over cluster nodes, volumes, namespaces, quotas; permissions to configure sub-accounts and their permissions
Ops team Own the read and write permissions over resources in all namespaces; read and write permissions over cluster nodes, volumes, namespaces, quotas
Developer Owns the read and write permission for resources visible in the console of all namespaces or selected name spaces
Read-only users Owns the read-only permission for resources visible in the console of all namespaces or selected name spaces
Custom The permission is subject to the selected ClusterRole. Please make sure that permissions of the selected

- **Namespace List**: specify the namespaces for which the permissions apply.

- **Permissions**: select **Custom** and click **Select Custom Permissions**. Then, select the desired permissions from the custom permission list. Here, we select the previously created custom permission `testClusterRole` as an example.

  > Note：
  > You can also click **Add Permission** to continue customizing the permissions.

9. Click **Done** to complete the authorization.

For your Reference

For more information, see the Kubernetes official documentation: Using RBAC for authorization.

Updating the TKE Cluster Access Credentials of Sub-accounts

Last updated：2022-03-30 18:09:30

Access Credentials

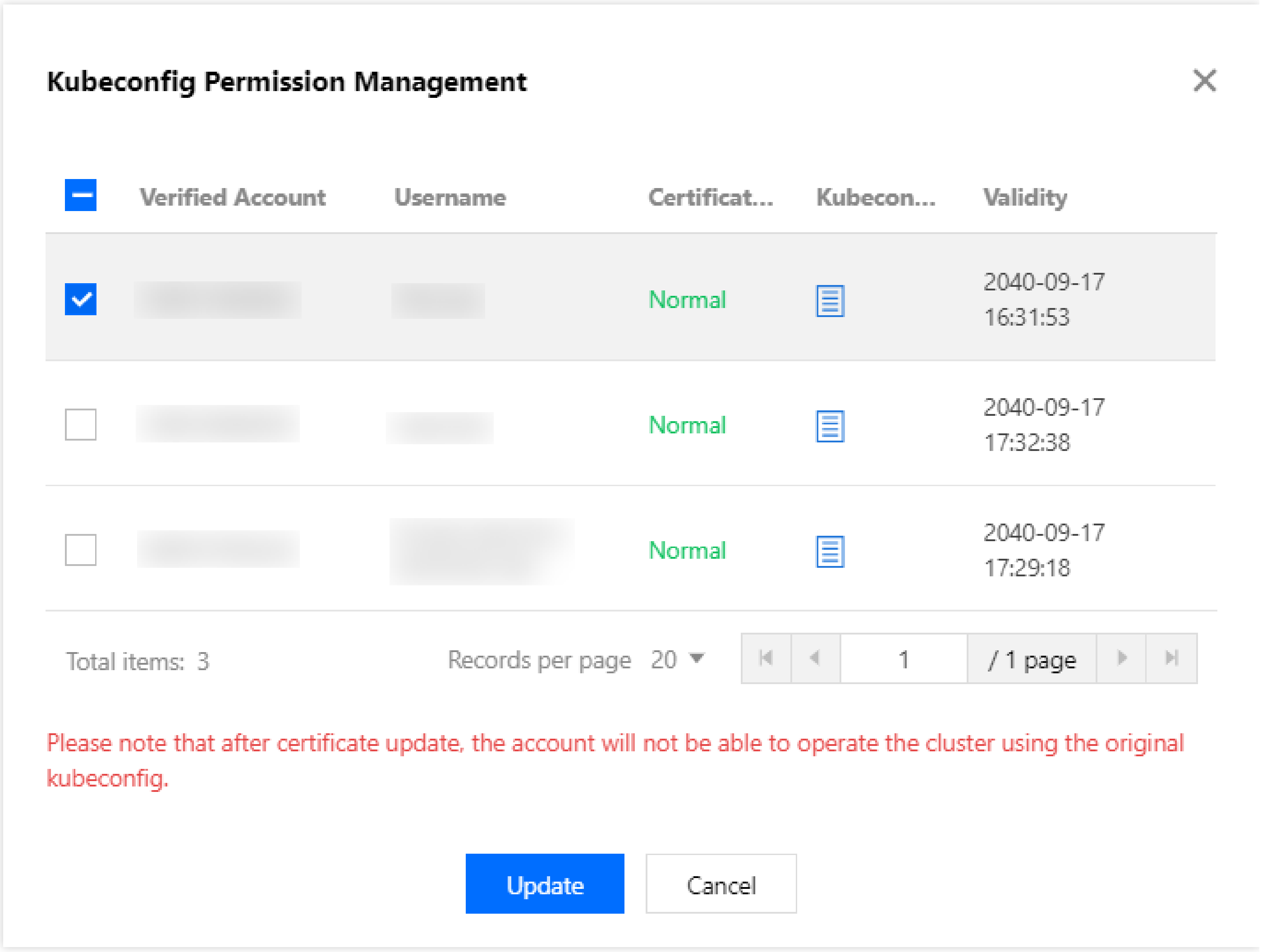
Tencent Kubernetes Engine (TKE) implements the following features based on x509 certificates:

- Each sub-account has a unique client certificate used for accessing Kubernetes API servers.
- Under the new authorization method adopted by TKE, when different sub-accounts obtain access credentials for a cluster (i.e., for accessing the basic information page of the cluster or calling the DescribeClusterKubeconfig API), they will obtain a unique x509 client certificate, which is issued by the self-signed CA of each cluster.
- When a sub-account accesses Kubernetes resources on the console, the backend uses the sub-account’s client certificate to access the Kubernetes API server by default.
- A sub-account can update its unique client certificate to prevent credential disclosure.
- A root account or an account that has `tke:admin` permission for a cluster can view and update the certificates of other sub-accounts.

Directions

1. Log in to the TKE console and click **Cluster** on the left sidebar.
2. On the **Cluster Management** page, click the ID of the target cluster.
3. On the cluster details page, click **Basic Information** on the left sidebar. In the **Cluster APIServer information** section, click **Kubeconfig**.

4. On the **Kubeconfig** page, select the authentication account and click **Update**, as shown in the following figure.



Kubeconfig Permission Management

▬	Verified Account	Username	Certificat...	Kubecon...	Validity
✓	[redacted]	[redacted]	Normal	▤	2040-09-17 16:31:53
☐	[redacted]	[redacted]	Normal	▤	2040-09-17 17:32:38
☐	[redacted]	[redacted]	Normal	▤	2040-09-17 17:29:18

Total items: 3 Records per page 20 1 / 1 page

Please note that after certificate update, the account will not be able to operate the cluster using the original kubeconfig.