Cloud Container Instance

User Guide

Issue 01

Date 2023-10-13

HUAWEI CLOUD COMPUTING TECHNOLOGIES CO., LTD.

Trademarks and Permissions

Notice

The purchased products, services and features are stipulated by the contract made between Huawei Cloud and the customer. All or part of the products, services and features described in this document may not be within the purchase scope or the usage scope. Unless otherwise specified in the contract, all statements, information, and recommendations in this document are provided "AS IS" without warranties, guarantees or representations of any kind, either express or implied.

The information in this document is subject to change without notice. Every effort has been made in the preparation of this document to ensure accuracy of the contents, but all statements, information, and recommendations in this document do not constitute a warranty of any kind, express or implied.

Contents

1 Permissions Management

1.1 Permissions Management for CCI

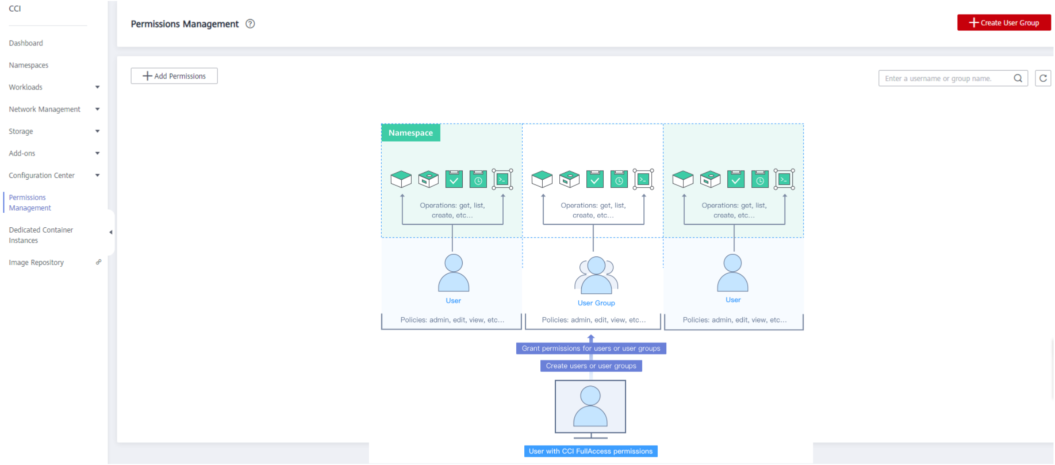
CCI permissions management allows you to grant permissions to your IAM users and user groups. It combines the advantages of Kubernetes Role-based Access Control (RBAC) authorization and Identity and Access Management (IAM) to provide a variety of authorization methods, including IAM fine-grained authorization, IAM token authorization, namespace-level authorization, and namespaced resource authorization.

- **Namespace-level permissions**: permissions granted based on Kubernetes RBAC roles. You can authorize users or user groups to perform operations on Kubernetes resources under specific namespace.
- **CCI permissions:** permissions granted based on IAM fine-grained authorization. You can authorize users to perform operations on namespaces, such as creating and deleting namespaces.

NOTE

- If you enable RBAC when you create a namespace, access to resources under the namespace is controlled by RBAC policies. If RBAC is disabled, RBAC policies will not take effect.
- After you create a namespace with RBAC enabled, you must authorize IAM users to perform operations on the namespace.
- The network, ClusterRole, and RoleBinding resources are not affected by RBAC policies but are controlled only by IAM fine-grained authentication. The network resources are controlled by network-related actions, and ClusterRole and RoleBinding are controlled by RBAC-related actions.
- You can grant permissions for all namespaces of an IAM user at the same time.

Figure 1-1 CCI permissions management


CCI
Dashboard
Namespaces
Workloads
Network Management
Storage
Add-ons
Configuration Center
Permissions Management
Dedicated Container Instances
Image Repository
Permissions Management
Create User Group
Add Permissions
Enter a username or group name.
Namespace
Operations: get, list, create, etc...
Operations: get, list, create, etc...
Operations: get, list, create, etc...
User
User Group
User
Policies: admin, edit, view, etc...
Policies: admin, edit, view, etc...
Policies: admin, edit, view, etc...
Grant permissions for users or user groups
Create users or user groups
User with CCI FullAccess permissions

Namespace Permissions

Kubernetes RBAC APIs define four objects: Role, ClusterRole, RoleBinding, and ClusterRoleBinding. Currently, CCI supports only ClusterRole and RoleBinding. The two objects are described as follows:

- **ClusterRole** specifies which actions can be performed on which resources. In the RBAC API, a role contains rules that represent a set of permissions. A role within a Kubernetes cluster is defined by a ClusterRole.
- **RoleBinding** binds roles to subjects (including users and user groups). A RoleBinding grants the permissions defined in a role to a user or user group. The user or group has the permissions granted through the bound ClusterRole.

Table 1-1 Two objects declared by the RBAC API

Type	Description
ClusterRole	A ClusterRole can be used to grant access to resources in a cluster.
RoleBinding	A RoleBinding binds a ClusterRole to subjects (users) in a namespace, granting the ClusterRole's permissions to those users.

⚠ CAUTION

Currently, you can only use ClusterRole to create a RoleBinding in a namespace.

In CCI, you can regulate users' or user groups' access to Kubernetes resources in a single namespace based on their Kubernetes RBAC roles.

Currently, there are four roles: **cluster-admin**, **admin**, **edit**, and **view**. For details, see **Table 1-2**.

Table 1-2 User/user group roles

Default ClusterRole	Description
cluster-admin	Allows access to all Kubernetes resource objects.
admin	Allows admin access that can be granted within a namespace using a RoleBinding. If used in a RoleBinding, it allows read/write access to most resources in a namespace. It does not allow write access to resource quota or to the namespace itself.
edit	Allows read/write access to most resources in a namespace.
view	Allows read-only access to most objects in a namespace. It does not allow access to secrets.

For more information about Kubernetes RBAC authorization, see **Using RBAC Authorization**.

1.2 Creating a User and Granting CCI Permissions

This section describes how to use **IAM** to implement fine-grained permissions control for your CCI resources. With IAM, you can:

- Create IAM users for employees based on your enterprise's organizational structure. Each IAM user will have their own security credentials for accessing CCI resources.
- Grant only the permissions required for users to perform a specific task.
- Entrust a cloud account or cloud service to perform efficient O&M on your CCI resources.

If your account does not require individual IAM users, skip this section.

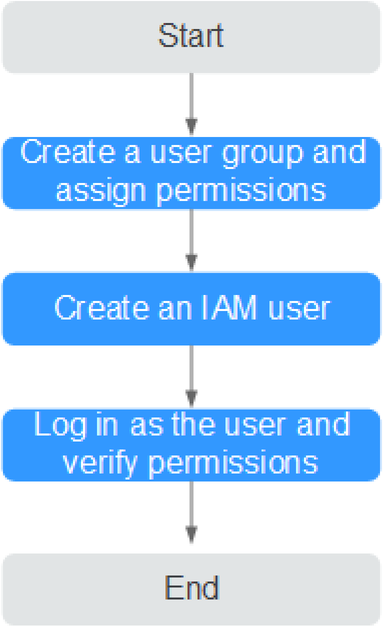
This section describes the procedure for granting permissions (see **Figure 1-2**).

Prerequisites

Learn about the permissions (see **Permissions Management**) supported by CCI.

Process Flow

Figure 1-2 Process of granting CCI permissions


Start
Create a user group and assign permissions
Create an IAM user
Log in as the user and verify permissions
End

1. **Create a user group and assign permissions**.

   Create a user group (for example, **Developers**) on the IAM console and assign the **CCI CommonOperations** policy to the group. CCI is a project-level service. When assigning CCI system-defined policies to users, you also need to assign the **IAM ReadOnlyAccess** policy to the users.

2. **Create an IAM user**.

   Create a user (for example, **James**) on the IAM console and add the user to the group created in **1**.

3. **Log in** and verify permissions.

   Log in to the management console as the user you created and verify that the user has the assigned permissions.

   – Choose **Service List** > **Cloud Container Instance**. In the navigation pane on the left, choose **Workloads** > **Deployments**. On the page displayed, click **Create from Image**. If the Deployment is created successfully, the **CCI CommonOperations** policy has taken effect.

   – Choose **Service List** > **Cloud Container Instance**. In the navigation pane on the left, choose **Namespaces**. On the page displayed, click **Create** for the target namespace type. If the namespace cannot be created, the **CCI CommonOperations** policy has taken effect.

1.3 Granting Namespace Permissions to a User or User Group

This section describes how to grant CCI users and user groups the permissions to various namespaced resources. **Process Flow** describes the process for granting permissions.

Configuration Description

- You need to have a cloud account. Only the account and IAM users who have been assigned the **CCI FullAccess** policy or all RBAC policies can grant permissions to other users.
- In this example, both a user group and a user are granted permissions to access namespaced resources. You have the choice to grant permissions to either users or user groups.
- You can use the process flow only to add namespace permissions policies for users or user groups. To edit permissions policies of users or user groups, click **Edit Policy** under **Operation** on the **Permissions Management** page.
- If you grant multiple permissions policies to a user or user group, all these policies will take effect at the same time. The permissions policies that you grant to a user group apply to all users in the user group.
- If you enable RBAC authentication, the union set of permissions policies of the same type will be used, and the intersection set of permissions policies of different types will be used. For example, if you add multiple IAM fine-grained policies to a user group, all these policies will take effect. Similarly, if you add multiple permissions policies to a user or user group on the CCI permissions management page, all these policies will take effect. A user with the **CCI CommonOperations** policy can create Deployments. However, if the user and the user group to which the user belongs are not granted the RBAC policy in the target namespace, the user will fail the authentication and cannot create the Deployment. That is, the intersection set of permissions policies of different types will be used.

Process Flow

A namespace is an abstract collection of resources and objects. You can create multiple namespaces in a cluster. Data is isolated between namespaces so namespaces can share the same cluster service without affecting each other. A namespace can act as a virtual cluster to meet diversified requirements.

This section describes how you can grant namespace permissions to the IAM user **James** and the user group **Developers** created in **Creating a User and Granting CCI Permissions**.

- **Step 1: Grant Namespace Permissions to an IAM User or User Group**
- **Step 2: Log In and Verify Permissions**

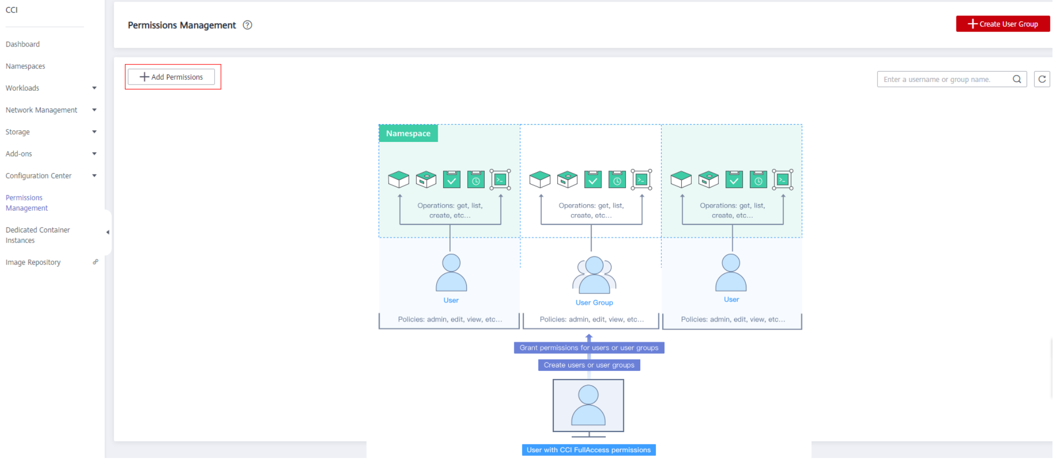
Step 1: Grant Namespace Permissions to an IAM User or User Group

In this step, you will grant user **James** who has the **CCI CommonOperations** policy permissions to view the target namespace.

Step 1 Log in to the CCI console. In the navigation pane, choose **Permissions Management**.

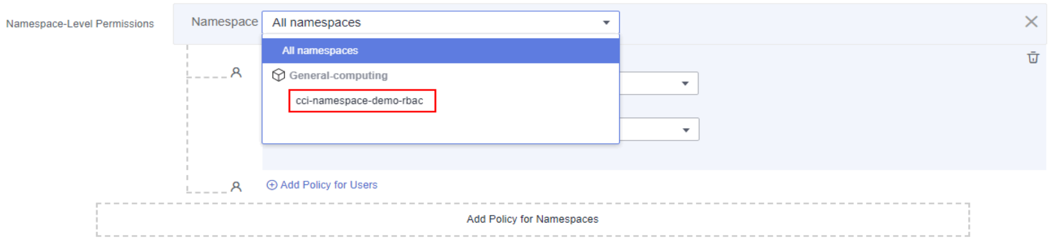
Step 2 Click **Add Permissions**.

Figure 1-3 Adding permissions


CCI
Dashboard
Namespaces
Workloads
Network Management
Storage
Add-ons
Configuration Center
Permissions Management
Dedicated Container Instances
Image Repository
Permissions Management
Create User Group
Add Permissions
Enter a username or group name.
Namespace
Operations: get, list, create, etc...
Operations: get, list, create, etc...
Operations: get, list, create, etc...
User
User Group
User
Policies: admin, edit, view, etc...
Policies: admin, edit, view, etc...
Policies: admin, edit, view, etc...
Grant permissions for users or user groups
Create users or user groups
User with CCI FullAccess permissions

Step 3 Select the namespace whose access you want to manage. In this example, select **cci-namespace-demo-rbac**.

Figure 1-4 Selecting a namespace



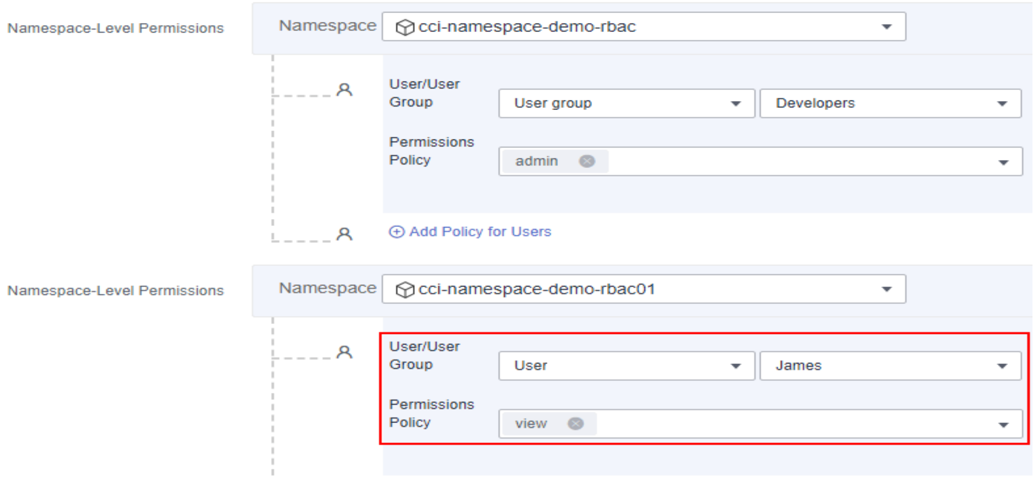
Step 4 Add the **admin** permissions policy for the user group **Developers**.

- **User/User Group**: Select **User group** from the drop-down list and then select **Developers** from the user group list.
- **Permissions Policy**: Select **admin**.

Step 5 Click **Add Policy for Namespaces**. Add permissions on another namespace (**cci-namespace-demo-rbac01**) for user **James**.

- **Namespace**: Select **cci-namespace-demo-rbac01**.
- **User/User Group**: Select **User** from the drop-down list and then select **James** from the user list.
- **Permissions Policy**: Select **view**.

Figure 1-5 Adding a namespace permissions policy



Namespace-Level Permissions

Namespace: cci-namespace-demo-rbac

User/User Group: User group, Developers

Permissions Policy: admin

Namespace-Level Permissions

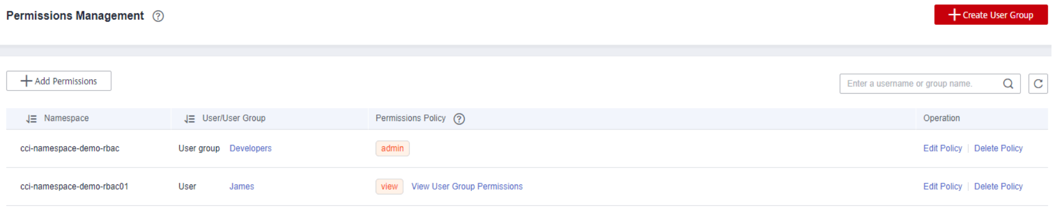
Namespace: cci-namespace-demo-rbac01

User/User Group: User, James

Permissions Policy: view

Step 6 Click **OK**. The namespace permissions granted to the user group and user are displayed in the permissions list.

Figure 1-6 Namespace permissions list



Permissions Management

Namespace	User/User Group		Permissions Policy	Operation
cci-namespace-demo-rbac	User group	Developers	admin	Edit Policy | Delete Policy
cci-namespace-demo-rbac01	User	James	view View User Group Permissions	Edit Policy | Delete Policy

NOTE

To sum up, the authorization result is as follows:

- The user group **Developers** has **admin** permissions on the namespace **cci-namespace-demo-rbac**. The **admin** permissions policy also applies to the IAM user **James** in the user group **Developers**.
- The IAM user **James** has **view** permissions on the namespace **cci-namespace-demo-rbac01**.

----End

Step 2: Log In and Verify Permissions

Use the username **James** and identity credential to log in to the CCI console and verify that the IAM user **James** has the namespace permissions.

Step 1 Enter the account name, username, and password, and click **Log In**.

- The account is that you used to create the IAM user.
- The username and password are those set by the account when creating the IAM user. You will be prompted to change the initial password at initial login.

If the login fails, contact the account owner to verify the username and password. Alternatively, you can reset the password.

Step 2 After you log in, switch to a region where the user has been granted permissions.

Step 3 Choose **Service List** > **Cloud Container Instance** to launch the CCI console. Then verify that the IAM user has namespace permissions.

----End

1.4 Granting Namespace Permissions to an Agency Account

Namespaced resource permissions management is an authorization method based on Kubernetes RBAC roles. You can authorize agency accounts to perform operations on Kubernetes resources under specific namespace.

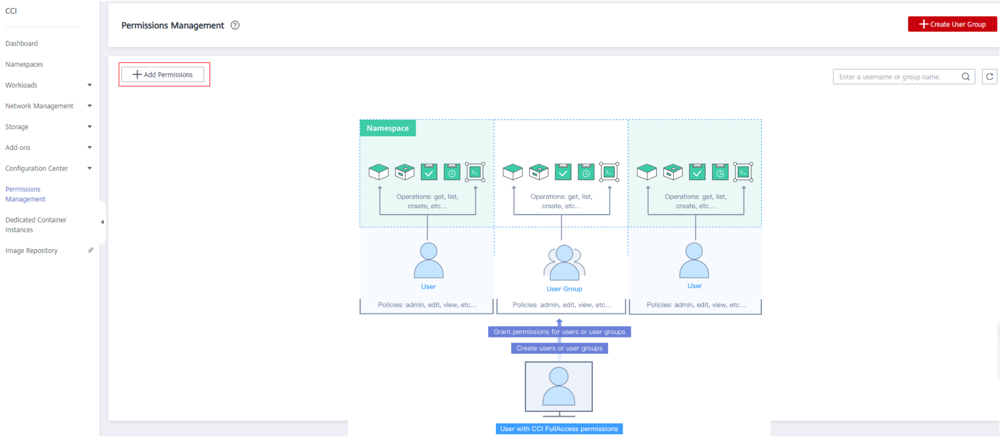
This section describes how to grant CCI agency accounts the permissions to various namespaced resources.

Procedure

Step 1 Log in to the CCI console. In the navigation pane, choose **Permissions Management**.

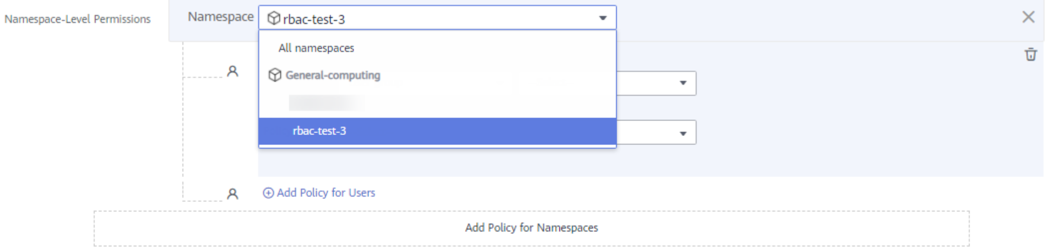
Step 2 Click **Add Permissions**.

Figure 1-7 Adding permissions


CCI
Dashboard
Namespaces
Workloads
Network Management
Storage
Add-ons
Configuration Center
Permissions Management
Dedicated Container Instances
Image Repository
Permissions Management
Create User Group
Add Permissions
Enter a username or group name.
Namespace
Operations: get, list, create, etc...
User
User Group
Policies: admin, edit, view, etc...
Grant permissions for users or user groups
Create users or user groups
User with CCI FullAccess permissions

Step 3 Select the namespace whose access you want to manage. In this example, select **rbac-test-3**.

Figure 1-8 Selecting a namespace



Step 4 Add the **admin** permissions policy for **Account**.

- **User/User Group**: Select **Account** from the drop-down list and then select the target delegated account from the account list.
- **Permissions Policy**: Select **admin**.

Step 5 Click **OK**. The namespaced permissions are granted to the delegated account.

----End

1.5 CCI Custom Policies

You can create custom policies to supplement the system-defined policies of CCI. For the actions that can be added to custom policies, see **Permissions Policies and Supported Actions**.

You can create custom policies in either of the following ways:

- Visual editor: Select a cloud service, specify actions and resources, and add request conditions. You do not need to have knowledge of JSON syntax.
- JSON: Create a policy in the JSON format from scratch or based on an existing policy.

The following provides some example custom policies of CCI.

Example Custom Policies

- Example 1: Updating a namespace

```
{
    "Version": "1.1",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "cci:namespace:update"
            ]
        }
    ]
}
```

- Example 2: Denying namespace deletion

  A policy with only "Deny" permissions must be used in conjunction with other policies for it to take effect. If you assign both "Allow" and "Deny" to a user, the "Deny" permissions take precedence over the "Allow" permissions.

  The following method can be used if you need to assign permissions of the **CCI FullAccess** policy to a user but you want to prevent the user from deleting namespaces (**cci:namespace:delete**). Create a custom policy for denying namespace deletion, and attach both policies to the group to which the user belongs. Then, the user can perform all operations on CCI except deleting namespaces. The following is an example of a deny policy:

```
{
    "Version": "1.1",
    "Statement": [
        {
            "Action": [
                "cci:namespace:delete"
            ],
            "Effect": "Deny"
        }
    ]
}
```

- Example 3: Defining permissions for multiple services in a policy

  A custom policy can contain the actions of multiple services that are of the global or project-level type. The following is an example policy containing actions of multiple services:

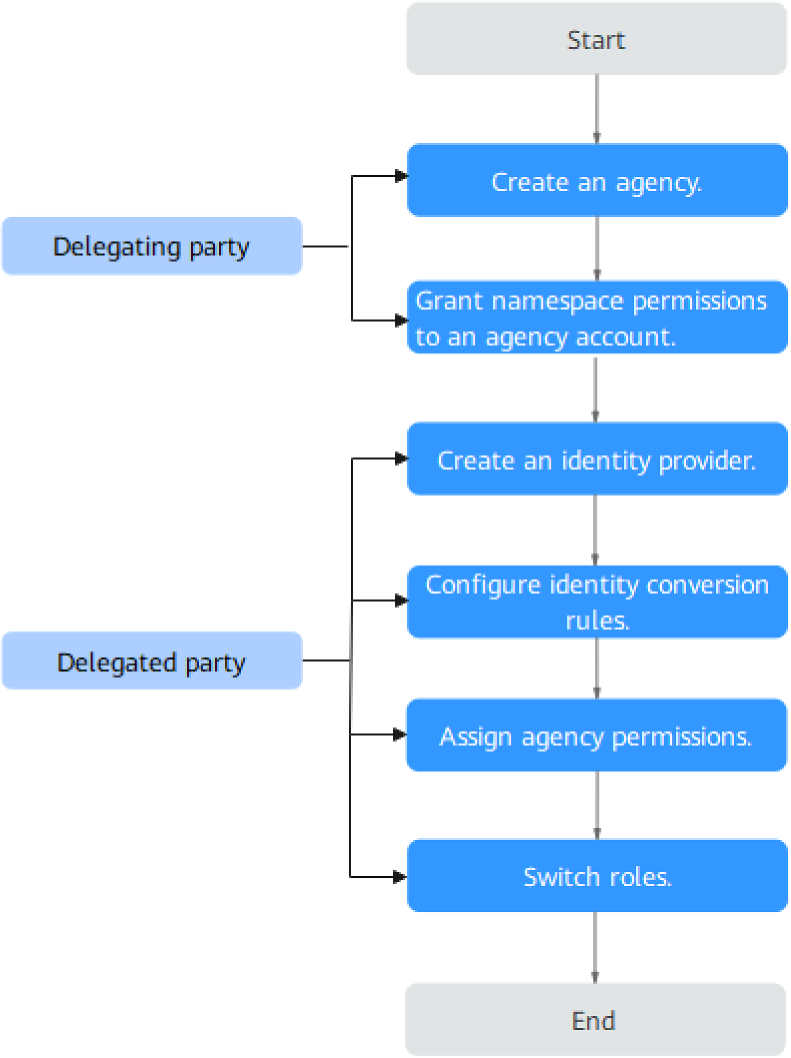
  ```
  {
     "Version": "1.1",
     "Statement": [
        {
           "Action": [
              "ecs:cloudServers:resize",
              "ecs:cloudServers:delete",
              "ecs:cloudServers:delete",
              "ims:images:list",
              "ims:serverImages:create"
           ],
           "Effect": "Allow"
        }
     ]
  }
  ```

1.6 Delegating a Federated User to Manage Resources

If you want to delegate a federated user of another account (account B) to manage resources in your account (account A), log in using account A, create an agency for account B, and grant namespace permissions to account B. Then, log in using account B and perform federated identity authentication for it. After the authentication is complete, account B assigns the agency permissions to the federated user so that the federated user can switch to the agency of account A. Log in to Huawei Cloud as the federated user and switch the role to manage resources in account A.

This section describes how to delegate federated users to manage resources. **Figure 1-9** shows the operation process.

Figure 1-9 Process of delegating a federated user to manage resources


Start
Delegating party
Create an agency.
Grant namespace permissions to an agency account.
Create an identity provider.
Delegated party
Configure identity conversion rules.
Assign agency permissions.
Switch roles.
End

Procedure

To delegate account B to manage resources in account A as a federated user, perform the following steps:

Step 1 **Create an agency (by the delegating party).**

Log in to the IAM console as the delegating party (account A). Create an agency, enter the account name of the delegated party (account B), and grant permissions of the **CCI FullAccess** policy to the delegated party. Users granted these permissions can create, delete, query, and update all CCI resources.

Step 2 **Grant namespaced resource permissions to an agency account (by the delegating party).**

Log in to the CCI console as the delegating party (account A). On the **Permissions Management** page, grant permissions of resources in the namespace to the delegated party (account B). You can set permissions for different delegated accounts to operate Kubernetes resources under a specified namespace.

Step 3 **Perform federated identity authentication (by the delegated party)**.

Log in to the delegated party (account B) and perform federated identity authentication.

Before delegating a federated user to manage resources, you need to perform federated identity authentication on the delegated party. The authentication process consists of two steps: Establish a trust relationship and create an identity provider, and then configure identity conversion rules.

NOTE

After an identity provider is created, a default identity conversion rule is also created. You need to click **Edit Rule** to update or delete the default rule and create one. If you add a new rule with the default rule not deleted, the default rule may be matched, and the new rule does not take effect.

Step 4 **Assign permissions to a user (by the delegated party)**.

If a user under the delegated party (account B) wants to manage account A's resources, the delegated party (account B) must assign agency permissions to the user. To enable a federated user to manage resources of the delegating party (account A), the delegated party (account B) needs to assign the permissions of the custom policy **federation_agency** to the user group (**federation_group**) to which the federated user belongs. **federation_group** is also the federated user group that is written into the identity conversion rules.

Step 5 **Switch roles (by the delegated party)**.

Account B and the federated user with agency permissions can switch their roles to the delegating party (account A) to manage its resources.

----End

2 Environment Configuration

Logging In to the CCI Console

Log in to the CCI console and grant CCI the permission to access other cloud services.

Step 1 Log in to the management console.

Step 2 Click 📍 in the upper left corner and select a region.

CCI is available only in regions **LA-Sao Paulo 1**, **AP-Bangkok**, and **AP-Singapore**.

NOTE

CCI does not allow you to create resources in sub-projects.

Step 3 Switch to the CCI console.

Step 4 If this is the first time you are logging in to the CCI console, click **Agree** to grant CCI the permission to access other cloud services.

After the permission is granted, an agency named **cci_admin_trust** is created. You can view the agency on the IAM console.

----End

(Optional) Uploading Images

The cloud platform provides the SoftWare Repository for Container (SWR) service for you to upload container images to the image repository. You can easily pull these images when creating workloads on CCI. For details about how to upload images, see **Uploading an Image Through a Container Engine Client**.

NOTICE

Currently, CCI does not support third-party image repositories.

After **Enterprise Management** is enabled, if an IAM user needs to use private images in your account, you need to log in to the CCI console using the account, choose **Image Repository**, and grant the required permission to the user on the SWR console.

You can use either of the following methods to grant permission to an IAM user:

- On the details page of an image, click the **Permissions** tab, click **Add Permission**, and then grant the read, write, or manage permission to the user. For details, see **Granting Permissions for a Specific Image**.
- On the details page of an organization, click the **Users** tab, click **Add Permission**, and then grant the read, write, or manage permission to the user. For details, see **Granting Permissions for an Organization**.

(Optional) Creating a Load Balancer

A load balancer allows your workloads to be accessed from external networks. For details about how to create a load balancer, see **Creating a Load Balancer**.

Step 1 Log in to the management console.

Step 2 Choose **Service List** > **Networking** > **Elastic Load Balance**.

Step 3 On the **Elastic Load Balance** page, click **Buy Elastic Load Balancer**.

Specify the required parameters to create a load balancer.

NOTE

A load balancer can work on a public or private network.

----End

(Optional) Preparing SSL Certificates

CCI allows workloads to be accessed over HTTPS. You can specify your own SSL certificate when you create a workload.

SSL certificates are classified into authoritative and self-signed certificates. Authoritative certificates are issued by CAs. You can obtain authoritative certificates from third-party certificate agents. A client trusts websites that use authoritative certificates by default. Self-signed certificates are self-issued by users, typically using OpenSSL. By default, self-signed certificates are untrusted by the client. The browser will display an alarm message when you access a website that uses a self-signed certificate. You can continue to access the website by ignoring the alarm.

For details about SSL certificates, see **SSL Certificates**.

3 Namespace

Namespaces are used to logically divide your resources into different groups, especially in scenarios where a large number of users work across multiple projects.

Currently, CCI provides **general-computing** resources. When creating a namespace, you must choose a resource type so that workloads you create can run in the corresponding cluster.

- General-computing: Pods with CPU resources can be created, which are ideal for general computing scenarios.

NOTE

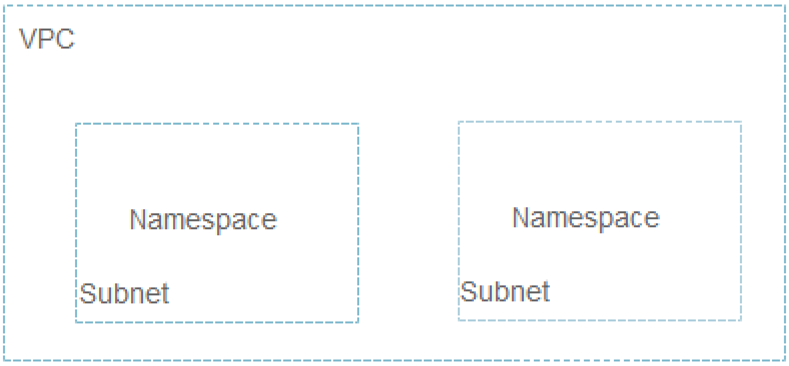
- One account can create a maximum of five namespaces in one region.
- General-computing resources support x86-based images.

Relationship Between Namespaces and Networks

Each namespace requires a separate subnet, as shown in **Figure 3-1**. When you create a namespace, you need to associate it with an existing VPC or a new VPC. If you create a VPC, create a subnet for the namespace. Containers and other resources created in this namespace will run in the VPC and subnet you select.

If you want to run resources of multiple services in the same VPC, you need to consider network planning, including subnet CIDR block division and IP address planning.

Figure 3-1 Relationship between namespaces and VPC subnets



Application Scenarios

Namespaces can implement partial environment isolation. If you have a large number of projects and personnel, you can create different namespaces based on project attributes, such as production, test, and development.

Creating a Namespace

Step 1 Log in to the CCI console. In the navigation pane, choose **Namespaces**.

Step 2 On the page displayed on the right, click **Create** for the target namespace type.

Step 3 Enter a name for the namespace.

NOTE

The namespace name must be globally unique in CCI.

Step 4 Set RBAC policies.

After RBAC is enabled, the access to resources in the namespace is controlled by the RBAC policies. For details, see **Namespace Permissions**.

Step 5 Select an enterprise project. In CCI, each namespace can belong to one enterprise project, but an enterprise project can have multiple namespaces.

NOTE

- Skip this step if the Enterprise Project Management Service (EPS) is not enabled. To enable the service, see **Enabling Enterprise Center**. If you are an IAM user, pay attention to the notice provided in **(Optional) Uploading Images**.
- After you specify an enterprise project, both the namespace and the network and storage resources that are automatically created for the namespace belong to the enterprise project. You must migrate these resources together with the namespace. For example, when you migrate a namespace from project 1 to project 2, you must also migrate the network and storage resources associated with the namespace to project 2. Otherwise, the workloads in this namespace may not run properly.

Step 6 Configure a VPC.

You can use an existing VPC or create a VPC. If you create a VPC, it is recommended that you set the VPC CIDR block to 10.0.0.0/8–22, 172.16.0.0/12–22, or 192.168.0.0/16–22.

NOTICE

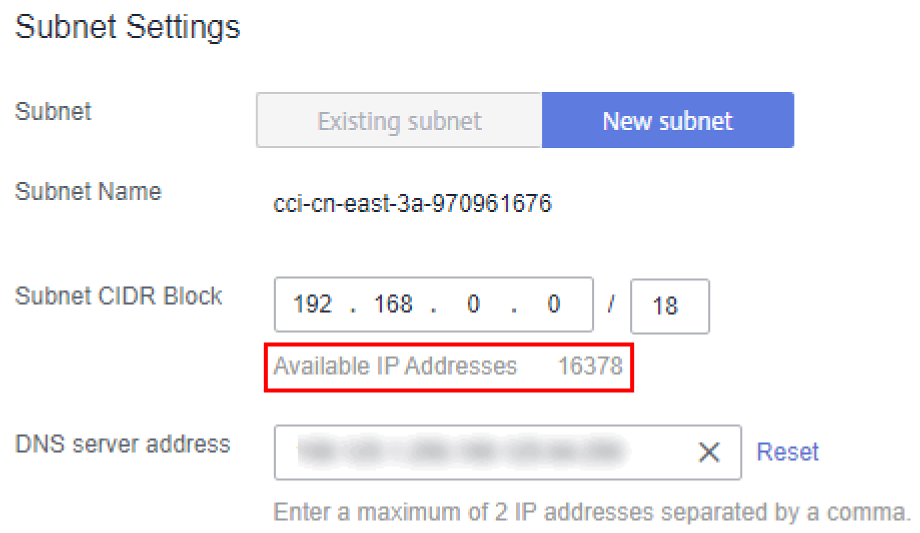
You must not set the VPC CIDR block and subnet CIDR block to 10.247.0.0/16, because this CIDR block is reserved for workload access. If you select this CIDR block, IP address conflicts may occur, which may result in failure to create a workload or service unavailability. If you do not need to access pods through workloads, you can select this CIDR block.

After the namespace is created, you can view VPC and subnet information by choosing **Network Management** > **Networks**.

Step 7 Configure a subnet CIDR block.

Ensure sufficient available IP addresses to create workloads.

Figure 3-2 Configuring a subnet



Subnet Settings

Subnet: Existing subnet | New subnet

Subnet Name: cci-cn-east-3a-970961676

Subnet CIDR Block: 192 . 168 . 0 . 0 / 18

Available IP Addresses 16378

DNS server address: [redacted] Reset

Enter a maximum of 2 IP addresses separated by a comma.

NOTE

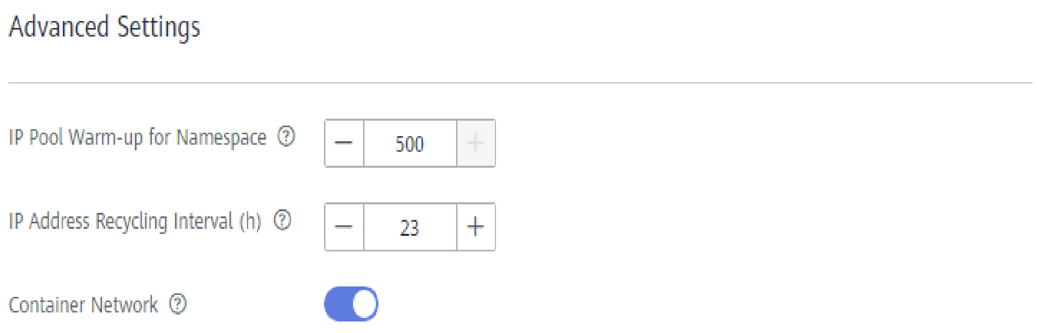
- Some IP addresses (**10** by default) in the configured subnet will be warmed up for the created namespace.
- You can set the number of IP addresses to be warmed up in **Step 8**.
- Warming up IP addresses for the created namespace affects the deletion of the configured subnet and VPC. They can be deleted only after the namespace is deleted.

Step 8 Configure advanced settings.

Each namespace provides an IP resource pool. You can customize the pool size to reduce the duration for applying for IP addresses and improve the workload creation efficiency.

For example, 200 pods are running every day. During peak traffic hours, the IP resource pool instantly scales out to provide 500 IP addresses. After a specified interval (for example, 23 hours), the number of IP addresses that exceed the pool size (500 - 200 = 300) will be reclaimed.

Figure 3-3 Configuring advanced settings



Advanced Settings

IP Pool Warm-up for Namespace: 500

IP Address Recycling Interval (h): 23

Container Network: on

- **IP Pool Warm-up for Namespace**: CCI creates an IP pool with the number of IP addresses you specify here for the namespace, and will warm up these IP addresses to accelerate workload creation. The IP pool can contain a maximum of 500 IP addresses.

- **IP Address Recycling Interval (h)**: Warmed-up IP addresses that become idle can be recycled within the duration that you specify here.
- **Container Network**: Enable this option if you want CCI to start the container network in advance so that containers can connect to the network as soon as they are started.

Step 9 Click **Create**.

After the creation is complete, you can view the VPC and subnet information on the namespace details page.

----End

Deleting a Namespace

NOTICE

Deleting a namespace will remove all data resources related to the namespace, including workloads, ConfigMaps, secrets, and SSL certificates.

Step 1 Log in to the CCI console. In the navigation pane, choose **Namespaces**. On the page displayed, click the namespace you want to delete.

Step 2 In the upper right corner, click **Delete**. In the dialog box that is displayed, enter **DELETE** and click **Yes**.

NOTE

To delete a VPC or subnet, go to the **VPC** console.

----End

Creating a Namespace Using kubectl

For details, see **Namespace and Network**.

4 Workload

4.1 Pods

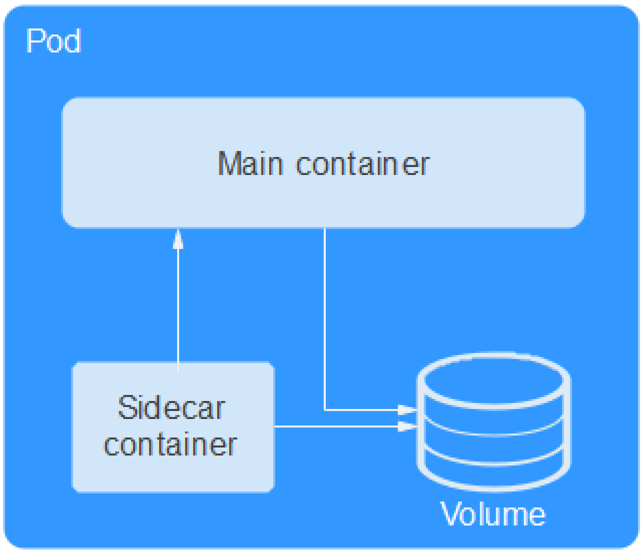
What Is Pod?

A pod is the smallest and simplest unit in the Kubernetes object model that you create or deploy. A pod encapsulates one or more containers, storage resources, a unique network IP address, and options that govern how the container(s) should run.

Pods can be used in either of the following ways:

- One container runs in one pod. This is the most common usage of pods in Kubernetes. You can view the pod as a single encapsulated container, but Kubernetes directly manages pods instead of containers.
- Multiple containers that need to be coupled and share resources run in a pod. In this scenario, an application contains a main container and several sidecar containers, as shown in **Figure 4-1**. For example, the main container is a web server that provides file services from a fixed directory, and the sidecar container periodically downloads files to the directory.

Figure 4-1 Pod

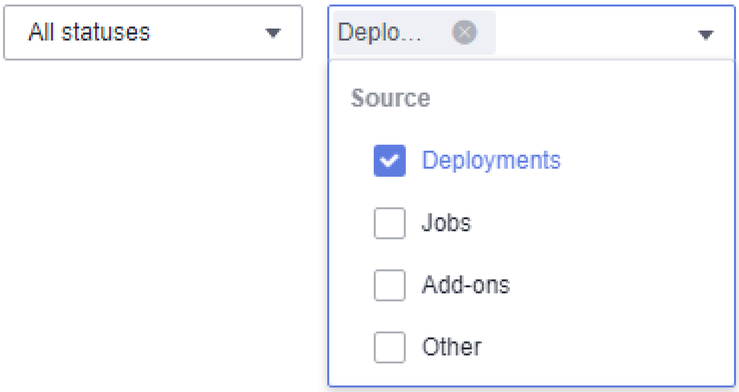


In Kubernetes, pods are rarely created directly. Instead, controllers such as Deployments and jobs, are used to manage pods. Controllers can create and manage multiple pods, and provide replica management, rolling upgrade, and self-healing capabilities. A controller generally uses a pod template to create corresponding pods.

Viewing Pods

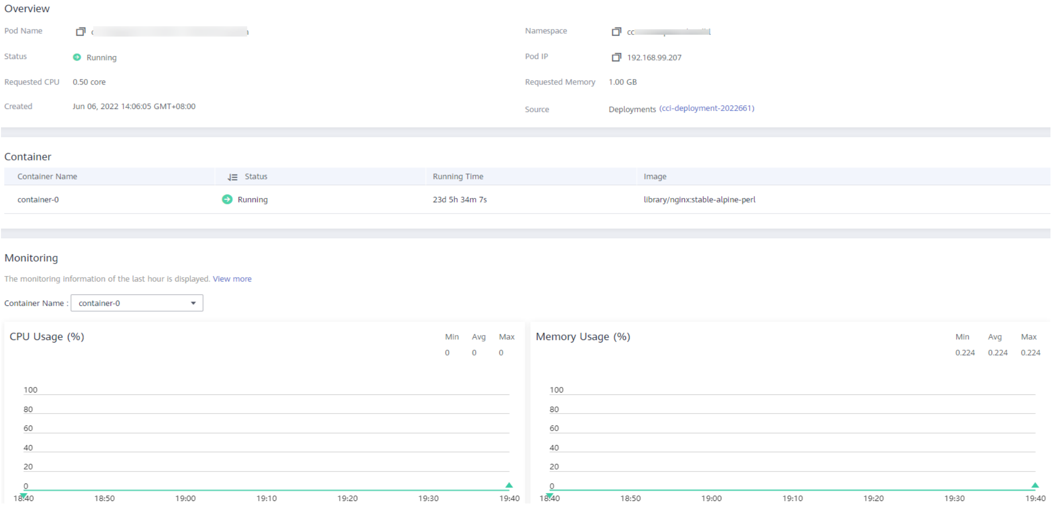
You can create pods by calling the **API** or running the kubectl command. However, these pods cannot be managed on the console because they are not used to deploy a workload or execute a job. To solve this problem, CCI provides pod management, which allows you to filter pods by source.

Figure 4-2 Selecting a pod source



You can view details about all pods, including basic information, container composition, monitoring data, and events. You can use the web-terminal to access pods. In addition, you can view pod logs and delete pods.

Figure 4-3 Pod details



Overview

Pod Name	[redacted]	Namespace	[redacted]
Status	Running	Pod IP	192.168.99.207
Requested CPU	0.50 core	Requested Memory	1.00 GB
Created	Jun 06, 2022 14:06:05 GMT+08:00	Source	Deployments (cci-deployment-2022661)

Container

Container Name	Status	Running Time	Image
container-0	Running	23d 5h 34m 7s	library/nginx:stable-alpine-perl

Monitoring

The monitoring information of the last hour is displayed. View more

Container Name : container-0

Creating a Pod Using kubectl

For details, see **Pod** in the *Developer Guide*.

4.2 Deployments

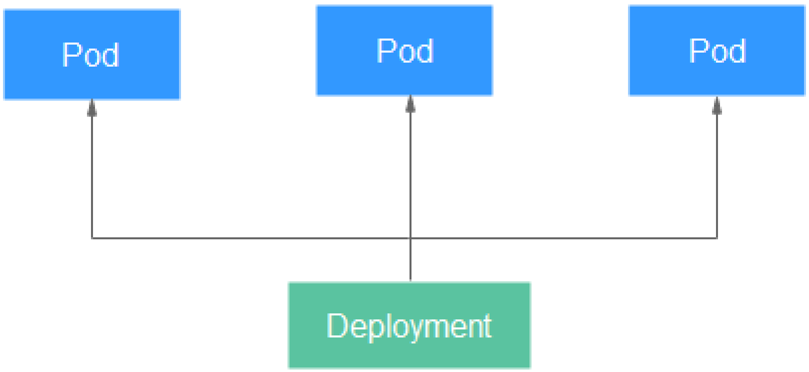
A Deployment is a service-oriented encapsulation of pods. A Deployment may manage one or more pods. These pods have the same role, and requests are routed across the pods. All pods in a Deployment share the same volume.

As described in **Pods**, a pod is the smallest and simplest unit in the Kubernetes object model that you create or deploy. It is designed to be an ephemeral, one-off entity. A pod can be evicted when node resources are insufficient and it automatically disappears when a cluster node fails. Kubernetes provides controllers to manage pods. Controllers can create and manage pods, and provide replica management, rolling upgrade, and self-healing capabilities. The most commonly used controller is Deployment.

A Deployment can contain one or more pod replicas. Each pod replica has the same role. Therefore, the system automatically distributes requests to multiple pod replicas of a Deployment.

A Deployment integrates a lot of functions, including rollout deployment, rolling upgrade, replica creation, and restoration of online jobs. To some extent, you can use Deployments to realize unattended rollout, which greatly reduces operation risks and improves rollout efficiency.

Figure 4-4 Deployment



Creating a Deployment

Step 1 Log in to the CCI console. In the navigation pane, choose **Workloads** > **Deployments**. On the page displayed, click **Create from Image**.

Step 2 Configure basic information.

- **Workload Name**

  Enter 1 to 63 characters starting and ending with a letter or digit. Only lowercase letters, digits, hyphens (-), and periods (.) are allowed. Do not enter consecutive periods or place a hyphen before or after a period. The workload name cannot be changed after creation. If you need to change the name, create another workload.

- **Namespace**

  Select a namespace. If no namespaces are available, create one by following the procedure provided in **Namespace**.

- **Description**

  Enter a description, which cannot exceed 250 characters.

- **Pods**

  Specify the number of pods. A workload can have one or more pods. Each pod consists of one or more containers with the same specifications. Configure multiple pods for a workload if you want higher reliability. If one pod is faulty, the workload can still run properly.

- **Pod Specifications**

  For details about the pod specifications, see "Pod Specifications" in the **Notes and Constraints**.

- **Container Settings**

  A pod generally contains only one container. A pod can also contain multiple containers created from different images. If your application needs to run on multiple containers in a pod, click **Add Container** and then select an image.

NOTICE

If different containers in a pod listen to the same port, a port conflict will occur and the pod may fail to start. For example, if an Nginx container (which listens to port 80) has been added to a pod, a port conflict will occur when another HTTP container in the pod tries to listen to port 80.

- **My Images**: images you have uploaded to SWR

  NOTE

  - If you are an IAM user, you must obtain permissions before you can use the private images of your account. For details on how to set permissions, see **(Optional) Uploading Images**.
  - Currently, CCI does not support third-party image repositories.
  - A single layer of the decompressed image must not exceed 20 GB.
- **Open Source Images**: displays public images in the image center.
- **Shared Images**: images shared by others through SWR

Select the image version and set the container name and CPU and memory specifications (**the minimum configuration of a single container is 0.25 vCPUs and 0.2 GiB**). You can also enable the collection of standard output files. If you enable file collection, you will be billed for the log storage space you use.

NOTE

AOM provides each account with 500-MB log storage space for free each month. You will be billed for any extra space you use on a pay-per-use basis. For details, see **Product Pricing Details**.

You can also configure the following advanced settings for containers:

- **Storage**: You can mount persistent volumes to containers. Currently, Elastic Volume Service (EVS) and SFS Turbo volumes are supported. Click the **EVS Volumes** or **SFS Turbo Volumes** tab, and set the volume name, capacity, container path, and disk type. After the workload is created, you can manage the storage volumes. For details, see **EVS Volumes** or **SFS Turbo Volumes**.
- **Log Collection**: Application logs will be collected in the path you set. You need to configure policies to prevent logs from being over-sized. Click **Add Log Storage**, enter a container path for storing logs, and set the upper limit of log file size. After the workload is created, you can view logs on the AOM console. For details, see **Log Management**.
- **Environment Variables**: You can manually set environment variables or add variable references. Environment variables add flexibility to workload configuration. The environment variables for which you have assigned values during container creation will take effect upon container startup. This saves you the trouble of rebuilding the container image.

  To manually set variables, enter the variable name and value.

  To reference variables, set the variable name, reference type, and referenced value for each variable. The following variables can be referenced: PodIP (pod IP address), PodName (pod name), and Secret. For details about how to create a secret reference, see **Secrets**.

- **Health Check**: Container health can be checked regularly when the container is running. For details about how to configure health checks, see **Setting Health Check Parameters**.
- **Lifecycle**: Lifecycle scripts specify actions that applications take when a lifecycle event occurs. For details about how to configure the scripts, see **Container Lifecycle**.
- **Startup Commands**: You can set the commands to be executed immediately after the container is started. Startup commands correspond to the **ENTRYPOINT** startup instructions of the container engine. For details, see **Setting Container Startup Commands**.
- **Configuration Management**: You can mount ConfigMaps and secrets to a container. For details about how to create ConfigMaps and secrets, see **ConfigMaps** and **Secrets**.

Step 3 Click **Next: Configure Access Settings** to configure access information.

Three options are available:

- **Do not use**: No entry is provided for other workloads to access the current workload. This mode is ideal for scenarios where custom service discovery is used or where access entry is not required.
- **Intranet access**: Configure a domain name or internal domain name/private IP address for the current workload so that other workloads can access the current workload in an internal network. Two internal network access modes are available: Service and ELB. For details about the private network access, see **Private Network Access**.
- **Internet access**: Configure an entry to allow other workloads to access the current workload from the Internet. HTTP, HTTPS, TCP, and UDP are supported. For details about the public network access, see **Public Network Access**.

Step 4 Click **Next: Configure Advanced Settings** and configure advanced settings.

- **Upgrade Policy**: **Rolling upgrade** and **In-place upgrade** are available.
  - **Rolling upgrade**: Gradually replaces an old pod with a new pod. During the upgrade, service traffic is evenly distributed to the old and new pods to ensure service continuity.

    Maximum Number of Unavailable Pods: Maximum number of unavailable pods allowed in a rolling upgrade. If the number is equal to the total number of pods, services may be interrupted. Minimum number of alive pods = Total pods – Maximum number of unavailable pods
  - **In-place upgrade**: Deletes an old pod and then creates a new one. Services will be interrupted during the upgrade.
- **Client DNS Configuration**: You can replace and append domain name resolution configurations. For parameter details, see **Client DNS Configuration**.

Step 5 Click **Next: Confirm**. After you confirm the configuration, click **Submit**. Then click **Back to Deployment List**.

In the workload list, if the workload status is **Running**, the workload is created successfully. You can click the workload name to view workload details and press **F5** to view the real-time workload status.

If you want to access the workload, click the **Access Settings** tab to obtain the access address.

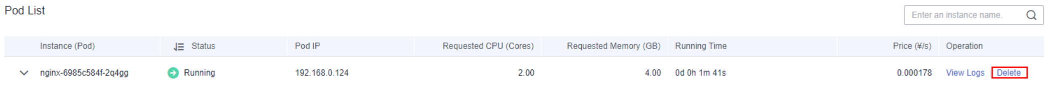
----End

Deleting a Pod

You can manually delete pods. Because pods are controlled by a controller, a new pod will be created immediately after you delete a pod. Manual pod deletion is useful when an upgrade fails halfway or when service processes need to be restarted.

Delete a pod, as shown in **Figure 4-5**.

Figure 4-5 Deleting a pod



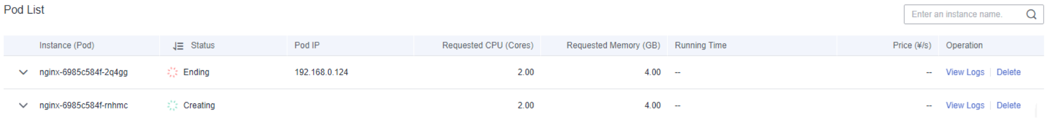
Pod List

Enter an instance name.

Instance (Pod)	Status	Pod IP	Requested CPU (Cores)	Requested Memory (GB)	Running Time	Price (¥/s)	Operation
nginx-6985c584f-2q4gg	Running	192.168.0.124	2.00	4.00	0d 0h 1m 41s	0.000178	View Logs Delete

A new pod is created immediately after you delete the pod, as shown in **Figure 4-6**.

Figure 4-6 Result of deleting a pod



Pod List

Enter an instance name.

Instance (Pod)	Status	Pod IP	Requested CPU (Cores)	Requested Memory (GB)	Running Time	Price (¥/s)	Operation
nginx-6985c584f-2q4gg	Ending	192.168.0.124	2.00	4.00	--	--	View Logs | Delete
nginx-6985c584f-rnhmc	Creating		2.00	4.00	--	--	View Logs | Delete

Creating a Deployment Using kubectl

For details, see **Deployment**.

Troubleshooting a Failure to Pull the Image

If there is an event indicating that the image fails to be pulled on the workload details page, locate the fault by following the procedure provided in **What Do I Do If an Event Indicating That the Image Failed to Be Pulled Occurs?**

Troubleshooting a Failure to Restart the Container

If there is an event indicating that the container fails to be restarted on the workload details page, locate the fault by following the procedure provided in **What Do I Do If an Event Indicating That the Container Failed to Be Restarted Occurs?**

4.3 Jobs

A job is responsible for batch processing of short lived one-off tasks that are executed only once. It ensures that one or more pods are successfully completed.

A job is a resource object that Kubernetes uses to control batch tasks. Batch jobs are different from long-term servo jobs (such as Deployments). The former can be started and terminated at specific time, while the latter runs unceasingly until it is terminated. The pods managed by a job will be automatically removed after successfully completing tasks based on user-defined configurations.

This run-to-completion feature of jobs is especially suitable for one-off tasks, such as continuous integration (CI). It works with the per-second billing of CCI to implement pay-per-use in the real sense.

Creating a Job

Step 1 Log in to the CCI console. In the navigation pane, choose **Workloads** > **Jobs**. On the page displayed, click **Create from Image**.

Step 2 Configure basic information.

- **Job Name**

  Enter 1 to 63 characters starting and ending with a letter or digit. Only lowercase letters, digits, hyphens (-), and periods (.) are allowed. Do not enter consecutive periods or place a hyphen before or after a period. The job name cannot be changed after creation. If you need to change the name, create another job.

- **Namespace**

  Select a namespace. If no namespaces are available, create one by following the procedure provided in **Namespace**.

- **Description**

  Enter a description, which cannot exceed 250 characters.

- **Pod Specifications**

  You can select **GPU-accelerated** and allocate GPUs to the workload only if the namespace is a GPU-accelerated namespace.

  For details about the pod specifications, see "Pod Specifications" in the **Notes and Constraints**.

- **Container Settings**

  A pod generally contains only one container. A pod can also contain multiple containers created from different images. If your application needs to run on multiple containers in a pod, click **Add Container** and then select an image.

  NOTICE

  If different containers in a pod listen to the same port, a port conflict will occur and the pod may fail to start. For example, if an Nginx container (which listens to port 80) has been added to a pod, a port conflict will occur when another HTTP container in the pod tries to listen to port 80.

- **My Images**: images you have uploaded to SWR

  NOTE

  > If you are an IAM user, you must obtain permissions before you can use the private images of your account. For details on how to set permissions, see **(Optional) Uploading Images**.
  >
  > Currently, CCI does not support third-party image repositories.

- **Open Source Images**: displays public images in the image center.
- **Shared Images**: images shared by others through SWR

Select the image version and set the container name and CPU and memory specifications (**the minimum configuration of a single container is 0.25 vCPUs and 0.2 GiB**). You can also enable the collection of standard output files. If you enable file collection, you will be billed for the log storage space you use.

You can also configure the following advanced settings for containers:

- **Storage**: You can mount persistent volumes to containers. Currently, EVS and SFS Turbo volumes are supported. On the **EVS Volumes** or **SFS Turbo Volumes** tab, set the volume name, capacity, container path, and disk type. After the job is created, you can manage the storage volumes. For details, see **EVS Volumes**, or **SFS Turbo Volumes**.
- **Log Collection**: Application logs will be collected in the path you set. You need to configure policies to prevent logs from being over-sized. Click **Add Log Storage**, enter a container path for storing logs, and set the upper limit of log file size. After the workload is created, you can view logs on the AOM console. For details, see **Log Management**.
- **Environment Variables**: You can manually set environment variables or add variable references. Environment variables add flexibility to workload configuration. The environment variables for which you have assigned values during container creation will take effect upon container startup. This saves you the trouble of rebuilding the container image.

  To manually set variables, enter the variable name and value.

  To reference variables, set the variable name, reference type, and referenced value for each variable. The following variables can be referenced: PodIP (pod IP address), PodName (pod name), and Secret. For details about how to create a secret reference, see **Secrets**.
- **Liveness Probe**: You can configure a liveness probe for customized health check of the container. If the container fails the check, the CCI will stop the container and determine whether to restart the container based on the restart policy. For details about how to configure a liveness probe, see **Setting Health Check Parameters**.
- **Lifecycle**: Lifecycle scripts specify actions that applications take when a lifecycle event occurs. For details about how to configure the scripts, see **Container Lifecycle**.
- **Startup Commands**: You can set the commands to be executed immediately after the container is started. Startup commands correspond to the **ENTRYPOINT** startup instructions of the container engine. For details, see **Setting Container Startup Commands**.
- **Configuration Management**: You can mount ConfigMaps and secrets to a container. For details about how to create ConfigMaps and secrets, see **ConfigMaps** and **Secrets**.

Step 3 Click **Next: Configure Advanced Settings** and configure advanced settings.

Jobs can be classified into one-off jobs and custom jobs.

- One-off job: A one-off job creates one pod each time. The job is completed when the pod is successfully executed.
- Custom job: You can set the number of executions and the number of concurrent executions for a custom job. **Completions** specifies the number of pods that need to be successfully executed until the job is completed. **Parallelism** specifies the maximum number of pods that can run concurrently during the execution of the job. The number of parallel jobs should be less than the number of times the job is executed.

You can set the timeout period for the job. When the job execution duration exceeds the timeout period, the job will be identified as failed, and all pods under this job will be deleted. If you leave this parameter blank, the job will never time out.

Step 4 Click **Next: Confirm**. After you confirm the configuration, click **Submit**. Then click **Back to Job List**.

If the job status is **Executing**, the job is created successfully. You can click the job name to view job details and press **F5** to view the real-time job status.

----End

Creating a Job Using kubectl

For details, see **Creating a Job**.

4.4 Cron Jobs

A cron job runs a job on a specified schedule. A cron job object is similar to a line of a crontab file in Linux.

Creating a Cron Job

Step 1 Log in to the CCI console. In the navigation pane, choose **Workloads** > **Cron Jobs**. On the page displayed, click **Create from Image**.

Step 2 Configure basic information.

- **Job Name**

  Enter 1 to 52 characters starting and ending with a letter or digit. Only lowercase letters, digits, hyphens (-), and periods (.) are allowed. Do not enter consecutive periods or place a hyphen before or after a period. The job name cannot be changed after creation. If you need to change the name, create another job.
- **Namespace**

  Select a namespace. If no namespaces are available, create one by following the procedure provided in **Namespace**.
- **Description**

  Enter a description, which cannot exceed 250 characters.

- **Pod Specifications**

  You can select **GPU-accelerated** and allocate GPUs to the workload only if the namespace is a GPU-accelerated namespace.

  For details about the pod specifications, see "Pod Specifications" in the **Notes and Constraints**.

- **Container Settings**

  A pod generally contains only one container. A pod can also contain multiple containers created from different images. If your application needs to run on multiple containers in a pod, click **Add Container** and then select an image.

  NOTICE

  If different containers in a pod listen to the same port, a port conflict will occur and the pod may fail to start. For example, if an Nginx container (which listens to port 80) has been added to a pod, a port conflict will occur when another HTTP container in the pod tries to listen to port 80.

  - **My Images**: images you have uploaded to SWR

    NOTE

    If you are an IAM user, you must obtain permissions before you can use the private images of your account. For details on how to set permissions, see **(Optional) Uploading Images**.

    Currently, CCI does not support third-party image repositories.

  - **Open Source Images**: displays public images in the image center.
  - **Shared Images**: images shared by others through SWR

  Select the image version and set the container name and CPU and memory specifications (**the minimum configuration of a single container is 0.25 vCPUs and 0.2 GiB**). You can also enable the collection of standard output files. If you enable file collection, you will be billed for the log storage space you use.

  You can also configure the following advanced settings for containers:

  - **Log Collection**: Application logs will be collected in the path you set. You need to configure policies to prevent logs from being over-sized. Click **Add Log Storage**, enter a container path for storing logs, and set the upper limit of log file size. After the workload is created, you can view logs on the AOM console. For details, see **Log Management**.
  - **Environment Variables**: You can manually set environment variables or add variable references. Environment variables add flexibility to workload configuration. The environment variables for which you have assigned values during container creation will take effect upon container startup. This saves you the trouble of rebuilding the container image.

    To manually set variables, enter the variable name and value.

    To reference variables, set the variable name, reference type, and referenced value for each variable. The following variables can be referenced: PodIP (pod IP address), PodName (pod name), and Secret. For details about how to create a secret reference, see **Secrets**.

  - **Liveness Probe**: You can configure a liveness probe for customized health check of the container. If the container fails the check, the CCI will stop the container and determine whether to restart the container based on the restart policy. For details about how to configure a liveness probe, see **Setting Health Check Parameters**.
  - **Lifecycle**: Lifecycle scripts specify actions that applications take when a lifecycle event occurs. For details about how to configure the scripts, see **Container Lifecycle**.
  - **Startup Commands**: You can set the commands to be executed immediately after the container is started. Startup commands correspond to the **ENTRYPOINT** startup instructions of the container engine. For details, see **Setting Container Startup Commands**.
  - **Configuration Management**: You can mount ConfigMaps and secrets to a container. For details about how to create ConfigMaps and secrets, see **ConfigMaps** and **Secrets**.

Step 3 Click **Next: Configure Timing Rule** and configure advanced settings.

- **Concurrency Policy**
  - **Forbid**: A new job cannot be created until the previous job is completed.
  - **Allow**: New jobs can be created continuously.
  - **Replace**: A new job replaces the previous job when it is time to create a job even if the previous job has not been completed.
- **Schedule**: Set the schedule according to which the job is executed.
- **Job Record**: Set the number of records to be retained for successful jobs and failed jobs.

Step 4 Click **Next: Confirm**. After you confirm the configuration, click **Submit**. Then click **Back to Cron Job List**.

If the job status is **Started**, the cron job is created successfully. You can click the job name to view job details and press **F5** to view the real-time job status.

----End

Creating a Cron Job Using kubectl

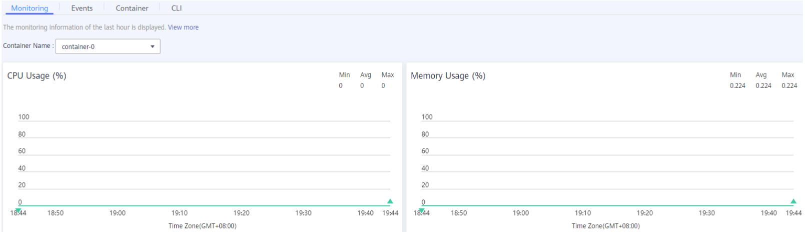
For details, see **Creating a Cron Job**.

4.5 Viewing Resource Usage

After you have created a workload, you may want to know the resource usage rates of each pod.

CCI allows you to monitor the CPU or GPU usage and memory usage of each pod. Go to the details page of a Deployment, job, or Cron job, and click the arrow icon of a pod in the pod list. On the **Monitoring** tab page, view the resource usage, as shown in **Figure 4-7**. You can also view the resource usage of each pod by choosing **Workloads** > **Pods** in the navigation pane.

Figure 4-7 Viewing monitoring data



The monitoring information of the last hour is displayed. View more

Container Name : container-0

4.6 Setting Container Startup Commands

Starting the container is to start the main process. However, some preparations must be made before the main process is started. For example, you configure or initialize MySQL databases before running MySQL servers. You can set **ENTRYPOINT** or **CMD** in the Dockerfile when you create an image. As shown in the following, the **ENTRYPOINT ["top", "-b"]** command is set in the Dockerfile. This command will be executed during container startup.

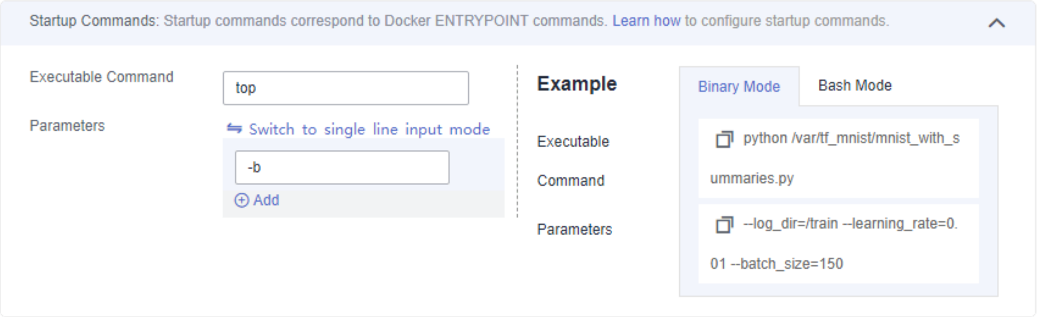
```
FROM ubuntu
ENTRYPOINT ["top", "-b"]
```

NOTICE

The startup command must be supported by the container image. Otherwise, the container startup will fail.

In CCI, you can also set the container startup command. For example, to add the preceding command in the Dockerfile, you can click **Add** and enter the **top** command, and then click **Add** again and enter **-b** in the **Advanced Settings** area when you create a workload, as shown in the following figure.

Figure 4-8 Startup command



Startup Commands: Startup commands correspond to Docker ENTRYPOINT commands. Learn how to configure startup commands.

Executable Command: top

Parameters: -b

Example

Executable Command: python /var/tf_mnist/mnist_with_summaries.py

Parameters: --log_dir=/train --learning_rate=0.01 --batch_size=150

When the container engine runs, only one **ENTRYPOINT** command is supported. The startup command that you set in CCI will overwrite the **ENTRYPOINT** and **CMD** commands that you set in the Dockerfile when you created the image. The following table lists the rules.

Image Entrypoint	Image CMD	Command for Running a Container	Parameter for Running a Container	Command Executed
[touch]	[/root/test]	Not set	Not set	[touch /root/test]
[touch]	[/root/test]	[mkdir]	Not set	[mkdir]
[touch]	[/root/test]	Not set	[/opt/test]	[touch /opt/test]
[touch]	[/root/test]	[mkdir]	[/opt/test]	[mkdir /opt/test]

4.7 Container Lifecycle

Setting Container Lifecycle Parameters

Based on Kubernetes, CCI provides containers with **lifecycle hooks**, which enable containers to run code triggered by events during their management lifecycle. For example, if you want a container to perform a certain operation before it is stopped, you can register a hook. The following lifecycle hooks are provided:

- Post-Start Processing: triggered immediately after the container is started
- Pre-Stop Processing: triggered immediately before the container is stopped

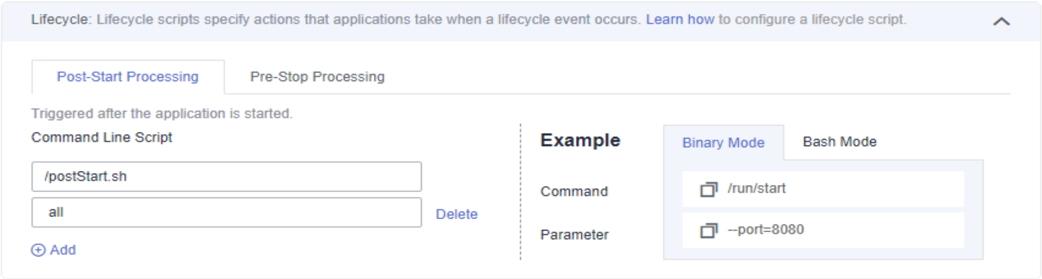
NOTE

Currently, CCI supports only hook handlers of the Exec type, which execute a specific command.

During workload creation, expand **Advanced Settings** and click the **Post-Start Processing** or **Pre-Stop Processing** tab in the **Lifecycle** area.

For example, if you want to run the **/postStart.sh all** command in the container, configure data on the page as shown in the following figure. The first row indicates the script name and the second row indicates a parameter setting.

Figure 4-9 Command settings



Lifecycle: Lifecycle scripts specify actions that applications take when a lifecycle event occurs. Learn how to configure a lifecycle script.

Triggered after the application is started.

Command Line Script

/postStart.sh

all

Example

Command /run/start

Parameter --port=8080

Setting Container Lifecycle Parameters Using kubectl

For details, see **Lifecycle Management**.

4.8 Setting Health Check Parameters

Container health can be checked regularly when the container is running.

CCI provides two health check methods based on Kubernetes:

- Liveness probe: checks whether a containerized application is alive. The liveness probe is similar to the **ps** command for checking whether a process is running. If the containerized application fails the check, the container will be restarted. If the containerized application passes the check, no operation will be performed.
- Readiness probe: checks whether a containerized application is ready to handle requests. An application may take a long time to start up and provide services, for example, because it needs to load disk data or wait for the startup of an external module. In this case, application processes are running, but the application is not ready to provide services. This is where the readiness probe is useful.

Health Check Modes

- **HTTP Request Mode**

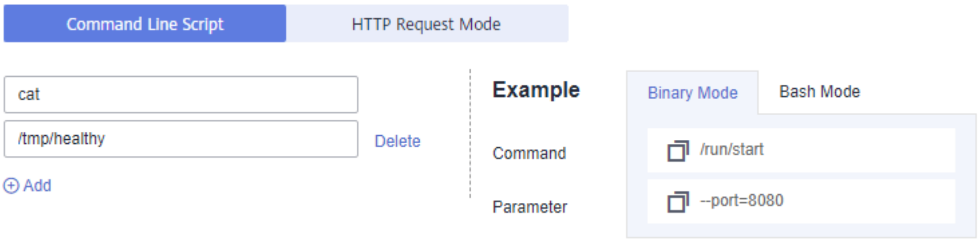
  The probe sends an HTTP GET request to the container. If the probe receives a 2xx or 3xx status code, the container is healthy.

- **Command Line Script**

  The probe runs a command in the container and checks the exit status code. If the exit status code is 0, the probe is healthy.

  For example, if you want to run the **cat /tmp/healthy** command to check whether the **/tmp/healthy** directory exists, configure data as shown in the following figure.

Figure 4-10 Command setting



cat

/tmp/healthy

Command /run/start

Parameter --port=8080

Common Parameter Description

Table 4-1 Health check parameters

Parameter	Description
Time Window (s)	Delay time (unit: second). For example, if you set this parameter to **10**, the probe starts 10 seconds after the container is started.

Parameter	Description
Timeout Period (s)	Timeout period (unit: second). For example, if you set this parameter to **10**, the container must return a response in 10 seconds. Otherwise, the probe is counted as failed. If you set this parameter to **0** or do not specify any value, the default value (1 second) is used.

Setting Health Check Using kubectl

- For details about how to set the liveness probe, see **Liveness Probe**.
- For details about how to set the readiness probe, see **Readiness Probe**.

4.9 Web-Terminal

The web-terminal provides the container connection function to help you quickly debug the container.

Constraints and Restrictions

- The web-terminal logs in to the container by using sh shell by default. Therefore, the container must support sh shell.
- You can log in only to running containers by using the web-terminal.
- You need to enter **exit** in the web-terminal during exit; otherwise, the sh process will not be terminated.

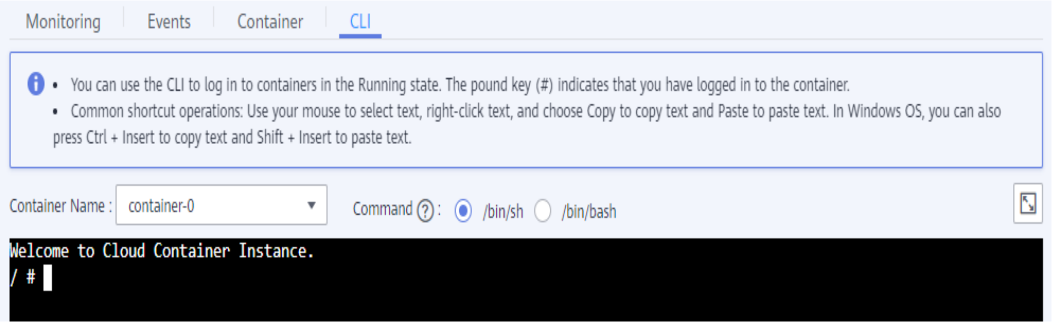
Connecting to the Container by Using the Web-Terminal

Step 1 Log in to the CCI console. In the navigation pane, choose **Workloads** > **Deployments**. On the page displayed, click the target workload.

Step 2 In the **Pod List** area of the workload details page, click the arrow icon at the left of the pod and then click the **CLI** tab.

If **#** is displayed, you have logged in to the container.

Figure 4-11 Container CLI



- You can use the CLI to log in to containers in the Running state. The pound key (#) indicates that you have logged in to the container.
- Common shortcut operations: Use your mouse to select text, right-click text, and choose Copy to copy text and Paste to paste text. In Windows OS, you can also press Ctrl + Insert to copy text and Shift + Insert to paste text.

Container Name : container-0

Command : ● /bin/sh ○ /bin/bash

```
Welcome to Cloud Container Instance.
/ #
```

----End

4.10 Upgrading a Workload

You can upgrade a workload after you create it. There are two ways to upgrade a workload.

- **Rolling upgrade**: Gradually replaces an old pod with a new pod. During the upgrade, service traffic is evenly distributed to the old and new pods to ensure service continuity.
- **In-place upgrade**: Deletes an old pod and then creates a new one. Services will be interrupted during the upgrade.

Upgrading a Workload

Step 1 Log in to the CCI console. In the navigation pane, choose **Workloads** > **Deployments**. On the page displayed, click the target workload. Then, click **Upgrade** in the upper right corner of the workload details page.

Step 2 Modify pod specifications.

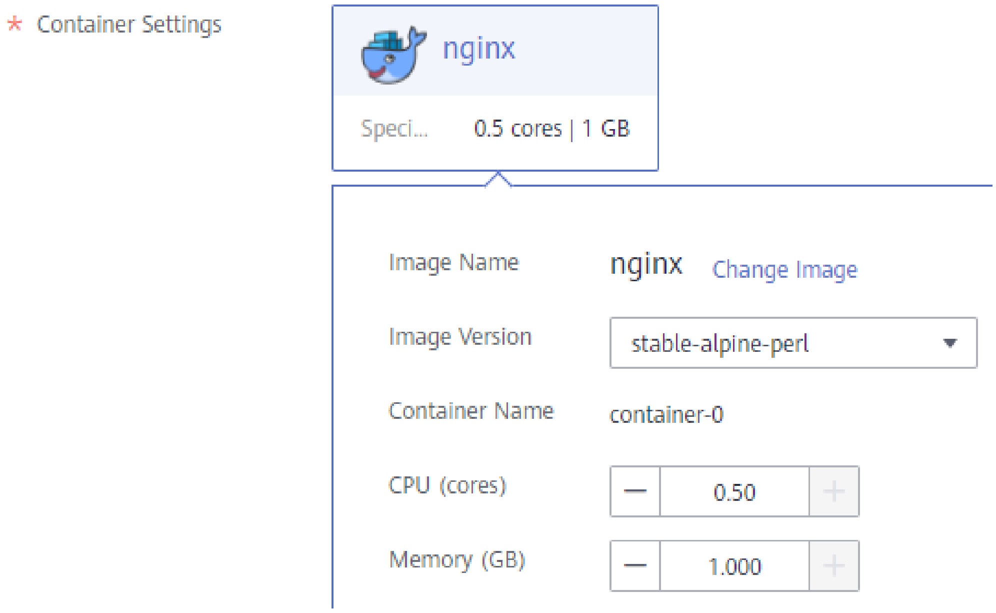
You can select **GPU-accelerated** and allocate GPUs to the workload only if the namespace is a GPU-accelerated namespace.

For details about the pod specifications, see "Pod Specifications" in the **Notes and Constraints**.

Step 3 Modify container settings.

1. Click **Change Image** to select a new image.

Figure 4-12 Changing the image



* Container Settings

nginx

Speci... 0.5 cores | 1 GB

Image Name nginx

Image Version stable-alpine-perl

Container Name container-0

CPU (cores) 0.50

Memory (GB) 1.000

   - **My Images**: images you have uploaded to SWR
   - **Open Source Images**: displays public images in the image center.

   - **Shared Images**: images shared by others through SWR

2. Select the image version and set the container name and CPU and memory specifications (the minimum configuration of a single container is 0.25 vCPUs and 0.2 GiB). You can also enable the collection of standard output files. If you enable file collection, you will be billed for the log storage space you use.

   NOTE

   AOM provides each account with 500-MB log storage space for free each month. You will be billed for any extra space you use on a pay-per-use basis. For details, see **Product Pricing Details**.

   You can also configure the following advanced settings for containers:

   - **Storage**: You can mount persistent volumes to containers. Currently, EVS and SFS Turbo volumes are supported. Click the **EVS Volumes** or **SFS Turbo Volumes** tab, and set the volume name, capacity, container path, and disk type. After the workload is created, you can manage the storage volumes. For details, see **EVS Volumes** or **SFS Turbo Volumes**.
   - **Log Collection**: Application logs will be collected in the path you set. You need to configure policies to prevent logs from being over-sized. Click **Add Log Storage**, enter a container path for storing logs, and set the upper limit of log file size. After the workload is created, you can view logs on the AOM console. For details, see **Log Management**.
   - **Environment Variables**: You can manually set environment variables or add variable references. Environment variables add flexibility to workload configuration. The environment variables for which you have assigned values during container creation will take effect upon container startup. This saves you the trouble of rebuilding the container image.

     To manually set variables, enter the variable name and value.

     To reference variables, set the variable name, reference type, and referenced value for each variable. The following variables can be referenced: PodIP (pod IP address), PodName (pod name), and Secret. For details about how to create a secret reference, see **Secrets**.
   - **Health Check**: Container health can be checked regularly when the container is running. For details about how to configure health checks, see **Setting Health Check Parameters**.
   - **Lifecycle**: Lifecycle scripts specify actions that applications take when a lifecycle event occurs. For details about how to configure the scripts, see **Container Lifecycle**.
   - **Startup Commands**: You can set the commands to be executed immediately after the container is started. Startup commands correspond to the **ENTRYPOINT** startup instructions of the container engine. For details, see **Setting Container Startup Commands**.
   - **Configuration Management**: You can mount ConfigMaps and secrets to a container. For details about how to create ConfigMaps and secrets, see **ConfigMaps** and **Secrets**.

Step 4 Click **Next** and select an upgrade policy.

Two options are available: **Rolling upgrade** and **In-place upgrade**.

- **Rolling upgrade**: Gradually replaces an old pod with a new pod. During the upgrade, service traffic is evenly distributed to the old and new pods to ensure service continuity.

Maximum Number of Unavailable Pods: Maximum number of unavailable pods allowed in a rolling upgrade. If the number is equal to the total number of pods, services may be interrupted. Minimum number of alive pods = Total pods – Maximum number of unavailable pods

- **In-place upgrade**: Deletes an old pod and then creates a new one. Services will be interrupted during the upgrade.

Step 5 Click **Next** and then **Submit**.

----End

Upgrading a Workload Using kubectl

For details about how to use kubectl to upgrade a workload, see **Upgrade** in **Deployment**.

4.11 Scaling a Workload

This section describes two workload scaling methods: auto scaling and manual scaling.

- Auto scaling: Supports metric-based, scheduled, and periodic policies. When the configuration is complete, pods can be automatically added or deleted based on resource changes or a specified schedule.
- Manual scaling: Increases or decreases the number of pods immediately after the configuration is complete.

NOTICE

If a pod mounted with an EVS volume is deleted, the EVS disk will not be deleted. If a new pod with the same name is created, the new pod cannot be mounted to any EVS volume.

Auto Scaling

NOTE

Currently, auto scaling is supported only for Deployments.

An appropriate auto scaling policy eliminates the need to manually adjust resources in response to service changes and traffic peaks, thus helping you reduce the workforce and resources required. CCI supports the following types of auto scaling policies:

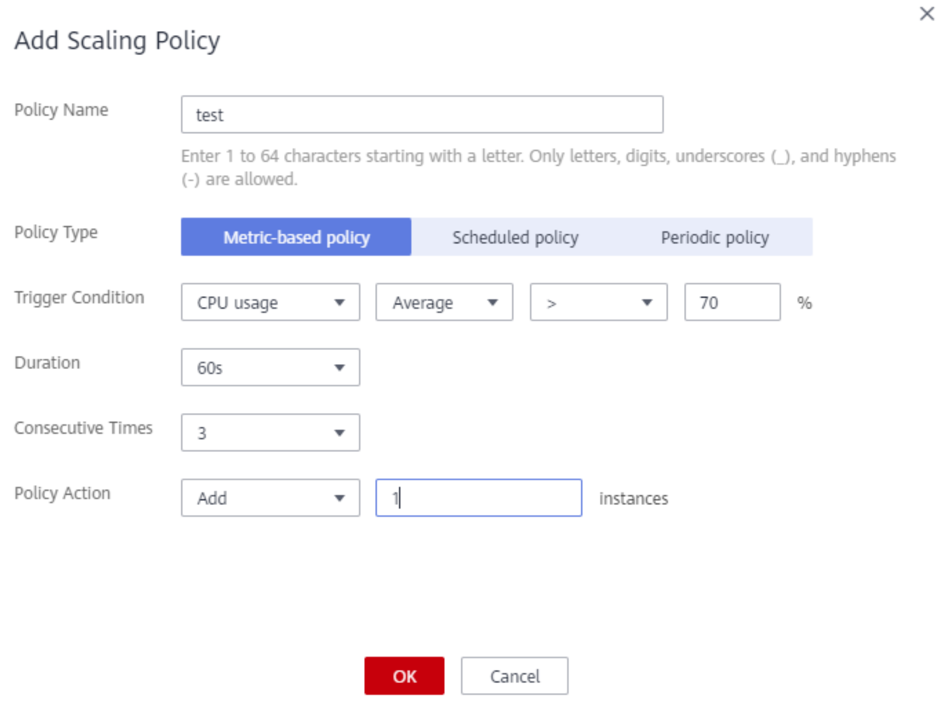
Metric-based: Scales the workload based on CPU/memory usage. You can specify a CPU/memory usage threshold. If the usage is higher or lower than the threshold, instances can be automatically added or deleted.

Scheduled: Scales the workload at a specified time. A scheduled scaling policy is ideal for scenarios such as flash sales and anniversary promotions.

Periodic: Scales the workload on a daily, weekly, or monthly basis. A periodic scaling policy is ideal for applications that have periodic traffic changes.

- Configure a metric-based auto scaling policy.
  a. Log in to the CCI console. In the navigation pane, choose **Workloads** > **Deployments**. On the page displayed, click the target Deployment.
  b. In the **Scaling** area of the Deployment details page, click **Auto Scaling** and then click **Add Scaling Policy**.

Figure 4-13 Adding a metric-based auto scaling policy



Add Scaling Policy

Policy Name: test

Enter 1 to 64 characters starting with a letter. Only letters, digits, underscores (_), and hyphens (-) are allowed.

Policy Type: Metric-based policy

Trigger Condition: CPU usage, Average, >, 70 %

Duration: 60s

Consecutive Times: 3

Policy Action: Add 1 instances

Table 4-2 Parameters of a metric-based auto scaling policy

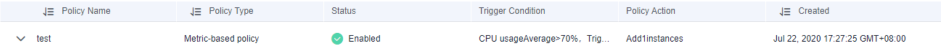
Parameter	Description
Policy Name	Enter a name for the policy.
Policy Type	Select **Metric-based policy**.
Trigger Condition	Select **CPU usage** or **Memory usage**. If you set the trigger condition to **average memory usage > 70%**, the scaling policy will be triggered when the average memory usage exceeds 70%.
Duration	Statistical period. Select a value from the drop-down list. If you set this parameter to **60**, metric statistics will be collected every 60 seconds.
Consecutive Times	If you set this parameter to **3**, the configured action will be triggered when the threshold is reached for 3 consecutive statistical periods.

Parameter	Description
Policy Action	Specify the action to be executed when the policy is triggered. You can specify an action to add or reduce the number of instances.

c. Click **OK**.

The policy is added to the policy list, and its status is **Enabled**.

Figure 4-14 Policy enabled



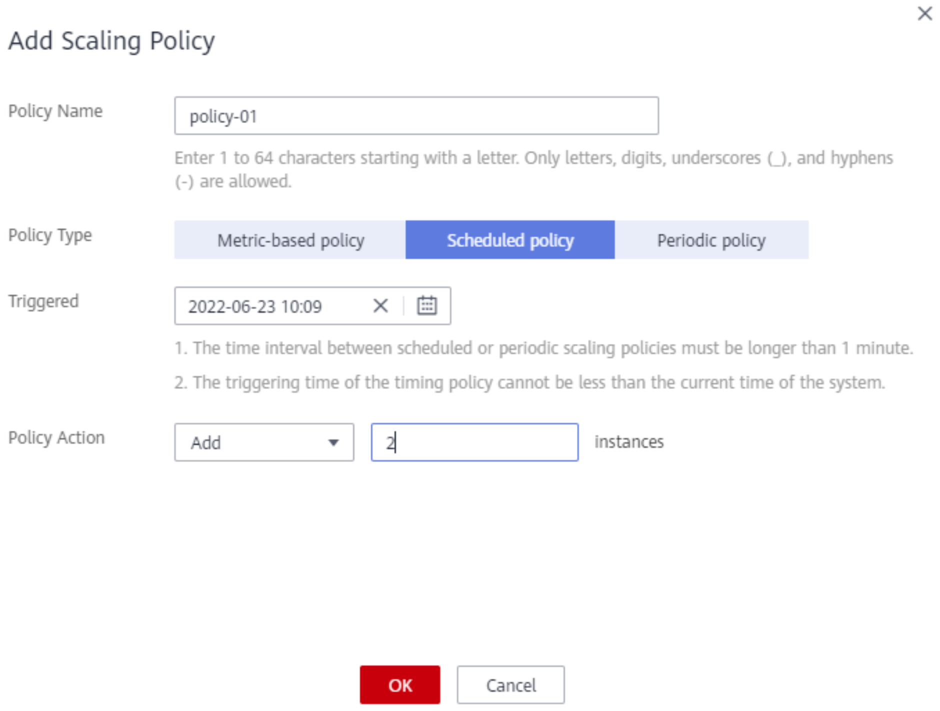
	Policy Name	Policy Type	Status	Trigger Condition	Policy Action	Created
∨	test	Metric-based policy	✓ Enabled	CPU usageAverage>70%，Trig...	Add1instances	Jul 22, 2020 17:27:25 GMT+08:00

When the trigger condition is met, the auto scaling policy will be executed.

- Configure a scheduled auto scaling policy.

a. In the **Scaling** area, click **Auto Scaling** and then click **Add Scaling Policy**.

Figure 4-15 Adding a scheduled auto scaling policy



Add Scaling Policy

Policy Name: policy-01

Enter 1 to 64 characters starting with a letter. Only letters, digits, underscores (_), and hyphens (-) are allowed.

Policy Type: Metric-based policy | Scheduled policy | Periodic policy

Triggered: 2022-06-23 10:09

1. The time interval between scheduled or periodic scaling policies must be longer than 1 minute.

2. The triggering time of the timing policy cannot be less than the current time of the system.

Policy Action: Add 2 instances

Table 4-3 Parameters of a scheduled auto scaling policy

Parameter	Description
Policy Name	Enter a name for the policy.
Policy Type	Select **Scheduled Policy**.

Parameter	Description
Triggered	Specify the time when you want the policy to be triggered.
Policy Action	Specify the action to be executed when the policy is triggered. You can specify an action to add or reduce the number of instances.

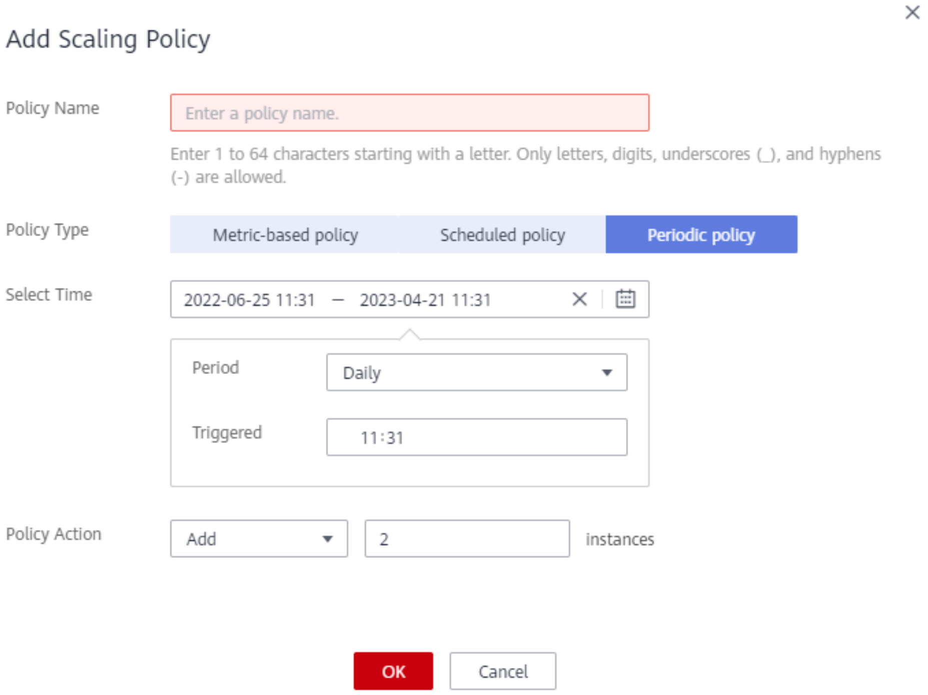
b. Click **OK**.

The policy is added to the policy list, and its status is **Enabled**.

- Configure a periodic auto scaling policy.

a. In the **Scaling** area, click **Auto Scaling** and then click **Add Scaling Policy**.

Figure 4-16 Adding a periodic auto scaling policy



Add Scaling Policy

Policy Name: Enter a policy name.

Enter 1 to 64 characters starting with a letter. Only letters, digits, underscores (_), and hyphens (-) are allowed.

Policy Type: Periodic policy

Select Time: 2022-06-25 11:31 — 2023-04-21 11:31

Period: Daily

Triggered: 11:31

Policy Action: Add 2 instances

Table 4-4 Parameters of a periodic auto scaling policy

Parameter	Description
Policy Name	Enter a name for the policy.
Policy Type	Select **Periodic Policy**.
Select Time	Select the time range, frequency, and time when you want the policy to be triggered.
Policy Action	Specify the action to be executed when the policy is triggered.

b. Click **OK**.

The policy is added to the policy list, and its status is **Enabled**.

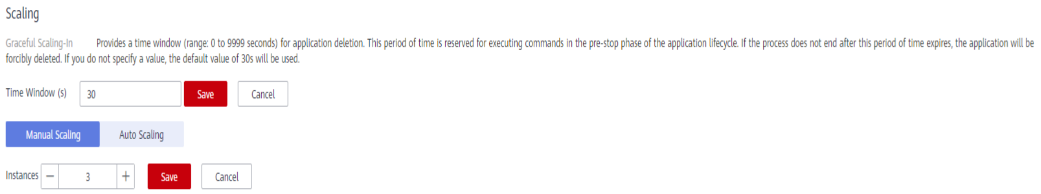
Manual Scaling

Step 1 Log in to the CCI console. In the navigation pane, choose **Workloads** > **Deployments**. On the page displayed, click the target Deployment.

Step 2 Under **Manual Scaling** in the **Scaling** area, click ✎ and modify the number of instances (for example, change the value to **3**), and then click **Save**. The scaling takes effect immediately.

CCI provides a time window for running pre-stop processing commands before an application is deleted. If a command process is still running when the time window expires, the application will be forcibly deleted.

Figure 4-17 Changing the number of instances



Scaling

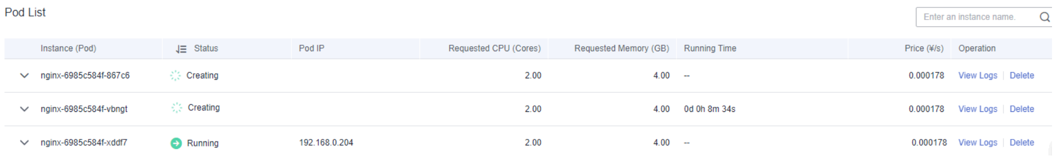
Graceful Scaling-In Provides a time window (range: 0 to 9999 seconds) for application deletion. This period of time is reserved for executing commands in the pre-stop phase of the application lifecycle. If the process does not end after this period of time expires, the application will be forcibly deleted. If you do not specify a value, the default value of 30s will be used.

Time Window (s) 30

Instances − 3 +

Step 3 In the pod list, you can see the new pods being created. When the status of all added pods changes to **Running**, the scaling is completed successfully.

Figure 4-18 Pod list after a manual scaling



Pod List

Instance (Pod)	Status	Pod IP	Requested CPU (Cores)	Requested Memory (GB)	Running Time	Price (¥/s)	Operation
nginx-6985c584f-867c6	Creating		2.00	4.00	--	0.000178	View Logs | Delete
nginx-6985c584f-vbngt	Creating		2.00	4.00	0d 0h 8m 34s	0.000178	View Logs | Delete
nginx-6985c584f-xddf7	Running	192.168.0.204	2.00	4.00	--	0.000178	View Logs | Delete

----End

4.12 Client DNS Configuration

CCI uses **dnsPolicy** to identify different DNS policies for each pod. The value of **dnsPolicy** can be either of the following:

- **None**: DNS settings from CCI are ignored. When using this policy, you can customize the DNS configuration by defining **dnsConfig**.
- **Default**: Private DNS is used to resolve the domain names of other cloud services and forward domain name requests to public DNS servers. For details, see **What Are the Private DNS Servers Provided by the Huawei Cloud DNS Service?**

  NOTE

  This policy can be used only when the coredns add-on is not installed in the namespace where the pod is located. If coredns has been installed, the Default policy will be overwritten by the ClusterFirst policy.

- **ClusterFirst**: coredns installed in the namespace resolves the domain names. Any DNS query that does not match the configured cluster domain suffix (.cluster.local), such as **www.kubernetes.io**, is forwarded to the upstream DNS server (private DNS by default).

  For details about how to configure the stub domain and upstream DNS server, see **Add-on Management**.

  NOTE

  This policy can be used only when the coredns add-on has been installed in the namespace where the pod is located. If coredns is not installed, the ClusterFirst policy will be overwritten by the Default policy.

If **dnsPolicy** is not specified, its default value is set based on whether the coredns add-on is installed. If coredns is installed, **ClusterFirst** is used by default. If coredns is not installed, **Default** is used.

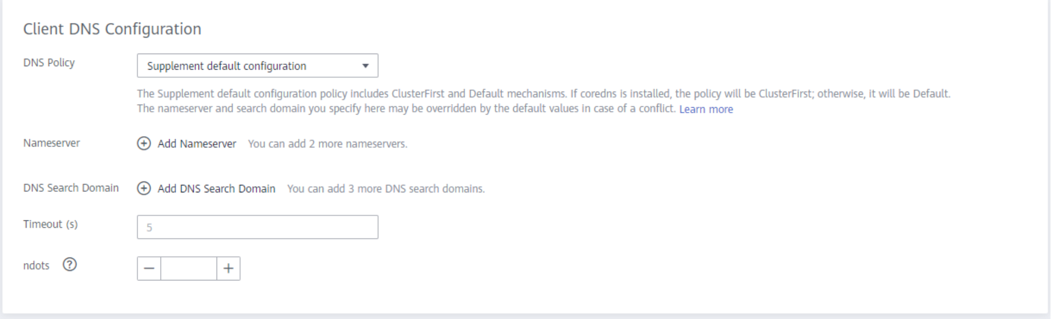
dnsConfig description:

dnsConfig specifies DNS parameters for applications. The DNS parameter settings will be merged into the DNS configuration file generated based on **dnsPolicy**. If **dnsPolicy** is set to **None**, the DNS configuration specified by **dnsConfig** will overwrite the content in the DNS configuration file. If **dnsPolicy** is not set to **None**, the DNS parameters specified by **dnsConfig** will be supplemented to the DNS configuration file.

- **nameservers**: a list of IP addresses that will be used as DNS servers for the pod. If **dnsPolicy** is set to **None** for a pod, the list must contain at least one IP address; otherwise, this property is optional. The servers listed will be merged into the nameservers generated from the chosen DNS policy in **dnsPolicy** with duplicate addresses removed.
- **searches**: a list of DNS search domains for hostname lookup in the pod. This property is optional. When specified, the provided list will be merged into the search domain names generated from the chosen DNS policy in **dnsPolicy**. Duplicate domain names are removed. Kubernetes allows for at most 6 search domains.
- **options**: a list of objects where each object may have a name property (required) and a value property (optional). The content in this property will be merged into the options generated from the chosen DNS policy in **dnsPolicy**. Common options include timeout, attempts, and ndots.

Configuring DNS Policies During Workload Creation on the CCI Console

Figure 4-19 Client DNS Configuration



Client DNS Configuration

DNS Policy: Supplement default configuration

The Supplement default configuration policy includes ClusterFirst and Default mechanisms. If coredns is installed, the policy will be ClusterFirst; otherwise, it will be Default. The nameserver and search domain you specify here may be overridden by the default values in case of a conflict. Learn more

Nameserver: Add Nameserver You can add 2 more nameservers.

DNS Search Domain: Add DNS Search Domain You can add 3 more DNS search domains.

Timeout (s): 5

ndots

- **Replace default configuration**: corresponds to the **None** policy. The nameserver, search domain, timeout, and ndots you specify here will take effect.
- **Supplement default configuration**: includes **ClusterFirst** and **Default** policies. The final value depends on whether the coredns add-on is installed. The DNS parameters you specify here will be supplemented to the DNS configuration file generated based on **dnsPolicy**.

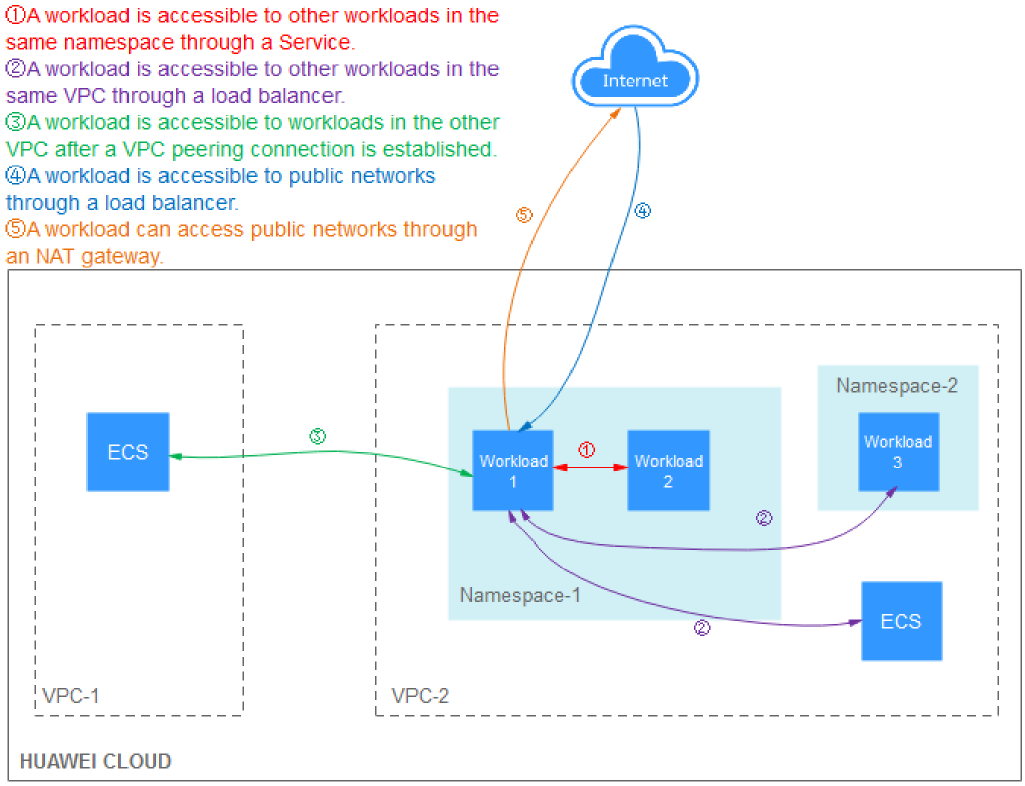
5 Workload Network Access

5.1 Network Access Overview

Workloads can be accesses over a private or public network, and they can also access the public network.

- **Private Network Access**: Access to intranet resources.
  - **Service**: allows workloads in the same namespace to access each other.
  - **ELB** (private network load balancer): allows a workload and other cloud resources (such as ECSs) in the same VPC to access each other. You can also use this method when you want workloads in the same VPC but different namespaces to access each other. The workload can be accessed using *Private domain name* or *Load balancer's IP address*:*Port* over HTTP/HTTPS and TCP/UDP. If other resources are in a VPC different from the workload, you can also create a **VPC peering connection** to enable communication between VPCs.
- **Public Network Access**: A workload can be accessed from public networks through a load balancer. The load balancer must be in the same VPC as the workload.
- **Accessing Public Networks from a Container**: Containers can access public networks by using SNAT rules, which are configured on the **NAT Gateway**.

Figure 5-1 Network access diagram


①A workload is accessible to other workloads in the same namespace through a Service.
②A workload is accessible to other workloads in the same VPC through a load balancer.
③A workload is accessible to workloads in the other VPC after a VPC peering connection is established.
④A workload is accessible to public networks through a load balancer.
⑤A workload can access public networks through an NAT gateway.
Internet
⑤
④
ECS
③
Workload 1
①
Workload 2
Namespace-2
Workload 3
②
Namespace-1
②
ECS
VPC-1
VPC-2
HUAWEI CLOUD

5.2 Private Network Access

The following two methods are available for private network access:

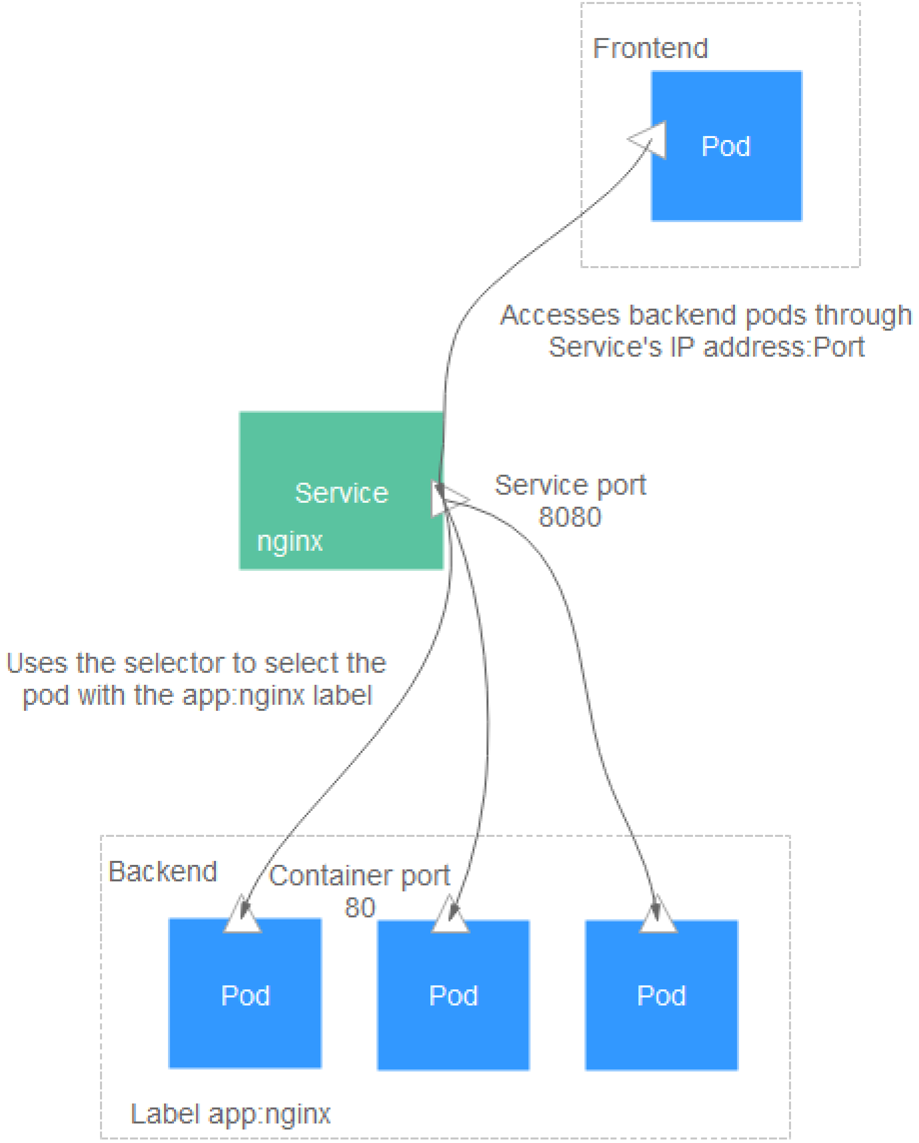
- **Workload Access Through a Service**: Use this method when workloads in the same namespace need to access each other.
- **Workload Access Through a Private Network Load Balancer**: Use this method when a workload need to access other cloud resources (such as ECSs) in the same VPC. You can also use this method when you want workloads in the same VPC but different namespaces to access each other. The workload can be accessed using *Private domain name* or *Load balancer's IP address*:*Port* over HTTP/HTTPS and TCP/UDP. If other resources are in a VPC different from the current workload, you can also create a **VPC peering connection** to enable communication between VPCs.

Pod is the smallest resource unit in the workload. Accessing a workload is to access the pods in the workload. Pods in a workload can be dynamically created and destroyed, for example, during capacity scaling or rolling upgrade. In this case, the pod addresses will change, which makes it inconvenient to access pods.

To solve this problem, CCI provides the coredns add-on (used for internal domain name resolution). Pod changes are managed by workloads and are not visible to users.

A workload can be accessed using *Service name*:*Workload access port*, where the access port is mapped to the container port. As shown in the following figure, if the pod in the frontend needs to access a pod in the backend, it only needs to access **nginx:8080**.

Figure 5-2 Workload access through the Service


Frontend
Pod
Accesses backend pods through Service's IP address:Port
Service
nginx
Service port 8080
Uses the selector to select the pod with the app:nginx label
Backend
Container port 80
Pod
Pod
Pod
Label app:nginx

Setting Service-based Workload Access When Creating a Workload

To enable a workload to be accessed through *Service name*:*Workload access port*, configure the following parameters when creating the workload:

- **Service Name**: Specify the Service, which will be used to manage pod access. For details, see **Service**.
- **coredns**: Enable this option if you want to use the coredns add-on. The coredns add-on resolves internal domain names of workloads. If you do not install this add-on, the workload cannot be accessed through *Service name*:*Workload access port*.
- **Workload Port Settings**:
  - **Protocol**: Specify the protocol that will be used to access the workload. Select **TCP** or **UDP**.

    - **Workload Access Port**: Specify the port for accessing the workload.
    - **Container Port**: Specify the port on which the container listens. The workload access port will be mapped to the container port.

Setting Service-based Workload Access After a Workload Is Created

You can configure Service-based access settings after a workload is created. The settings have no impact on the workload status and take effect immediately.

Step 1 Log in to the CCI console. In the navigation pane, choose **Network Management** > **Services**. On the page displayed, click **Create Service**.

Step 2 On the **Create Service** page, select **ClusterIP** for **Access Type**.

Step 3 Set intra-cluster access parameters.

- **Service Name**: Specify the Service, which will be used to manage pod access.
- **Namespace**: Specify the namespace that the workload belongs to.
- **Workload**: Select a workload that you want to add the Service for.
- **Port Settings**:
    - **Protocol**: Specify the protocol that will be used to access the workload. Select **TCP** or **UDP**.
    - **Access Port**: Specify the port for accessing the workload.
    - **Container Port**: Specify the port on which the container listens. The workload access port will be mapped to the container port.

Step 4 Click **Submit**. The ClusterIP Service will be added for the workload.

----End

Creating a Service Using kubectl

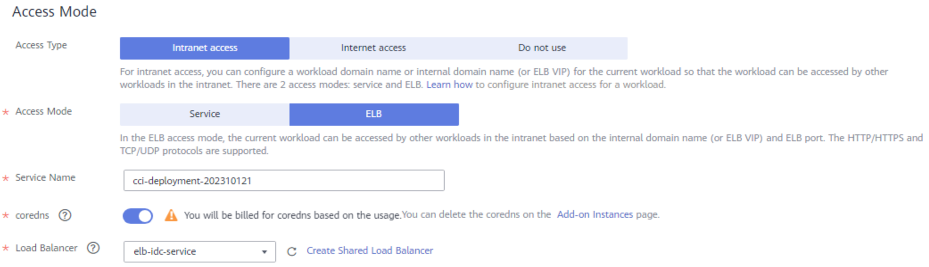
For details, see **Service**.

Workload Access Through a Private Network Load Balancer

If you want a workload to be accessed by other cloud resources or CCI workloads in other namespaces, bind a private network shared load balancer when you create the workload. In this way, the workload can be accessed over the private IP address of the load balancer.

When you configure access settings, select a private network load balancer and follow the description in **Public Network Access** to configure other parameters.

Figure 5-3 Setting ELB-based workload access when creating a workload



Access Mode

Access Type: Intranet access | Internet access | Do not use

For intranet access, you can configure a workload domain name or internal domain name (or ELB VIP) for the current workload so that the workload can be accessed by other workloads in the intranet. There are 2 access modes: service and ELB. Learn how to configure intranet access for a workload.

* Access Mode: Service | ELB

In the ELB access mode, the current workload can be accessed by other workloads in the intranet based on the internal domain name (or ELB VIP) and ELB port. The HTTP/HTTPS and TCP/UDP protocols are supported.

* Service Name: cci-deployment-202310121

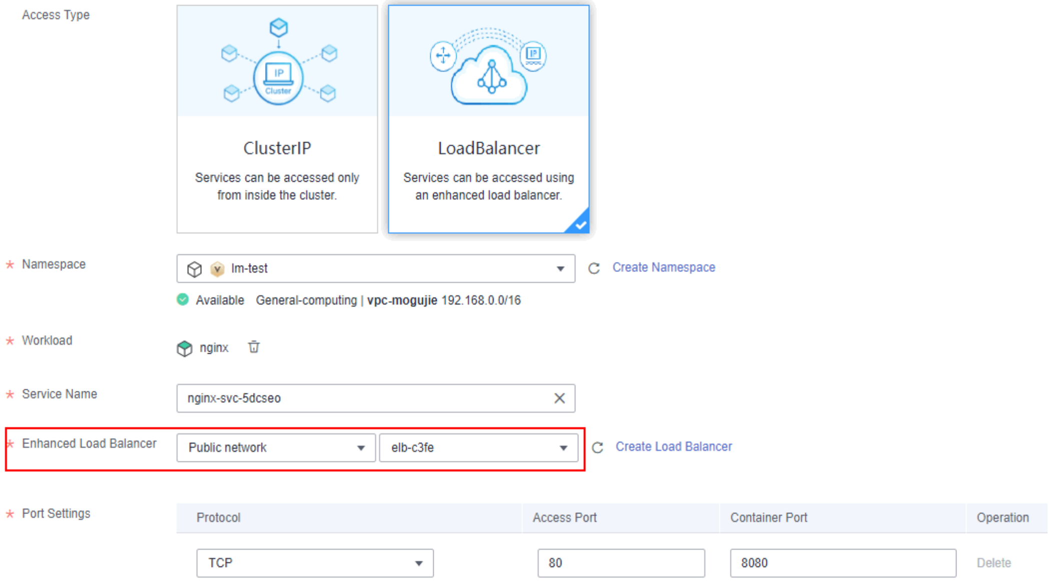
* coredns ⓘ: You will be billed for coredns based on the usage.You can delete the coredns on the Add-on Instances page.

* Load Balancer ⓘ: elb-idc-service

NOTE

CCI does not support dedicated load balancers.

Figure 5-4 Setting ELB-based workload access after a workload is created



Access Type

ClusterIP
Services can be accessed only from inside the cluster.

LoadBalancer
Services can be accessed using an enhanced load balancer.

* Namespace: lm-test — Create Namespace
Available General-computing | **vpc-mogujie** 192.168.0.0/16

* Workload: nginx

* Service Name: nginx-svc-5dcseo

* Enhanced Load Balancer: Public network, elb-c3fe — Create Load Balancer

* Port Settings

Protocol	Access Port	Container Port	Operation
TCP	80	8080	Delete

Setting Ingress-based Workload Access

You can configure ingress-based access settings after a workload is created. The settings have no impact on the workload status and take effect immediately.

Step 1 Log in to the CCI console. In the navigation pane, choose **Network Management** > **Ingresses**. On the page displayed, click **Create Ingress**.

Step 2 Set ingress parameters.

- **Ingress Name**: Enter a custom ingress name.
- **Namespace**: Select the namespace to which you want to add the ingress.
- **Load Balancer**: Specify a load balancer, which will automatically distribute Internet access traffic to multiple nodes running the workload.
- **External Port**: Specify a port number that is open to the ELB service address.
- **External Protocol**: **HTTP** and **HTTPS** are available. If you select **HTTPS**, choose a key certificate. For details about the certificate format, see **Certificate and Private Key Format**.

  NOTE

  – The key certificate **ingress-test-secret.yaml** is required only if you have selected HTTPS. For details on how to create a key, see **Secrets**.

  – If there is already an HTTPS ingress for the chosen port on the load balancer, the certificate of the new HTTPS ingress must be the same as the certificate of the existing ingress. This means that a listener has only one certificate. If two certificates, each with a different ingress, are added to the same listener of the same load balancer, only the certificate that was added first will take effect for the load balancer.

- **Domain Name**: (Optional) Specify the domain name that will be used for access. You need to have registered a domain name. Ensure that the domain name can be mapped to the IP address of the selected load balancer. If you have configured a domain name rule, the domain name must always be used for access.
- **Ingress Rule**
  - **Rule Matching**: Currently, only **Prefix match** is supported.

    Prefix match: Specify the prefix to be matched. If the mapping URL is **/healthz**, the URL that meets the prefix can be accessed. For example, **/healthz/v1** and **/healthz/v2**.
  - **URL**: Specify the access path to be registered.
  - **Service Name**: Select the Service whose ingress you want to add.
  - **Service Port**: Specify the port on which the container in the container image listens.

Step 3 Click **Submit**.

After the ingress is created, it is displayed in the ingress list.

----End

Updating a Service

After you add a Service, you can update the port configuration of the Service.

Step 1 Log in to the CCI console. In the navigation pane, choose **Network Management** > **Services**. On the **Services** page, select the corresponding namespace, and click **Update** in the row that contains the Service.

Step 2 Update intra-cluster access parameters.

- **Namespace**: Specify the namespace that the workload belongs to. The value is inherited from the workload creation page and cannot be changed.
- **Workload**: Specify the workload that you want to add a Service for. The value is inherited from the workload creation page and cannot be changed.
- **Service Name**: Specify the Service, which will be used to manage pod access. The value is inherited from the workload creation page and cannot be changed.
- **Port Settings**:
  - **Protocol**: Select a protocol used by the Service.
  - **Container Port**: Specify the port on which the workload listens. The Nginx workload listens on port 80.
  - **Access Port**: Specify the port mapped to the container port at the cluster-internal IP address. The workload can be accessed at <cluster-internal IP address>:<access port>. The port number range is 1–65535.

Step 3 Click **Submit**. The Service will be updated for the workload.

----End

Updating an Ingress

After adding an ingress, you can update its port, domain name, and route configuration.

Step 1 Log in to the CCI console. In the navigation pane, choose **Network Management** > **Ingresses**, select the corresponding namespace, and click **Update** in the row where the ingress to be updated resides.

Step 2 On the **Update Ingress** page, set the following parameters:

- **External Port**: Specify a port number that is open to the ELB service address.
- **Domain Name**: (Optional) Specify the domain name that will be used for access. You need to have registered a domain name. Ensure that the domain name can be mapped to the IP address of the selected load balancer. If you have configured a domain name rule, the domain name must always be used for access.
- **Ingress Rule**: You can click **Add Ingress Rule** to add a rule.
  - **Rule Matching**: Currently, only **Prefix match** is supported.

    Prefix match: Specify the prefix to be matched. If the mapping URL is **/healthz**, the URL that meets the prefix can be accessed. For example, **/healthz/v1** and **/healthz/v2**.
  - **URL**: Specify the access path to be registered, for example, **/healthz**.
  - **Service Name**: Select the Service whose ingress you want to update.
  - **Service Port**: Specify the port on which the container in the container image listens.

Step 3 Click **Submit**. The ingress will be updated for the workload.

----End

5.3 Public Network Access

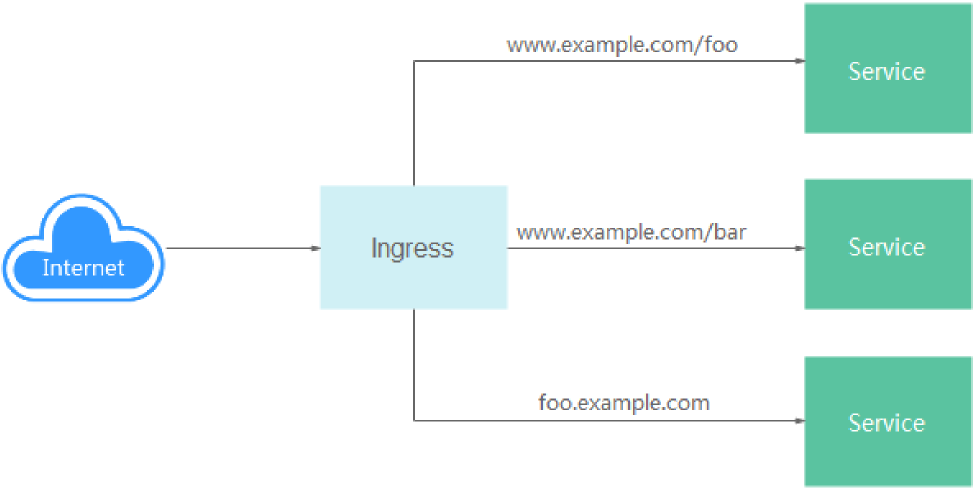
Overview

CCI allows workloads to be accessed from public networks. To implement this access, you need to bind a shared load balancer to a workload. The load balancer must be in the same VPC as the workload. Public network access at both Layer 4 and Layer 7 is supported.

- TCP and UDP are supported for Layer-4 public network access. After you complete the configuration, the workload can be accessed using *Public network IP address of the load balancer*:*Load balancer port*.
- HTTP and HTTPS are supported for Layer-7 public network access. After you complete the configuration, the workload can be accessed using **http://***Public network domain name or public network IP address of the load balancer*:*Load balancer port*/*Mapping path*.

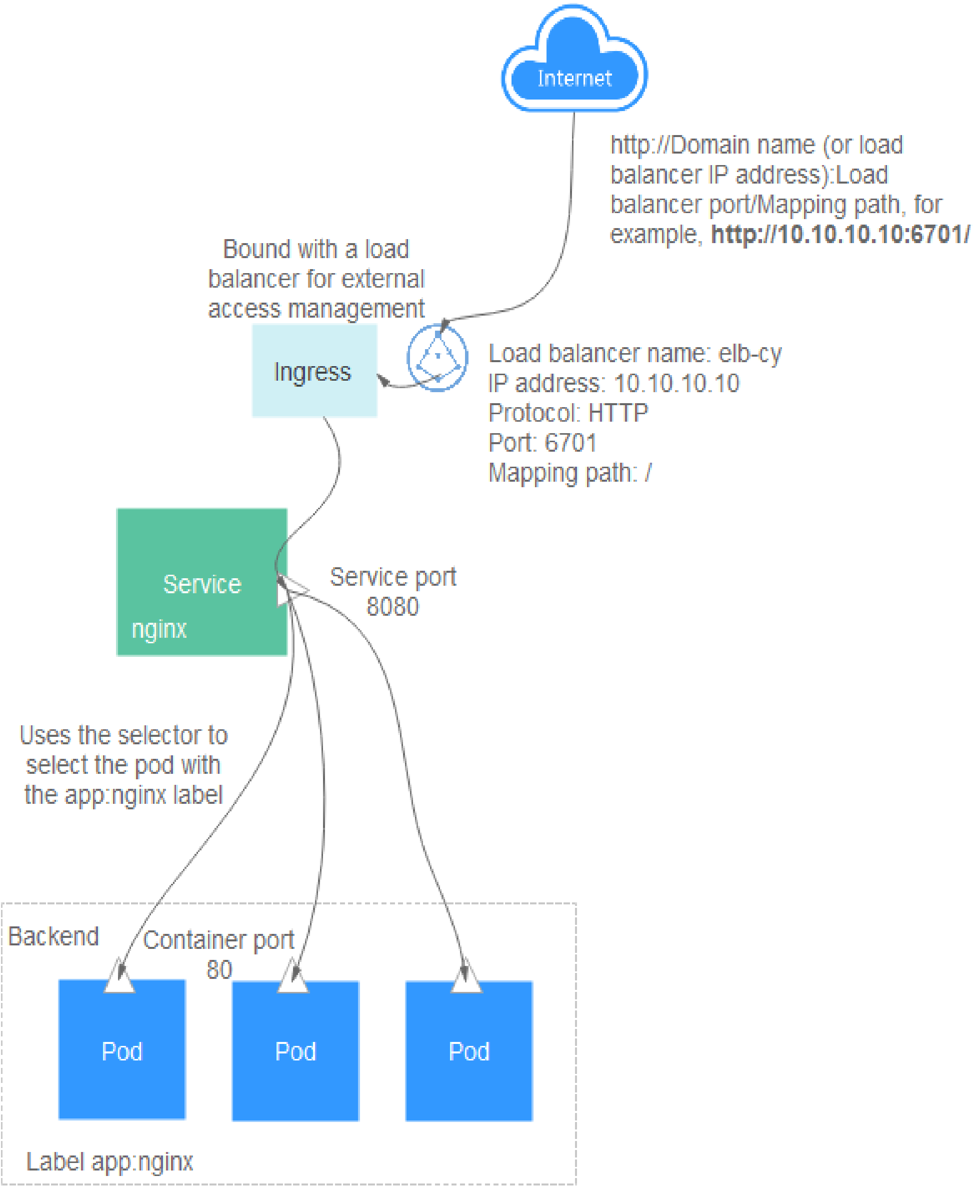
Services forward requests using TCP and UDP protocols at Layer 4. Ingresses forward requests using HTTP and HTTPS at Layer 7. You can use domain names and paths to achieve finer granularity, as shown in the following figure.

Figure 5-5 Ingress-Service



The following figure shows an example of accessing a workload using the HTTP protocol.

Figure 5-6 Public network access


Internet
http://Domain name (or load balancer IP address):Load balancer port/Mapping path, for example, **http://10.10.10.10:6701/**
Bound with a load balancer for external access management
Ingress
Load balancer name: elb-cy
IP address: 10.10.10.10
Protocol: HTTP
Port: 6701
Mapping path: /
Service
nginx
Service port 8080
Uses the selector to select the pod with the app:nginx label
Backend
Container port 80
Pod
Pod
Pod
Label app:nginx

Constraints and Restrictions

Before using an EIP, familiarize yourself with the **EIP restrictions**.

Setting Public Network Access When Creating a Workload

When you create a workload, select **Internet access** for **Access Type** and configure the following parameters:

- **Service Name**: Specify the Service, which will be used to manage pod access. For details, see **Service**.
- **coredns**: Enable this option if you want to use the coredns add-on. The coredns add-on resolves internal domain names of workloads. If you do not install this add-on, the workload cannot be accessed through *Service name*:*Workload access port*.

- **Load Balancer**: Select a load balancer. If no load balancer is available, click **Create Load Balancer** to create one.

NOTICE

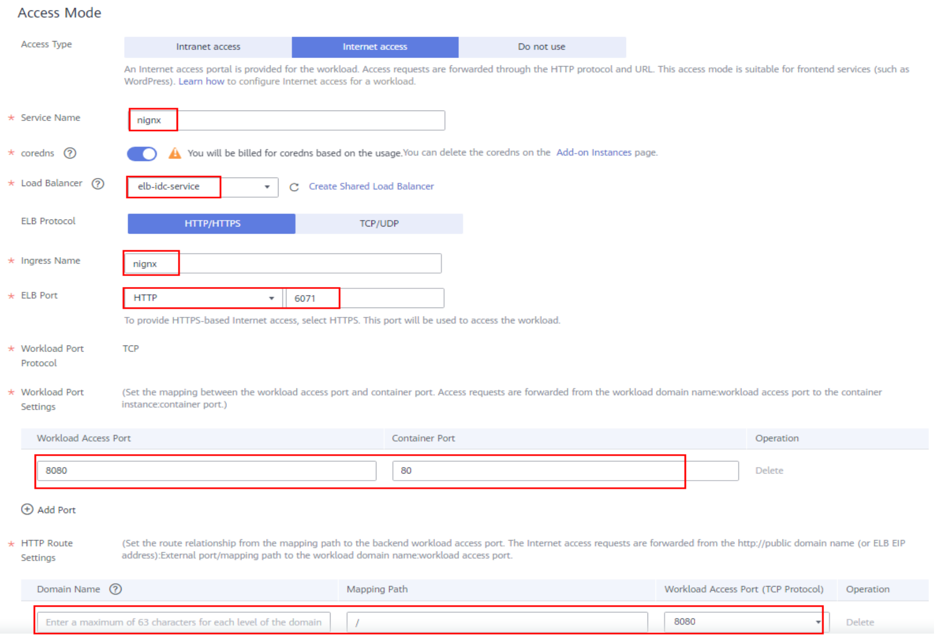
The load balancer must be in the same VPC as the workload.

CCI does not support dedicated load balancers. Create a shared load balancer.

- **ELB Protocol**: Specify the communication protocol for public network access, which can be **HTTP/HTTPS** or **TCP/UDP**.
- **Ingress Name**: Specify the ingress, which will be used to manage Layer-7 access. If you do not set this parameter, the workload name will be used as the ingress name by default. For details, see **Ingress**.
- **Public Domain Name** (configurable if you have selected HTTP or HTTPS): If you want the workload to be accessed using a domain name, you need to have purchased a public domain name and ensure that the domain name points to the EIP of the load balancer.
- **Certificate** (mandatory when the HTTPS protocol is selected): For details about how to import an SSL certificate, see **SSL Certificates**.
- **ELB Port**: Select the protocol and port for accessing the workload using the load balancer.
- **Workload Port Protocol**: Specify the communication protocol for accessing the workload, which can be **TCP** or **UDP**. If you have set the ELB protocol to **HTTP/HTTPS**, the workload port protocol will be automatically displayed as **TCP**.
- **Workload Port Settings**:
  - **Workload Access Port**: Specify the port for accessing the workload.
  - **Container Port**: Specify the port on which the container listens. The workload access port will be mapped to the container port.
- **HTTP Route Settings**:
  - **Mapping Path**: Specify the path to be accessed. It must start with a slash (/). For example, **/api/web**. It can also be the root path **/**.
  - **Workload Access Port**: Specify the previously configured workload access port.

As shown in the following figure, if the IP address of the load balancer is **10.10.10.10**, you can access the workload by visiting **http://10.10.10.10:6071/**.

Figure 5-7 Configuring public network access parameters



Access Mode

Access Type: Intranet access | **Internet access** | Do not use

An Internet access portal is provided for the workload. Access requests are forwarded through the HTTP protocol and URL. This access mode is suitable for frontend services (such as WordPress). Learn how to configure Internet access for a workload.

* Service Name: nignx

* coredns: You will be billed for coredns based on the usage.You can delete the coredns on the Add-on Instances page.

* Load Balancer: elb-idc-service

Create Shared Load Balancer

ELB Protocol: **HTTP/HTTPS** | TCP/UDP

* Ingress Name: nignx

* ELB Port: HTTP 6071

To provide HTTPS-based Internet access, select HTTPS. This port will be used to access the workload.

* Workload Port Protocol: TCP

* Workload Port Settings: (Set the mapping between the workload access port and container port. Access requests are forwarded from the workload domain name:workload access port to the container instance:container port.)

Workload Access Port	Container Port	Operation
8080	80	Delete

Add Port

* HTTP Route Settings: (Set the route relationship from the mapping path to the backend workload access port. The Internet access requests are forwarded from the http://public domain name (or ELB EIP address):External port/mapping path to the workload domain name:workload access port.

Domain Name	Mapping Path	Workload Access Port (TCP Protocol)	Operation
Enter a maximum of 63 characters for each level of the domain	/	8080	Delete

Setting Public Network Access After a Workload Is Created

You can configure Service-based access settings after a workload is created. The settings have no impact on the workload status and take effect immediately.

Step 1 Log in to the CCI console. In the navigation pane, choose **Network Management** > **Services**. On the page displayed, click **Create Service**.

Step 2 On the **Create Service** page, select **LoadBalancer** for **Access Type**.

Step 3 Set ELB-based access parameters.

- **Service Name**: Specify the Service, which will be used to manage pod access.
- **Namespace**: Specify the namespace that the workload belongs to.
- **Workload**: Select a workload that you want to add the Service for.
- **Load Balancer**: Select a public network load balancer. If no load balancer is available, click **Create Load Balancer** to create one.

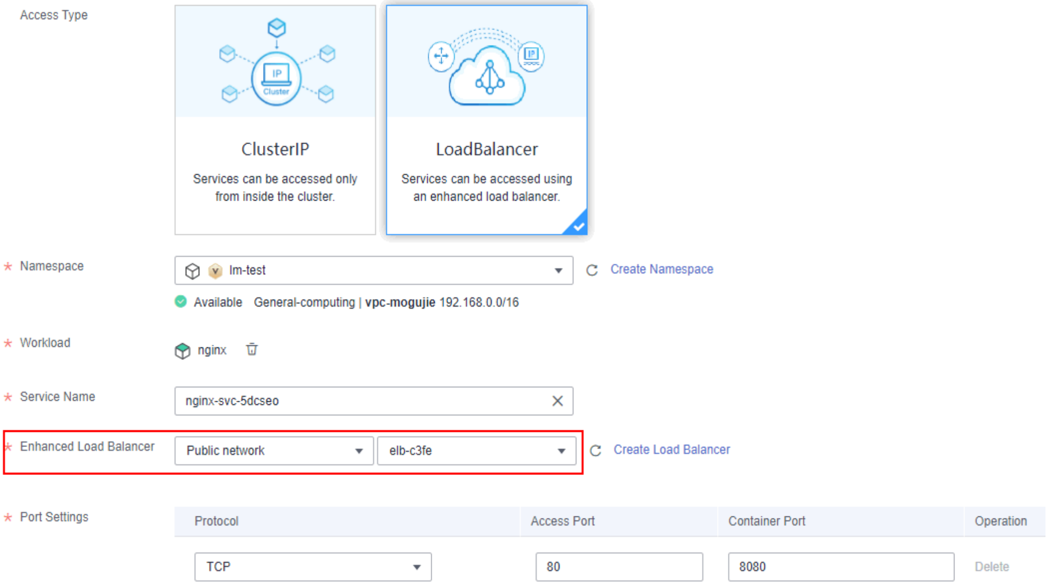
NOTICE

The load balancer must be in the same VPC as the workload.

CCI does not support dedicated load balancers. Create a shared load balancer.

- **Port Settings**:
  - **Protocol**: Specify the protocol that will be used to access the workload. Select **TCP** or **UDP**.
  - **Access Port**: Specify the port for accessing the workload.
  - **Container Port**: Specify the port on which the container listens. The workload access port will be mapped to the container port.

Figure 5-8 Setting public network access after a workload is created



Access Type

ClusterIP
Services can be accessed only from inside the cluster.

LoadBalancer
Services can be accessed using an enhanced load balancer.

* Namespace: lm-test — Create Namespace
Available General-computing | vpc-mogujie 192.168.0.0/16

* Workload: nginx

* Service Name: nginx-svc-5dcseo

* Enhanced Load Balancer: Public network, elb-c3fe — Create Load Balancer

* Port Settings

Protocol	Access Port	Container Port	Operation
TCP	80	8080	Delete

Step 4 Click **Submit**. The LoadBalancer Service will be added for the workload.

----End

Setting DNAT-based Workload Access

After a workload is created, you can access the workload pods through ELB or DNAT from a public network. The following describes how to set DNAT-based workload access:

Step 1 Create a NAT gateway.

Step 2 Use kubectl to create a DNAT Service. For details, see **Service**. The following is an example of creating a DNAT Service:

```
apiVersion: v1
kind: Service
metadata:
  name: nginx
  namespace: default  # User namespace. The default value is default.
  annotations:
    kubernetes.io/elb.class: dnat       # The type is set to DNAT.
    kubernetes.io/natgateway.id: 4b8cda3d-3543-4ebd-a55e-ca610b3b3c43 # NAT gateway ID
spec:
  loadBalancerIP: 100.85.218.195       # EIP used by DNAT
  selector:
    app: nginx
  ports:
  - name: service0
    targetPort: 80     # Port exposed by the pod
    port: 8080         # DNAT access port
    protocol: TCP
   type: LoadBalancer   # Service type
```

Step 3 After the Service is created, you can run the **kubectl describe <*service_name*> -n <*service_namespace*>** command to view the Service update status.

----End

After the Service is created and updated, you can access the pod by using EIP and port.

Constraints:

1. One DNAT rule can be forwarded to only one backend. Therefore, a DNAT Service can be associated with only one backend pod. If more than one backend pod is associated, the DNAT rule fails.
2. A maximum of 200 DNAT rules can be added to a NAT gateway. For details, see the **NAT Gateway** documentation.
3. You can view the information about the DNAT Service on the frontend. Do not modify the information on the frontend.
4. If the subnet does not use the default route, add a route of the NAT gateway to the corresponding route table.
5. If you configure the DNAT Service for the gateway used by the SNAT rule, ensure that the EIP used by the DNAT Service is different from that bound to the SNAT rule.
6. For details about how to use the NAT gateway, see the **NAT Gateway** documentation.

Setting Ingress-based Workload Access

You can configure ingress-based access settings after a workload is created. The settings have no impact on the workload status and take effect immediately.

Step 1 Log in to the CCI console. In the navigation pane, choose **Network Management** > **Ingresses**. On the page displayed, click **Create Ingress**.

Step 2 Set ingress parameters.

- **Ingress Name**: Enter a custom ingress name.
- **Namespace**: Select the namespace to which you want to add the ingress.
- **Load Balancer**: Specify a load balancer, which will automatically distribute Internet access traffic to multiple nodes running the workload.
- **External Port**: Specify a port number that is open to the ELB service address.
- **External Protocol**: **HTTP** and **HTTPS** are available. If you select **HTTPS**, choose a key certificate. For details about the certificate format, see **Certificate and Private Key Format**.

  NOTE

  - The key certificate **ingress-test-secret.yaml** is required only if you have selected HTTPS. For details on how to create a key, see **Secrets**.
  - If there is already an HTTPS ingress for the chosen port on the load balancer, the certificate of the new HTTPS ingress must be the same as the certificate of the existing ingress. This means that a listener has only one certificate. If two certificates, each with a different ingress, are added to the same listener of the same load balancer, only the certificate that was added first will take effect for the load balancer.
- **Domain Name**: (Optional) Specify the domain name that will be used for access. You need to have registered a domain name. Ensure that the domain name can be mapped to the IP address of the selected load balancer. If you have configured a domain name rule, the domain name must always be used for access.
- **Ingress Rule**

- **Rule Matching**: Currently, only **Prefix match** is supported.

  Prefix match: Specify the prefix to be matched. If the mapping URL is **/healthz**, the URL that meets the prefix can be accessed. For example, **/healthz/v1** and **/healthz/v2**.
- **URL**: Specify the access path to be registered.
- **Service Name**: Select the Service whose ingress you want to add.
- **Service Port**: Specify the port on which the container in the container image listens.

Step 3 Click **Submit**.

After the ingress is created, it is displayed in the ingress list.

----End

Troubleshooting the Failure to Access a Workload from the Public Network

1. A workload can be accessed from the public network only when it is in the running state. If your workload is abnormal or not ready, it cannot be accessed from the public network.
2. It may take 1 to 3 minutes from the time when the workload was created to the time for it to be ready for public network access. During this time period, the network route has not yet been configured. As a result, the workload cannot be accessed from the public network.
3. If the workload cannot be accessed 3 minutes after being created, click the workload. On the displayed details page, click the **Events** tab under **Access Settings** to check whether any alarm events are reported. The following are two common events:
   - Listener port is repeated: This event occurs when you delete a workload for which a load balancer port is configured, and immediately after that, create a workload using the same load balancer port. It takes some time for a load balancer port to be deleted. It is recommended that you delete the workload and create it again or wait for 5–10 minutes until the Internet access is normal.
   - Create listener failed: This event typically occurs because the listener quota is exceeded. Select another load balancer with a sufficient quota.
4. The workload is inaccessible 3 minutes after it is created, and there is no alarm event. The possible reason is that no corresponding process is actually listening to the user-configured container port. Currently, CCI cannot detect this type of exception. You need to check whether the image is listening to the container port. If the container port is being properly listened to, the access failure may be due to the load balancer. In this case, check the status of the load balancer.

Enabling Public Network Access Using kubectl

To enable the access to a workload from the public network, two Kubernetes objects (that is, Service and ingress) are required. For details, see **Service** and **Ingress**.

Updating a Service

After you add a Service, you can update the port configuration of the Service.

Step 1 Log in to the CCI console. In the navigation pane, choose **Network Management** > **Services**. On the **Services** page, select the corresponding namespace, and click **Update** in the row that contains the Service.

Step 2 Update load balancing parameters.

- **Namespace**: Specify the namespace that the workload belongs to. The value is inherited from the workload creation page and cannot be changed.
- **Workload**: Specify the workload that you want to update the Service for. The value is inherited from the workload creation page and cannot be changed.
- **Service Name**: Specify the Service, which will be used to manage pod access. The value is inherited from the workload creation page and cannot be changed.
- **Load Balancer**: The value is inherited from the workload creation page and cannot be changed.
- **Port Settings**
  - **Protocol**: Specify the protocol that will be used to access the workload. Select **TCP** or **UDP**.
  - **Access Port**: Specify the port for accessing the workload.
  - **Container Port**: Specify the port on which the container listens. The workload access port will be mapped to the container port.

Step 3 Click **Submit**. The Service will be updated for the workload.

----End

Updating an Ingress

After adding an ingress, you can update its port, domain name, and route configuration.

Step 1 Log in to the CCI console. In the navigation pane, choose **Network Management** > **Ingresses**, select the corresponding namespace, and click **Update** in the row where the ingress to be updated resides.

Step 2 On the **Update Ingress** page, set the following parameters:

- **External Port**: Specify a port number that is open to the ELB service address.
- **Domain Name**: (Optional) Specify the domain name that will be used for access. You need to have registered a domain name. Ensure that the domain name can be mapped to the IP address of the selected load balancer. If you have configured a domain name rule, the domain name must always be used for access.
- **Ingress Rule**: Click **Add Ingress Rule** to add a rule.
  - **Rule Matching**: Currently, only **Prefix match** is supported.

    Prefix match: Specify the prefix to be matched. If the mapping URL is **/healthz**, the URL that meets the prefix can be accessed. For example, **/healthz/v1** and **/healthz/v2**.

- **URL**: Specify the access path to be registered, for example, **/healthz**.
- **Service Name**: Select the Service whose ingress you want to update.
- **Service Port**: Specify the port on which the container in the container image listens.

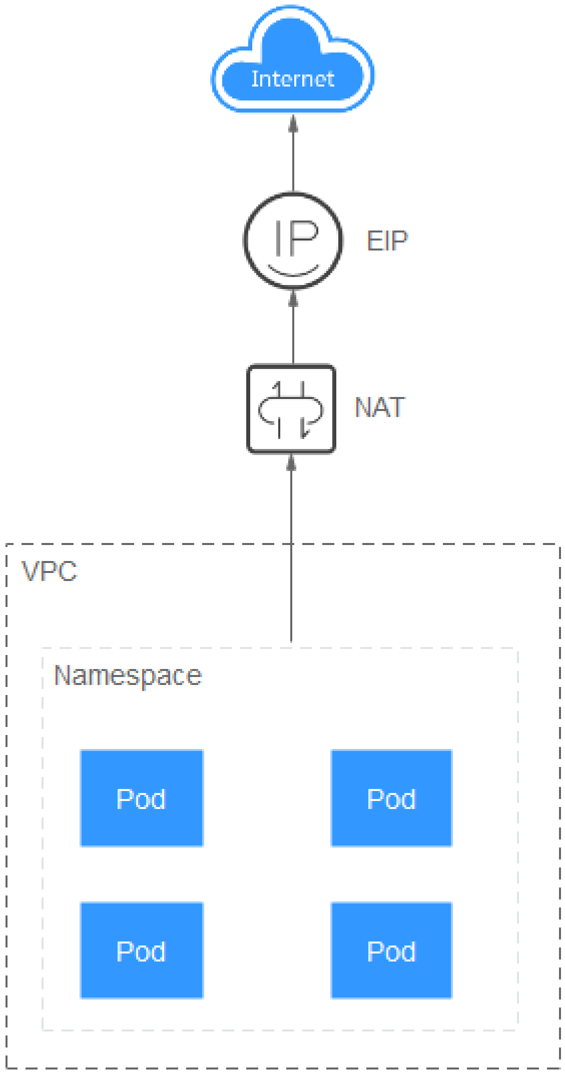
Step 3 Click **Submit**. The ingress will be updated for the workload.

----End

5.4 Accessing Public Networks from a Container

You can use **NAT Gateway** to enable containers in a VPC to access public networks. NAT Gateway provides source network address translation (SNAT), which translates private IP addresses to a public IP address by binding an EIP to the NAT gateway, providing secure and efficient access to the Internet. **Figure 5-9** shows the SNAT architecture. The SNAT function allows the pods in a VPC to access the Internet without being bound to an EIP. SNAT supports a large number of concurrent connections, which makes it suitable for applications involving a large number of requests and connections.

Figure 5-9 SNAT



To enable pods to access the Internet, perform the following steps:

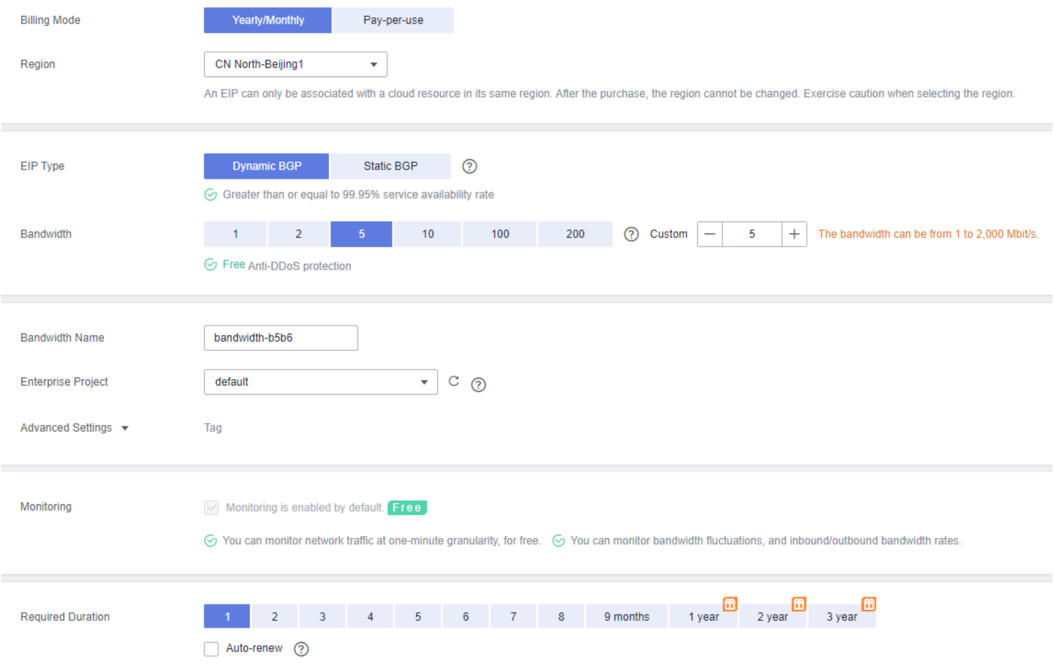
Step 1 **Buy** an EIP.

1. Log in to the management console.
2. Click 📍 in the upper left corner to select the desired region and project.
3. Choose **Service List** > **Networking** > **Virtual Private Cloud**.
4. In the navigation pane, choose **Elastic IP and Bandwidth** > **EIPs**.
5. On the **EIPs** page, click **Buy EIP**.
6. Set the parameters.

   NOTE

   Set **Region** to the region where pods are located.

Figure 5-10 Buying an EIP



Billing Mode	**Yearly/Monthly** · Pay-per-use
Region	CN North-Beijing1 An EIP can only be associated with a cloud resource in its same region. After the purchase, the region cannot be changed. Exercise caution when selecting the region.
EIP Type	**Dynamic BGP** · Static BGP Greater than or equal to 99.95% service availability rate
Bandwidth	1 · 2 · **5** · 10 · 100 · 200 · Custom − 5 + The bandwidth can be from 1 to 2,000 Mbit/s. Free Anti-DDoS protection
Bandwidth Name	bandwidth-b5b6
Enterprise Project	default
Advanced Settings	Tag
Monitoring	☑ Monitoring is enabled by default. Free You can monitor network traffic at one-minute granularity, for free. You can monitor bandwidth fluctuations, and inbound/outbound bandwidth rates.
Required Duration	**1** · 2 · 3 · 4 · 5 · 6 · 7 · 8 · 9 months · 1 year · 2 year · 3 year □ Auto-renew

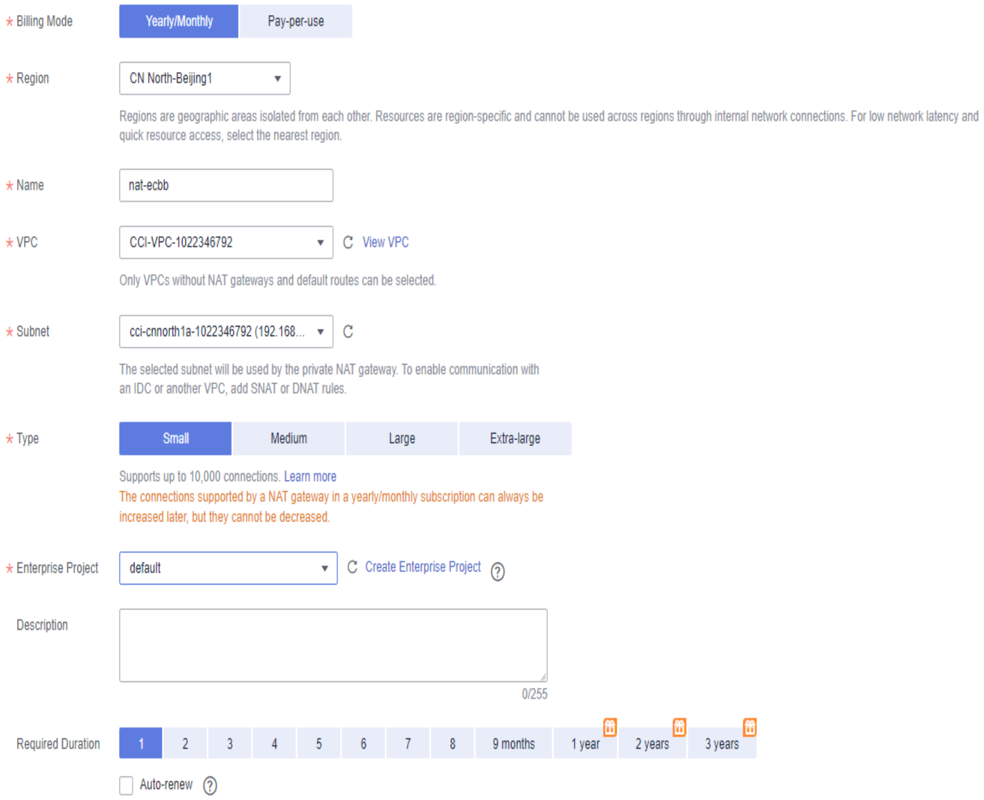
Step 2 Buy a NAT gateway. For details, see **Buy a Public NAT Gateway**.

1. Log in to the management console.
2. Click 📍 in the upper left corner to select the desired region and project.
3. Choose **Service List** > **Networking** > **NAT Gateway**.
4. On the displayed page, click **Buy NAT Gateway**.
5. Set the parameters.

NOTE

Select the VPC and subnet that you have configured for the **namespace** where the pods are located.

Figure 5-11 Buying a NAT gateway



* Billing Mode: Yearly/Monthly | Pay-per-use

* Region: CN North-Beijing1

Regions are geographic areas isolated from each other. Resources are region-specific and cannot be used across regions through internal network connections. For low network latency and quick resource access, select the nearest region.

* Name: nat-ecbb

* VPC: CCI-VPC-1022346792 View VPC

Only VPCs without NAT gateways and default routes can be selected.

* Subnet: cci-cnnorth1a-1022346792 (192.168...

The selected subnet will be used by the private NAT gateway. To enable communication with an IDC or another VPC, add SNAT or DNAT rules.

* Type: Small | Medium | Large | Extra-large

Supports up to 10,000 connections. Learn more

The connections supported by a NAT gateway in a yearly/monthly subscription can always be increased later, but they cannot be decreased.

* Enterprise Project: default Create Enterprise Project

Description:

0/255

Required Duration: 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 months | 1 year | 2 years | 3 years

□ Auto-renew

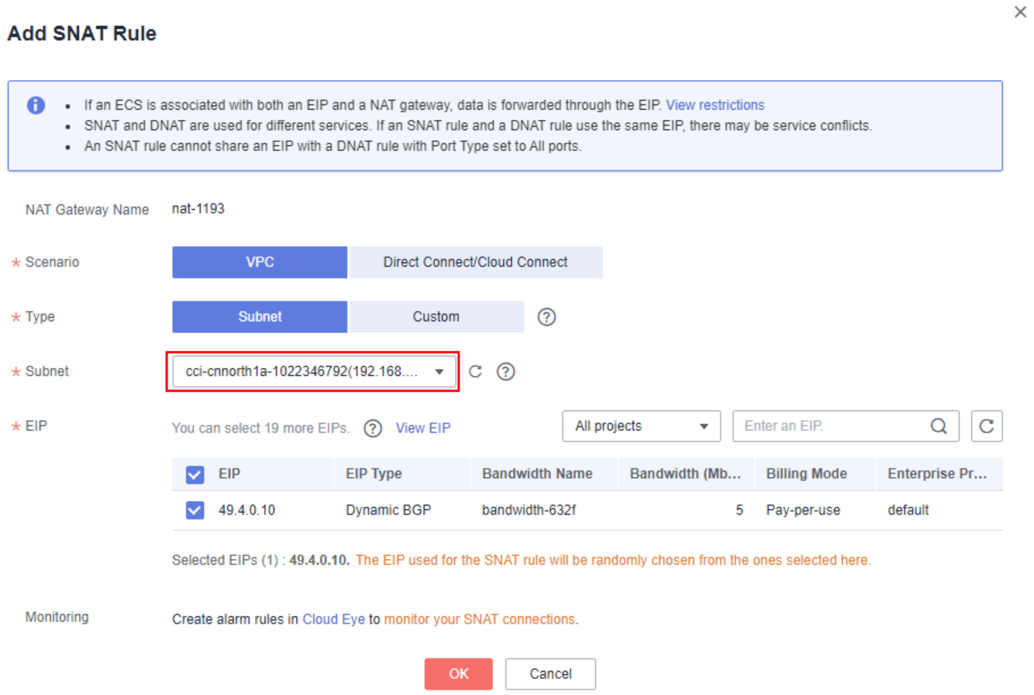
Step 3 Configure an SNAT rule and bind the EIP to the subnet. For details, see **Add an SNAT Rule**.

1. Log in to the management console.
2. Click in the upper left corner to select the desired region and project.
3. Choose **Service List** > **Networking** > **NAT Gateway**.
4. On the displayed page, click the name of the NAT gateway for which you want to add the SNAT rule.
5. On the **SNAT Rules** tab page, click **Add SNAT Rule**.
6. Set the parameters.

NOTE

Select the subnet that you have configured for the **namespace** where the pods are located.

Figure 5-12 Adding an SNAT rule



Add SNAT Rule

- If an ECS is associated with both an EIP and a NAT gateway, data is forwarded through the EIP. View restrictions
- SNAT and DNAT are used for different services. If an SNAT rule and a DNAT rule use the same EIP, there may be service conflicts.
- An SNAT rule cannot share an EIP with a DNAT rule with Port Type set to All ports.

NAT Gateway Name: nat-1193

* Scenario: VPC | Direct Connect/Cloud Connect

* Type: Subnet | Custom

* Subnet: cci-cnnorth1a-1022346792(192.168....

* EIP: You can select 19 more EIPs. View EIP

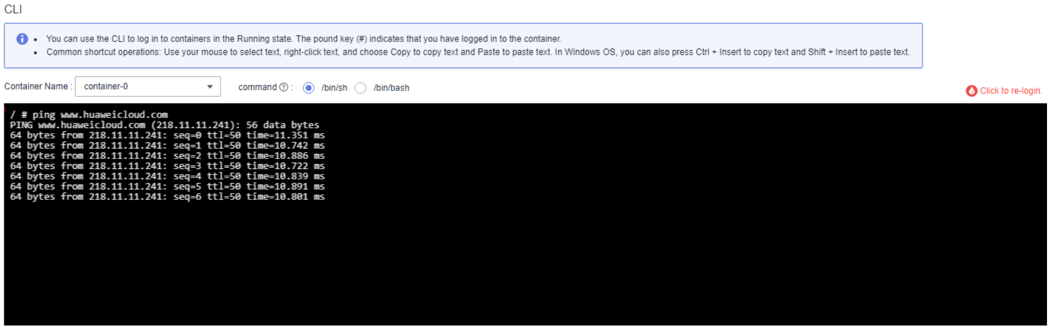
☑ EIP	EIP Type	Bandwidth Name	Bandwidth (Mb...	Billing Mode	Enterprise Pr...
☑ 49.4.0.10	Dynamic BGP	bandwidth-632f	5	Pay-per-use	default

Selected EIPs (1) : **49.4.0.10.** The EIP used for the SNAT rule will be randomly chosen from the ones selected here.

Monitoring: Create alarm rules in Cloud Eye to monitor your SNAT connections.

After the SNAT rule is configured, public networks can be accessed from the container, as shown in the following figure.

Figure 5-13 Accessing public networks from a container



CLI

- You can use the CLI to log in to containers in the Running state. The pound key (#) indicates that you have logged in to the container.
- Common shortcut operations: Use your mouse to select text, right-click text, and choose Copy to copy text and Paste to paste text. In Windows OS, you can also press Ctrl + Insert to copy text and Shift + Insert to paste text.

Container Name : container-0   command : ● /bin/sh ○ /bin/bash

```
/ # ping www.huaweicloud.com
PING www.huaweicloud.com (218.11.11.241): 56 data bytes
64 bytes from 218.11.11.241: seq=0 ttl=50 time=11.351 ms
64 bytes from 218.11.11.241: seq=1 ttl=50 time=10.742 ms
64 bytes from 218.11.11.241: seq=2 ttl=50 time=10.886 ms
64 bytes from 218.11.11.241: seq=3 ttl=50 time=10.722 ms
64 bytes from 218.11.11.241: seq=4 ttl=50 time=10.839 ms
64 bytes from 218.11.11.241: seq=5 ttl=50 time=10.891 ms
64 bytes from 218.11.11.241: seq=6 ttl=50 time=10.801 ms
```

----End

6 Storage Management

6.1 Overview

CCI supports multiple types of persistent storage to meet your requirements in different scenarios. You can use the following types of storage volumes when creating a workload:

- Elastic Volume Service (EVS) volumes

  You can mount an EVS volume into a container path. When the container is migrated, the mounted EVS volume is also migrated. EVS volumes are ideal for persistent data storage. For details, see **EVS Volumes**.

  NOTE

  When using EVS volumes to store data, note the following:

  - You cannot mount an EVS volume to multiple pods.
  - EVS disks in multiple AZs cannot be mounted to a single pod.

- SFS Turbo volumes

  You can create SFS Turbo volumes and mount them to specific container paths. SFS Turbo volumes are fast, on-demand, and scalable. They are suitable for DevOps, containerized microservices, and enterprise office applications. For details, see **SFS Turbo Volumes**.

PersistentVolumeClaim (PVC)

CCI uses PVCs to apply for and manage persistent storage. With PVCs, you only need to specify the type and capacity of storage resources and do not need to worry about how to create and release underlying storage resources.

You can bind a PVC to the volume in a pod and use persistent storage through the PVC.

On the CCI console, you can import existing EVS disks and SFS Turbo file systems. When you import these storage resources, CCI creates a PVC for them.

You can also purchase EVS disks on the CCI console. After the purchase, CCI will create PVCs for them and import them.

6.2 EVS Volumes

To meet data persistency requirements, CCI allows you to mount **EVS** volumes to containers. By using EVS disks, you can mount the file directory of a storage system to a container so that data in the volume is permanently preserved. Even if the container is deleted, only the volume is unmounted. Data in the volume is still stored in the storage system.

EVS supports three specifications: common I/O (previous-generation product), high I/O, and ultra-high I/O.

- Common I/O (previous-generation product): The backend storage is provided by the SATA storage medium. It is perfect for high-capacity application scenarios with low read/write rate requirements and less transaction processing, such as scenarios involving development, testing, and enterprise office applications.
- High I/O: The backend storage is provided by the SAS storage medium. It is perfect for application scenarios with relatively high performance, high read/write rate requirements, and real-time data storage requirements, such as scenarios involving file system creation and distributed file sharing.
- Ultra-high I/O: The backend storage is provided by the SSD storage medium. It is perfect for application scenarios with high performance, high read/write rate requirements, and data-intensive requirements, such as scenarios involving NoSQL, relational database, and data warehouses (such as Oracle RAC and SAP HANA).

Constraints

- EVS disks to be mounted are billed on a pay-per-use basis. For pricing details, see **EVS Billing**.
- You cannot import the following EVS disks if they not in the current AZ, they are unavailable or frozen, or they are system disks, CCE-associated disks, non-SCSI disks, dedicated disks, or HANA server dedicated disks (high I/O performance optimization/ultra-high I/O latency optimization).
- You can use an EVS volume only as a new disk. The content in the EVS volume that has not been mounted to CCI is invisible to the container.
- If you delete an imported EVS disk from the EVS console, it cannot be detected by CCI. Therefore, delete the EVS disk after you confirm that it is not being used by any workload.
- You can mount an EVS volume to only one pod. Otherwise, data may be lost.
- EVS disk expansion is imperceptible to CCI. You need to unbind the EVS disk on the **EVS** page of the CCI console before the expansion, and import it again after the expansion is complete.

Adding EVS Disks

Step 1 Log in to the CCI console. In the navigation pane, choose **Storage** > **EVS**.

- If you have purchased EVS disks on the **EVS** console, go to **Step 2**.
- If you have not purchased any EVS disk, go to **Step 3**.

Step 2 Click **Import**. On the **Import EVS Disk** page, select one or more EVS disks that you want to import and click **Import**.

> NOTE
>
> You can import an EVS disk into one namespace only. After you import an EVS disk into a namespace, it will not be available for import in other namespaces. **If you want to import an EVS disk that has its file system (ext4) formatted, ensure that no partition has been created for the disk. Otherwise, data may be lost.**

After you import the EVS disk, you can see the corresponding volume.

Step 3 Click **Buy EVS Volume**. On the displayed page, set the parameters, click **Next**, confirm the specifications, and click **Submit**.

- **PVC Name**: Enter the PVC name.
- **Namespace**: Select the namespace that the PVC belongs to.
- **AZ**: Specify the availability zone to which the disk belongs.
- **Type**: Specify the disk type, which can be common I/O (previous-generation product), high I/O, or ultra-high I/O.
- **Capacity**: The value range is **10** to **32768**, in GiB.

----End

Using EVS Volumes

After selecting a container in **Creating a Deployment**, expand **Advanced Settings** > **Storage**, click the **EVS Volumes** tab, and click **Add EVS Volume**.

> NOTE
>
> You can mount EVS volumes only to workloads that contain one container.

After you create a workload, you can view the relationship between the EVS disk and the workload by choosing **Storage** > **EVS**.

Creating EVS Volumes Using kubectl

For details, see **Using PersistentVolumeClaim to Apply for Persistent Storage**.

6.3 SFS Turbo Volumes

You can mount **SFS** Turbo file systems to containers. SFS Turbo volumes are fast, on-demand, and scalable. They are suitable for DevOps, containerized microservices, and enterprise office applications.

> NOTE
>
> Only SFS Turbo General file systems are supported.

Constraints

- SFS Turbo file systems to be mounted are billed on a pay-per-use basis. For pricing details, see **SFS Turbo Billing**.

- If an SFS Turbo file system is in use, the VPC where the file system is deployed cannot be modified. If the VPC is modified, the containers in CCI will not be able to access the file system.
- If an SFS Turbo file system is deleted, containers in CCI will become unavailable.

Importing SFS Turbo File Systems

CCI allows you to import existing SFS Turbo file systems.

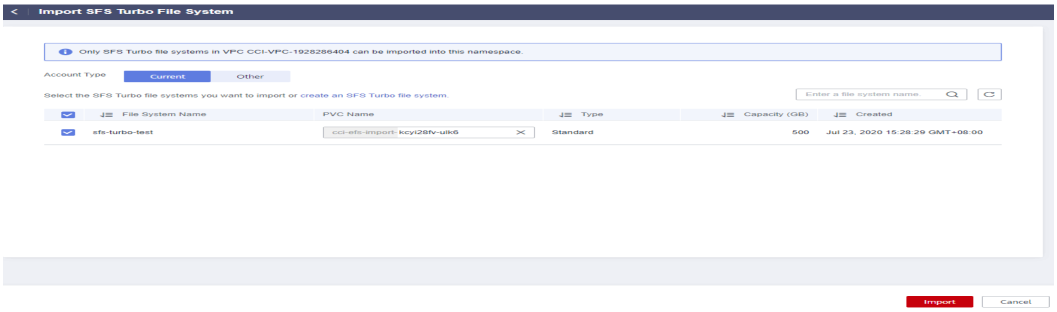
Step 1 Log in to the CCI console. In the navigation pane, choose **Storage** > **SFS Turbo**. On the page displayed, select a namespace and click **Import**.

Step 2 Select one or more SFS Turbo file systems that you want to import, and click **Import**.

If no SFS Turbo volumes are available, click **create an SFS Turbo file system** to create one.

After you create the SFS Turbo file system, go back to the **Import SFS Turbo File System** page on the CCI console. Then, select the created SFS Turbo file system and click **Import**.

Figure 6-1 Importing SFS Turbo file systems



Import SFS Turbo File System

Only SFS Turbo file systems in VPC CCI-VPC-1928286404 can be imported into this namespace.

Account Type: Current | Other

Select the SFS Turbo file systems you want to import or create an SFS Turbo file system.

☑	File System Name	PVC Name	Type	Capacity (GB)	Created
☑	sfs-turbo-test	cci-efs-import-kcyi28fv-ulk6	Standard	500	Jul 23, 2020 15:28:29 GMT+08:00

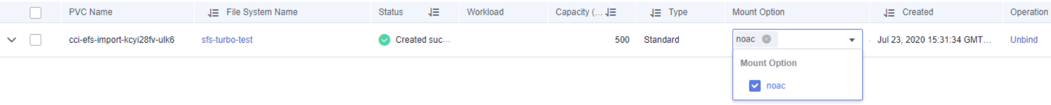
Step 3 Specify the mount option for the SFS Turbo volume to ensure real-time data access. If you mount an SFS Turbo volume to more than one pod, there will be a delay in pod metadata access due to local caching in pods.

You can set mount options for specific SFS Turbo volumes. Currently, only the **noac** mount option is supported. You can use this option to disable local file and directory caching, and allow pods to access data from the SFS Turbo volume in real time.

NOTE

The mount option is valid only for SFS Turbo volumes created in the current namespace.

Figure 6-2 Setting the mount option for an SFS Turbo volume



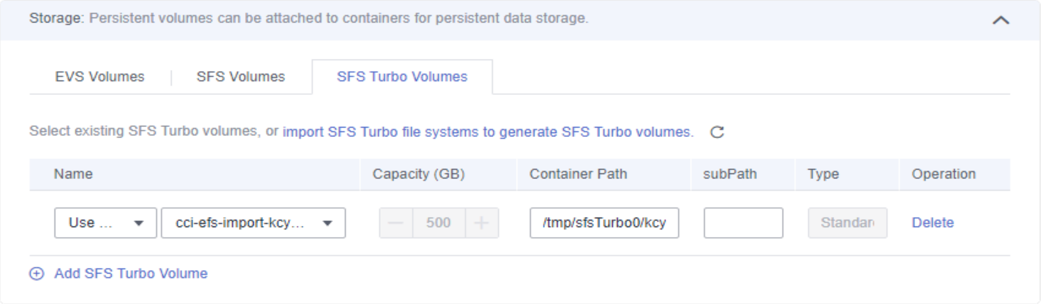
□	PVC Name	File System Name	Status	Workload	Capacity (...	Type	Mount Option	Created	Operation
□	cci-efs-import-kcyi28fv-ulk6	sfs-turbo-test	Created suc...		500	Standard	noac	Jul 23, 2020 15:31:34 GMT...	Unbind

----End

Using SFS Turbo Volumes

After you select a container image when you **create a Deployment** or **create a job**, expand **Advanced Settings** > **Storage**, click the **SFS Turbo Volumes** tab, and click **Add SFS Turbo Volume**.

Figure 6-3 Adding an SFS Turbo volume



Storage: Persistent volumes can be attached to containers for persistent data storage.

Select existing SFS Turbo volumes, or import SFS Turbo file systems to generate SFS Turbo volumes.

Name	Capacity (GB)	Container Path	subPath	Type	Operation
Use ... cci-efs-import-kcy...	500	/tmp/sfsTurbo0/kcy		Standar	Delete

NOTE

- When an SFS Turbo file system is being created, an independent VM will be created, which will take a long time. Therefore, you are advised to select existing SFS Turbo volumes.
- **subPath** is a sub-directory in the root path of the SFS Turbo file system. If such a sub-directory does not exist, it is automatically created in the SFS Turbo file system. Note that **subPath** must be a relative path.

Unbinding SFS Turbo Volumes

If you no longer require an imported SFS Turbo volume, you can unbind it from the SFS Turbo file system. After you unbind an SFS Turbo file system, you can no longer use it for your workloads.

NOTE

If you have mounted an SFS Turbo volume to a workload, you cannot unbind it from the SFS Turbo file system.

Step 1 Log in to the CCI console. In the navigation pane, choose **Storage** > **SFS Turbo**. In the SFS Turbo volume list, click **Unbind** next to the target volume.

Step 2 Read the message that is displayed and click **Yes**.

----End

7 Configuration Management

7.1 ConfigMaps

ConfigMaps are Kubernetes objects that you can use to store the configurations required by applications. After you create a ConfigMap, you can use it as a file in a containerized application.

Creating ConfigMaps

Step 1 Log in to the CCI console. In the navigation pane, choose **Configuration Center** > **ConfigMaps**. On the page displayed, select a namespace and click **Create ConfigMap**.

You can also use the YAML file to create a ConfigMap. Click **Create from YAML** in the upper right corner of the CCI console, enter the YAML definition for the ConfigMap, and click **OK**. For details about the YAML definition, see **YAML format**.

Step 2 Select a creation mode. CCI allows you to create a ConfigMap by manually specifying parameters or uploading a file.

- Method 1: manually specifying parameters. Configure parameters based on the description in **Table 7-1**. Parameters marked with an asterisk (*) are mandatory.

Table 7-1 Parameter description

Parameter	Description
Basic information	
* Name	Name of the ConfigMap. Enter 1 to 253 characters starting and ending with a letter or digit. Only lowercase letters, digits, hyphens (-), and periods (.) are allowed. Do not enter two consecutive periods or a period adjacent to a hyphen.
Description	Description of the ConfigMap.

Parameter	Description
Data	Configuration data to be stored in the ConfigMap. **Key** indicates the file name and **Value** indicates the file content. 1. Click **Add Data**. 2. Enter a key and a value.
Label	Labels are attached to various objects (such as workloads and services) in the form of key-value pairs. Labels define the identifiable properties of these objects and are used to manage and select them. 1. Click **Add Label**. 2. Enter a key and a value.

- Method 2: uploading a file.

  NOTE

  Ensure that the file is in JSON or YAML format and the file size is less than 1 MB. For details, see **ConfigMap File Format**.

  Click **Add File**, select an existing ConfigMap resource file, and click **Open**.

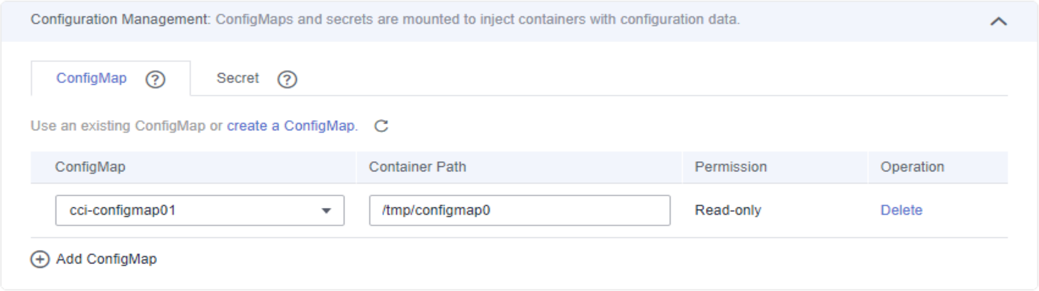
Step 3 Click **Create**.

----End

Using ConfigMaps

After you create a ConfigMap, mount it to the specified directory of a container during workload creation. For example, mount ConfigMap **cci-configmap01** to the **/tmp/configmap1** directory, as shown in the following figure.

Figure 7-1 Using a ConfigMap



Configuration Management: ConfigMaps and secrets are mounted to inject containers with configuration data.

Use an existing ConfigMap or create a ConfigMap.

ConfigMap	Container Path	Permission	Operation
cci-configmap01	/tmp/configmap0	Read-only	Delete

After you create the workload, a ConfigMap file will be created under **/tmp/configmap1**. The key of the ConfigMap indicates the file name, and the value indicates the file content.

ConfigMap File Format

A ConfigMap resource file must be in JSON or YAML format, and the file size cannot exceed 1 MB.

- JSON format

  An example of the **configmap.json** file is as follows:

```
{
  "kind": "ConfigMap",
  "apiVersion": "v1",
  "metadata": {
    "name": "nginxconf",
    "namespace": "cci-namespace-demo"
  },
  "data": {
    "nginx.conf": "server {\n    listen       80;\n    server_name  localhost;\n\n    location / {\n        root
html;\n        index  index.html index.htm;\n    }\n}"
  }
}
```

- YAML format

  An example of the **configmap.yaml** file is as follows:

```
kind: ConfigMap
apiVersion: v1
metadata:
  name: nginxconf
  namespace: cci-namespace-demo
data:
  nginx.conf: |-
    server {
        listen       80;
        server_name  localhost;

        location / {
            root   html;
            index  index.html index.htm;
        }
    }
```

Creating a ConfigMap Using kubectl

For details, see **ConfigMap**.

7.2 Secrets

Secrets are Kubernetes objects that you can use to store sensitive data such as passwords, tokens, certificates, and private keys. You can load a secret to a container as an environment variable or a file when the container is started.

NOTE

- Secrets and SSL certificates share the same quota.
- You are advised to encrypt the uploaded secret.

Creating Secrets

Step 1 Log in to the CCI console. In the navigation pane, choose **Configuration Center** > **Secrets**. On the page displayed, select a namespace and click **Create Secret**.

Step 2 Select a creation mode. CCI allows you to create a secret by manually specifying parameters or uploading a file.

- Method 1: manually specifying parameters. Configure parameters based on the description in **Table 7-2**. Parameters marked with an asterisk (*) are mandatory.

Table 7-2 Parameter description

Parameter	Description
Basic information	
* Name	Name of the secret. Enter 1 to 253 characters starting and ending with a letter or digit. Only lowercase letters, digits, hyphens (-), and periods (.) are allowed. Do not enter two consecutive periods or a period adjacent to a hyphen.
Description	Description of the secret.
* Data	Secret data that you want to use in the container. **Key** indicates the file name and **Value** indicates the file content. 1. Click **Add Data**. 2. Enter a key and a value. If you select **Auto transcoding**, the value you entered will be automatically encoded using Base64.
Label	Labels that you want to attach to various objects (such as applications, nodes, and services) in the form of key-value pairs. Labels define the identifiable properties of these objects and are used to manage and select them. 1. Click **Add Label**. 2. Enter a key and a value.

- Method 2: uploading a file.

NOTE

Ensure that the file is in JSON or YAML format and the file size is less than 2 MB. For details, see **Secret File Format**.

Click **Add File**, select an existing secret resource file, and click **Open**.

Step 3 Click **Create**.

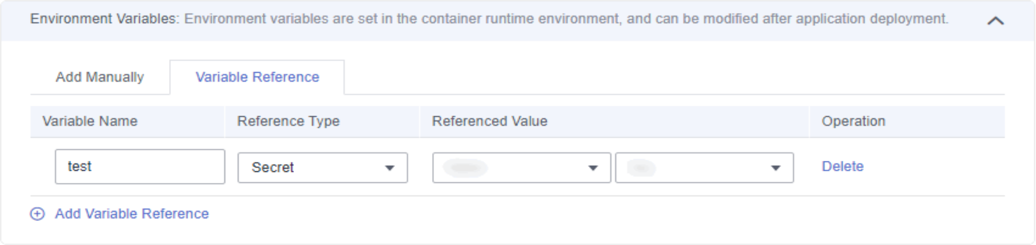
The newly created secret is displayed in the secret list.

----End

Using Secrets

After you create a secret, you can reference it as an environment variable or mount it to a container path during workload creation.

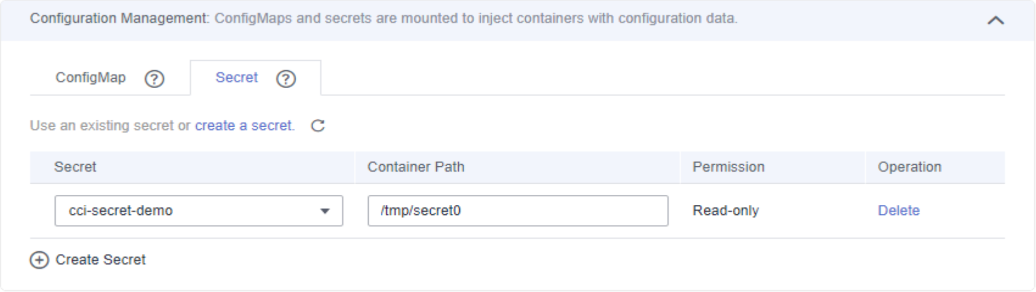
Figure 7-2 Referencing a secret as an environment variable



Environment Variables: Environment variables are set in the container runtime environment, and can be modified after application deployment.

Variable Name	Reference Type	Referenced Value	Operation
test	Secret		Delete

Figure 7-3 Mounting a secret to a container path



Configuration Management: ConfigMaps and secrets are mounted to inject containers with configuration data.

Use an existing secret or create a secret.

Secret	Container Path	Permission	Operation
cci-secret-demo	/tmp/secret0	Read-only	Delete

Secret File Format

- **secret.yaml** resource description file

  For example, to obtain the following key-value pairs and encrypt them for an application, you can use the secret.

  key1: value1

  key2: value2

  The content in the secret file **secret.yaml** is as follows. Base64 encoding is required for the value. For details about the Base64 encoding method, see **Base64 Encoding**.

```
apiVersion: v1
kind: Secret
metadata:
  name: mysecret           #Secret name
  annotations:
    description: "test"
  labels:
    label-01: value-01
    label-02: value-02
data:
key1: dmFsdWUx    #Base64 encoding required
key2: dmFsdWUy  #Base64 encoding required
type: Opaque         #Must be Opaque
```

- **secret.json** resource description file

  The content in the secret file **secret.json** is as follows:

```
{
    "apiVersion": "v1",
    "kind": "Secret",
    "metadata": {
        "annotations": {
            "description": "test"
        },
        "labels": {
            "label-01": "value-01",
            "label-02": "value-02"
        },
        "name": "mysecret"
    },
    "data": {
        "key1": "dmFsdWUx",
        "key2": "dmFsdWUy"
    },
    "type": "Opaque"
}
```

Base64 Encoding

To perform Base64 encoding on a character string, run the **echo -n** *Content to be encoded* **| base64** command. The following is an example:

```
root@ubuntu:~# echo -n "3306" | base64
MzMwNg==
```

Creating a Secret Using kubectl

For details, see **Secret**.

7.3 SSL Certificates

Secure Sockets Layer (SSL) is a security protocol designed to protect security and data integrity for Internet communications.

You can upload an SSL certificate to CCI. In HTTPS access, CCI will automatically install it to the Layer-7 load balancer for data transmission encryption.

NOTE

- Secrets and SSL certificates share the same quota.
- You are advised to encrypt the uploaded SSL certificate.

SSL Certificate Introduction

An SSL certificate indicates compliance with the SSL protocol. An SSL certificate is issued to a server by a trusted digital certificate authority (CA) after the CA has verified the identity of the server. SSL certificates have the functions of server authentication and data transmission encryption. After you install an SSL certificate, a server can encrypt the data transmitted between clients and the server and prevent information leakage. In addition, the SSL certificate verifies whether the websites visited by the server are authentic and reliable.

SSL certificates are classified into authoritative and self-signed certificates. Authoritative certificates are issued by CAs. You can obtain authoritative certificates from third-party certificate agents. A client trusts websites that use authoritative certificates by default. Self-signed certificates are self-issued by users, typically using OpenSSL. By default, self-signed certificates are untrusted by the client. The browser will display an alarm message when you access a website that uses a self-signed certificate. You can continue to access the website by ignoring the alarm.

Application Scenarios

After you install an SSL certificate, a server can encrypt the data transmitted between clients and the server and prevent information leakage. To enable secure public network access for a web application in CCI, set the workload access mode to **Internet access** and the ELB protocol to **HTTPS**, and then select the certificate for Internet access during workload creation.

Adding a Certificate

Step 1 Log in to the CCI console. In the navigation pane, choose **Configuration Center** > **SSL Certificates**. On the page displayed, select a namespace and click **Add Certificate**.

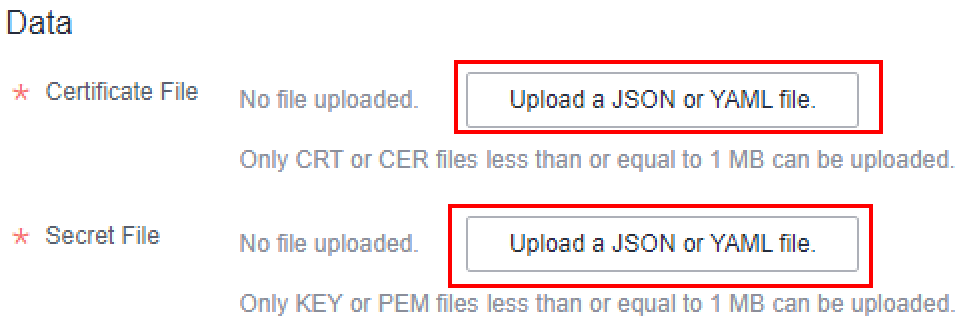
Step 2 Specify the name and description information of the SSL certificate.

Certificate name: Enter 1 to 253 characters starting and ending with a letter or digit. Only lowercase letters, digits, hyphens (-), and periods (.) are allowed. Do not enter two consecutive periods or a period adjacent to a hyphen.

Step 3 Upload the certificate file and private key file.

- **.crt** and **.cer** certificate files are supported, and the file size cannot exceed 1 MB. Ensure that the file content complies with the corresponding CRT or CER protocol.
- **.key** and **.pem** private key files are supported, and the file size cannot exceed 1 MB. You cannot encrypt private keys.

Figure 7-4 Uploading SSL certificate files



Data

* Certificate File No file uploaded. Upload a JSON or YAML file.

Only CRT or CER files less than or equal to 1 MB can be uploaded.

* Secret File No file uploaded. Upload a JSON or YAML file.

Only KEY or PEM files less than or equal to 1 MB can be uploaded.

Step 4 Click **Add**.

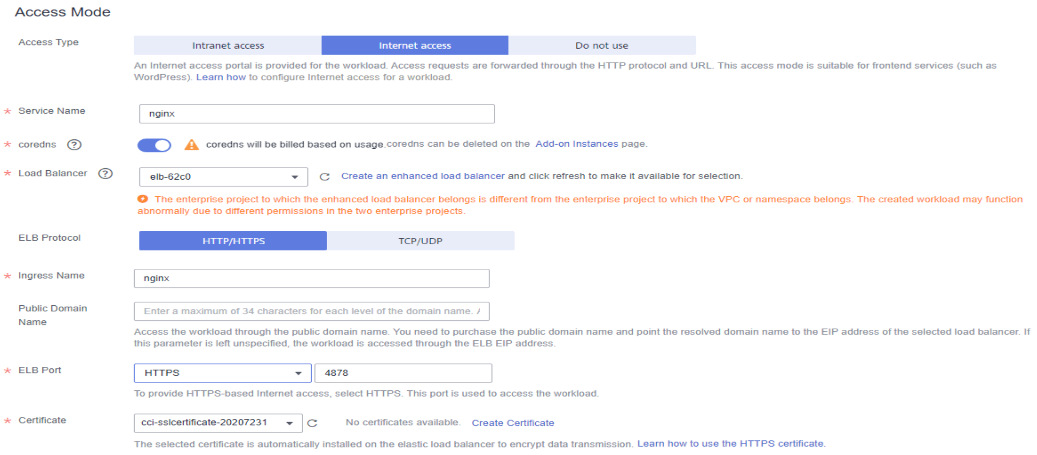
----End

Using an SSL Certificate

When the service has public network access, you can use the SSL certificate and set the ELB protocol to the HTTPS protocol.

During **workload creation**, set the workload access mode to **Internet access** and the ELB protocol to **HTTP/HTTPS**, and select the SSL certificate. The SSL certificate will be automatically installed on the ELB to encrypt data before it is transmitted.

Figure 7-5 Using an SSL certificate



Access Mode

Access Type: Intranet access | **Internet access** | Do not use

An Internet access portal is provided for the workload. Access requests are forwarded through the HTTP protocol and URL. This access mode is suitable for frontend services (such as WordPress). Learn how to configure Internet access for a workload.

* Service Name: nginx

* coredns: coredns will be billed based on usage.coredns can be deleted on the Add-on Instances page.

* Load Balancer: elb-62c0 — Create an enhanced load balancer and click refresh to make it available for selection.

The enterprise project to which the enhanced load balancer belongs is different from the enterprise project to which the VPC or namespace belongs. The created workload may function abnormally due to different permissions in the two enterprise projects.

ELB Protocol: **HTTP/HTTPS** | TCP/UDP

* Ingress Name: nginx

Public Domain Name: Enter a maximum of 34 characters for each level of the domain name. /

Access the workload through the public domain name. You need to purchase the public domain name and point the resolved domain name to the EIP address of the selected load balancer. If this parameter is left unspecified, the workload is accessed through the ELB EIP address.

* ELB Port: HTTPS 4878

To provide HTTPS-based Internet access, select HTTPS. This port is used to access the workload.

* Certificate: cci-sslcertificate-20207231 — No certificates available. Create Certificate

The selected certificate is automatically installed on the elastic load balancer to encrypt data transmission. Learn how to use the HTTPS certificate.

After you create the workload, CCI will create a certificate for the load balancer and name the certificate after the workload. If a certificate with a name starting with **beethoveen-cci-ingress** is created on CCI, do not delete or update it. Otherwise, an access exception may occur.

Updating and Deleting an SSL Certificate

- You can update a certificate before it expires. The workload that uses the certificate will also update it at the same time.
- Do not delete a certificate that is being used by a workload. Otherwise, the workload may become inaccessible.

8 Log Management

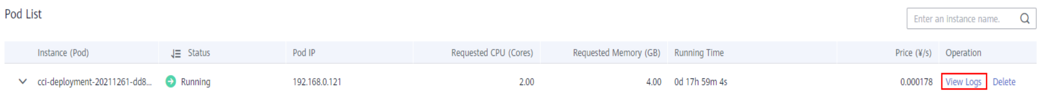
CCI allows you to mount a log storage volume for log collection. To write logs to the log storage volume, you only need to add the log storage volume when you **create a workload**.

If the service performance does not meet the expectation, it may be caused by excessive logging. For details, see **Why Service Performance Does Not Meet the Expectation?**

CCI is interconnected with **Application Operations Management (AOM)**. AOM collects the **.log** files in container log storage and dumps them in AOM.

You can click **View Logs** in the **Pod List** area on the details page of a workload to view logs.

Figure 8-1 Viewing logs



Pod List

Enter an instance name.

Instance (Pod)	Status	Pod IP	Requested CPU (Cores)	Requested Memory (GB)	Running Time	Price (¥/s)	Operation
cci-deployment-20211261-dd8...	Running	192.168.0.121	2.00	4.00	0d 17h 59m 4s	0.000178	View Logs Delete

Adding a Log Storage Volume

You can add a log storage volume for a container when you **create a workload**.

- **Container Log Path**: path to which the log storage volume is attached. Ensure that it matches the log output path of the application so that logs can be written to the log storage volume.

  NOTICE

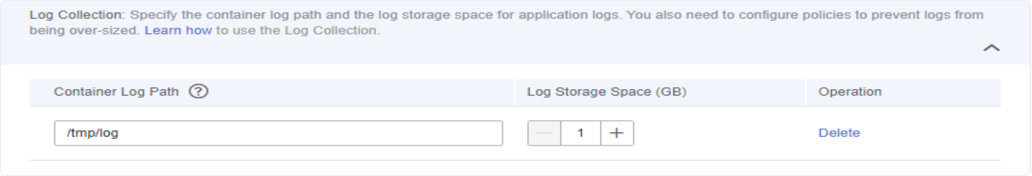
  1. Ensure that the log storage volume path does not exist in the current container. Otherwise, the content under the existing path will be cleared.
  2. Only **.log**, **.trace**, and **.out** files in the log path are collected.
  3. A maximum of 20 log files can be collected. Therefore, your logs can be exported to a maximum of 20 files in the log path.

- **Log Storage Space**: space of storing logs.

NOTICE

1. AOM provides each account 500 MB log storage space for free each month. You will be billed for any extra space you use on a pay-per-use basis. For details, see **Product Pricing Details**.
2. **Log Storage Space** can only be set to **1** or **2**. When the backend calls the API to create a workload, ensure the obtained value is 1 GiB or 2 GiB.
3. The space is free of charge. The collection will fail in the case of timeout. Therefore, for a log file larger than 2 GB, dump it in advance.

Figure 8-2 Using the log storage volume



Log Collection: Specify the container log path and the log storage space for application logs. You also need to configure policies to prevent logs from being over-sized. Learn how to use the Log Collection.

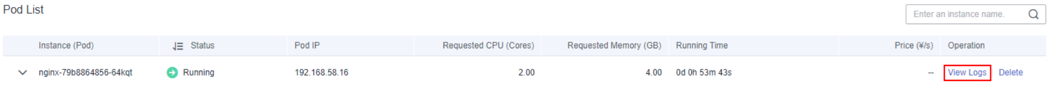
Container Log Path	Log Storage Space (GB)	Operation
/tmp/log	1	Delete

Viewing Logs

After you create the workload, you can view container logs.

Click the workload, and click **View Logs** in the same row as the pod.

Figure 8-3 Viewing logs



Pod List

Enter an instance name.

Instance (Pod)	Status	Pod IP	Requested CPU (Cores)	Requested Memory (GB)	Running Time	Price (¥/s)	Operation
nginx-79b8864856-64kqt	Running	192.168.58.16	2.00	4.00	0d 0h 53m 43s	--	View Logs Delete

You can view the logs of a container on the AOM console. For details about how to view logs in AOM, see **Viewing Log Files**.

9 Monitoring Management

CCI works with AOM to monitor pod resources, such as CPUs, memory, and disks. You can view the monitoring metric data of a pod on the CCI or AOM console.

Monitoring Metrics

You can view the metrics of container pods on the AOM console. For details, see **Table 9-1**.

Table 9-1 Monitoring metrics

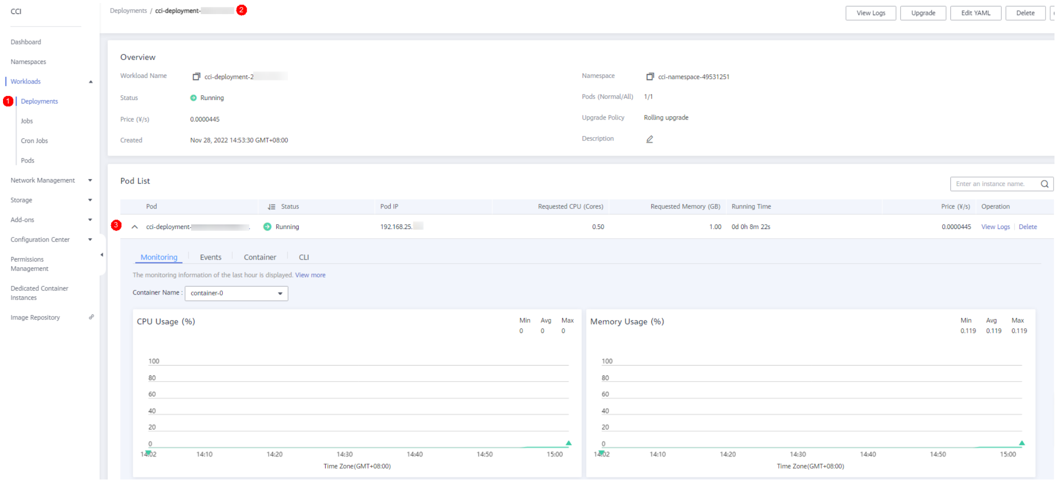
Metric ID	Name	Description	Value Range	Unit
cpuUsage	CPU usage	Percentage of the used CPU cores to the total CPU cores restricted for a measured object	0%–100%	%
cpuCoreLimit	Total CPU cores	Total number of CPU cores restricted for a measured object	≥ 1	Cores
cpuCoreUsed	Used CPU cores	Number of CPU cores used by a measured object	≥ 0	Cores
memCapacity	Total physical memory	Total physical memory restricted for a measured object	≥ 0	MB
memUsage	Physical memory usage	Percentage of the used physical memory to the total physical memory restricted for a measured object	0%–100%	%
memUsed	Used physical memory	Used physical memory of a measured object	≥ 0	MB

Viewing Pod Monitoring Data

You can view pod monitoring data on the CCI console.

Log in to the CCI console. In the navigation pane, choose **Workloads** > **Deployments**. On the page displayed, click the target workload. On the page displayed, click the downward arrow of the target pod to show its **Monitoring** tab page, which displays the CPU and memory usage of the pod in the last hour.

Figure 9-1 Pod list



Deployments / cci-deployment- 2

Overview

Workload Name cci-deployment-2

Status Running

Price (¥/s) 0.0000445

Created Nov 28, 2022 14:53:30 GMT+08:00

Namespace cci-namespace-49531251

Pods (Normal/All) 1/1

Upgrade Policy Rolling upgrade

Description

Pod List

Pod	Status	Pod IP	Requested CPU (Cores)	Requested Memory (GB)	Running Time	Price (¥/s)	Operation
3 cci-deployment-.	Running	192.168.25.	0.50	1.00	0d 0h 8m 22s	0.0000445	View Logs | Delete

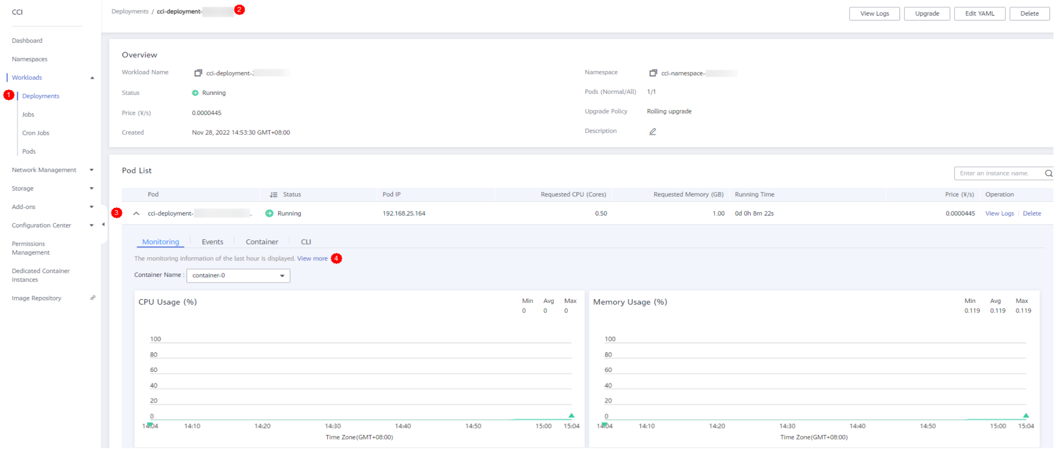
The monitoring information of the last hour is displayed. View more

Container Name : container-0

The resource monitoring information on the CCI console displays only the CPU and memory usage. You can view more monitoring metrics on the AOM console.

Step 1 Click **View more** under the **Monitoring** tab to go to the AOM console.

Figure 9-2 Pod monitoring



Deployments / cci-deployment- 2

Overview

Workload Name cci-deployment-

Status Running

Price (¥/s) 0.0000445

Created Nov 28, 2022 14:53:30 GMT+08:00

Namespace cci-namespace-

Pods (Normal/All) 1/1

Upgrade Policy Rolling upgrade

Description

Pod List

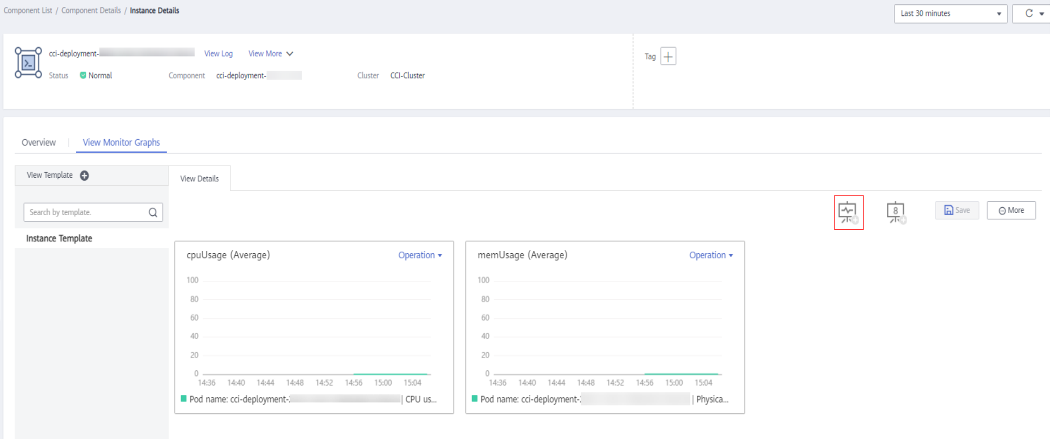
Pod	Status	Pod IP	Requested CPU (Cores)	Requested Memory (GB)	Running Time	Price (¥/s)	Operation
3 cci-deployment-.	Running	192.168.25.164	0.50	1.00	0d 0h 8m 22s	0.0000445	View Logs | Delete

The monitoring information of the last hour is displayed. View more 4

Container Name : container-0

Step 2 Click to add a line graph to the view template.

Figure 9-3 View template



Component List / Component Details / Instance Details

cci-deployment-[redacted]

Status: Normal

Component: cci-deployment-[redacted]

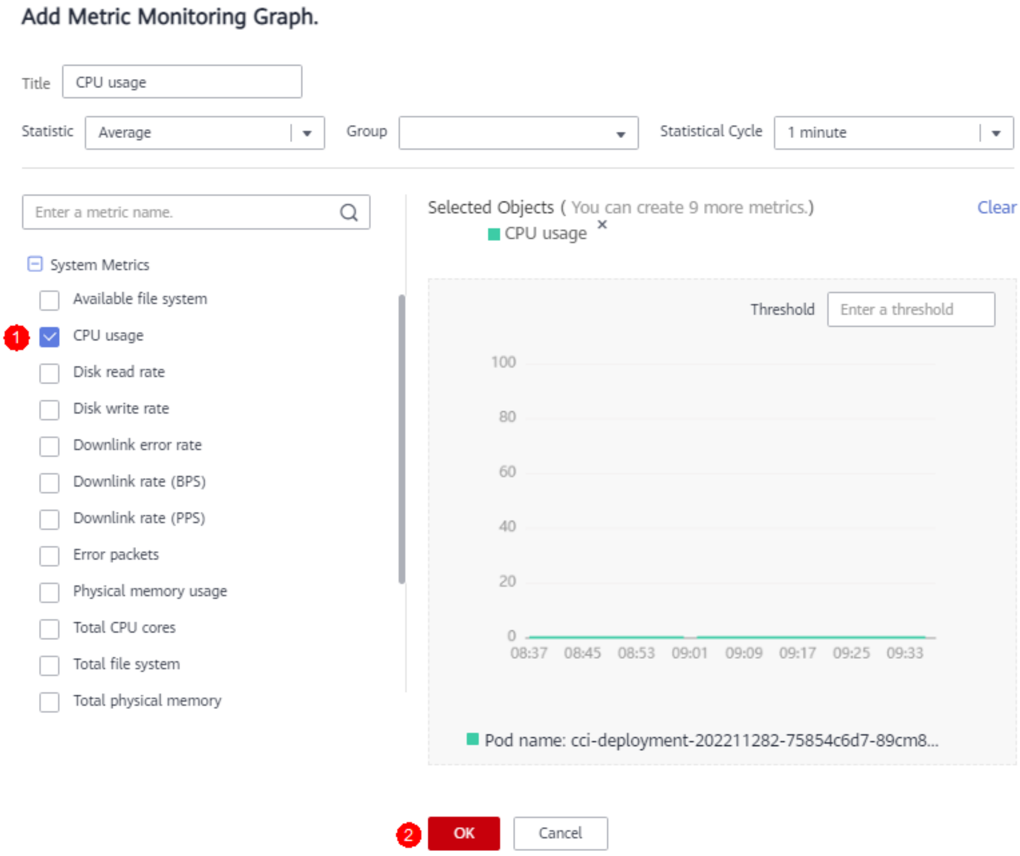
Cluster: CCI-Cluster

Pod name: cci-deployment-[redacted] | CPU us...

Pod name: cci-deployment-[redacted] | Physica...

Step 3 Select a system metric on the left of the page, for example, **CPU usage**, and click **OK**.

Figure 9-4 System metrics



Add Metric Monitoring Graph.

Title: CPU usage

Statistic: Average

Group:

Statistical Cycle: 1 minute

Selected Objects (You can create 9 more metrics.)

CPU usage

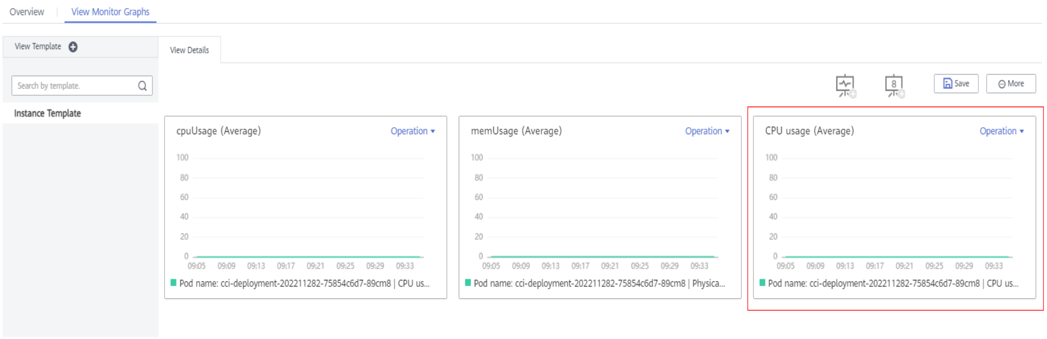
System Metrics

- ☐ Available file system
- ☑ CPU usage
- ☐ Disk read rate
- ☐ Disk write rate
- ☐ Downlink error rate
- ☐ Downlink rate (BPS)
- ☐ Downlink rate (PPS)
- ☐ Error packets
- ☐ Physical memory usage
- ☐ Total CPU cores
- ☐ Total file system
- ☐ Total physical memory

Pod name: cci-deployment-202211282-75854c6d7-89cm8...

Step 4 View the pod monitoring data on the **View Details** page.

Figure 9-5 Viewing monitor graphs



----End

10 Add-on Management

In addition to its underlying components, Kubernetes may have other components, which run as add-ons, such as Kubernetes DNS and Kubernetes Dashboard.

On the CCI console, install the coredns add-on to extend CCI features.

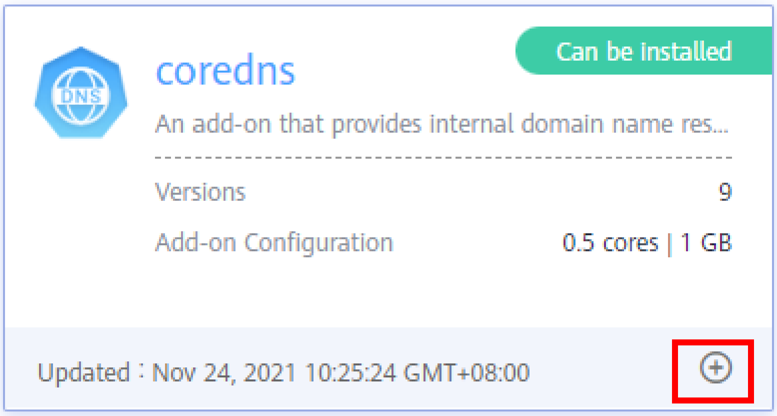
coredns

The coredns add-on provides the internal domain name resolution service for your other workloads. Do not delete or upgrade this workload; otherwise, the internal domain name resolution service will become unavailable.

Installing an Add-on

Step 1 Log in to the CCI console. In the navigation pane, choose **Add-ons** > **Add-on Marketplace**. Then, click ⊕ on the card of the add-on you want to install.

Figure 10-1 coredns add-on



coredns

Can be installed

An add-on that provides internal domain name res...

Versions 9

Add-on Configuration 0.5 cores | 1 GB

Updated：Nov 24, 2021 10:25:24 GMT+08:00

Step 2 Select a version from the **Add-on Version** drop-down list, and click **Submit**.

1. When installing coredns v2.5.9 or later, you must configure the following parameters:
   - **Stub Domain**: A DNS server that resolves user-defined domain names. The stub domain contains the suffix of the DNS domain name followed

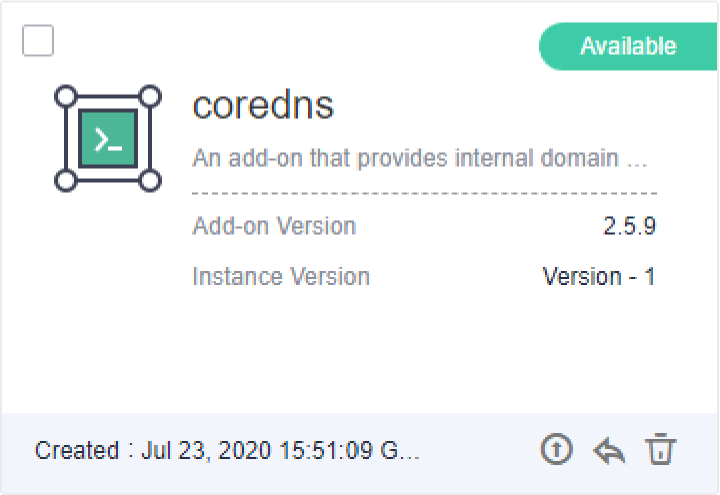
by one or more DNS IP addresses. For example, **acme.local -- 1.2.3.4,6.7.8.9** means that DNS requests with the **.acme.local** suffix are forwarded to a DNS listening at 1.2.3.4,6.7.8.9.

- **Upstream DNS Server**: A DNS server that resolves all domain names except intra-cluster service domain names and user-defined domain names. The value can be one or more DNS IP addresses, for example, **8.8.8.8,8.8.4.4**.

2. When installing coredns v2.5.10 or later, you can also configure the following parameter:
   - **Log Output**: You can select the types of domain name resolution logs to be printed based on service requirements, for example, **Success log** and **Error log**. For details, see **Configuring Log Output Options for coredns**.

After you complete the installation, the installed add-on is available at **Add-ons > Add-on Instances**.

Figure 10-2 coredns installed


Available
coredns
An add-on that provides internal domain ...
Add-on Version
2.5.9
Instance Version
Version - 1
Created : Jul 23, 2020 15:51:09 G...

----End

Configuring Stub Domains for coredns

As a cluster administrator, you can modify the ConfigMap for the CoreDNS Corefile to change how service discovery works. You can configure stub domains for coredns using the proxy plug-in.

Assume that you are a cluster administrator and you have a Consul DNS server located at 10.150.0.1 and all Consul domain names have the suffix **.consul.local**. To configure this Consul DNS server in coredns, you need to write the following information in the coredns ConfigMap:

```
consul.local:5353 {
        errors
        cache 30
        proxy . 10.150.0.1
    }
```

ConfigMap after modification:

```
apiVersion: v1
data:
  Corefile: |-
    .:5353 {
        cache 30
        errors
        health
        kubernetes cluster.local in-addr.arpa ip6.arpa {
          pods insecure
          upstream /etc/resolv.conf
          fallthrough in-addr.arpa ip6.arpa
        }
        loadbalance round_robin
        prometheus 0.0.0.0:9153
        proxy . /etc/resolv.conf
        reload
    }

    consul.local:5353 {
        errors
        cache 30
        proxy . 10.150.0.1
    }
kind: ConfigMap
metadata:
  name: coredns
  namespace: kube-system
```

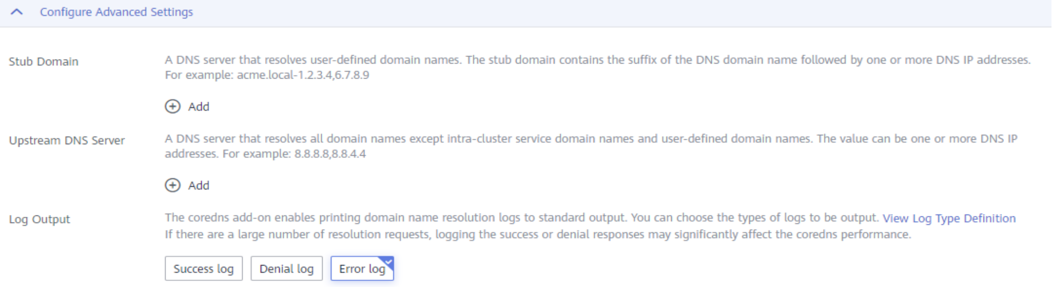
Cluster administrators can modify the ConfigMap for the CoreDNS Corefile to change how service discovery works. They can configure stub domains for coredns using the proxy plug-in.

Configuring Log Output Options for coredns

coredns uses the **log plug-in** to print the domain name resolution logs to standard output. You can configure **Log Output** to define the log content to be output, and view the **resolution logs** on the AOM console. If there is a large number of domain name resolution requests, frequent log printing may affect the coredns performance.

The log plug-in supports three log output options: **Success log**, **Denial log** and **Error log**.

Figure 10-3 Configuring options



Configure Advanced Settings

Stub Domain: A DNS server that resolves user-defined domain names. The stub domain contains the suffix of the DNS domain name followed by one or more DNS IP addresses. For example: acme.local-1.2.3.4,6.7.8.9

⊕ Add

Upstream DNS Server: A DNS server that resolves all domain names except intra-cluster service domain names and user-defined domain names. The value can be one or more DNS IP addresses. For example: 8.8.8.8,8.8.4.4

⊕ Add

Log Output: The coredns add-on enables printing domain name resolution logs to standard output. You can choose the types of logs to be output. View Log Type Definition
If there are a large number of resolution requests, logging the success or denial responses may significantly affect the coredns performance.

Success log Denial log [✓ Error log]

The backend configuration format is as follows:

```
log [NAMES...] [FORMAT] {
    class CLASSES...
}
```

NOTE

CLASSES indicates the classes of responses that should be logged. It is a list separated by spaces.

The log output options include:

- **Success log**:

  If this option is selected, the **success** response parameter is added to the **CLASSES** list of the log plug-in, and coredns prints the logs that are successfully resolved to the standard output.
- **Denial log**:

  If this option is selected, the **denial** response parameter is added to the **CLASSES** list of the log plug-in, and coredns prints the logs that fail to be resolved. For example, **NXDOMAIN** or **nodata** response (the name exists but the type does not exist) will be printed to the standard output.
- **Error log**:

  If this option is selected, the **error** response parameter is added to the **CLASSES** list of the log plug-in, and coredns prints logs about resolution errors to the standard output, for example, **SERVFAIL**, **NOTIMP**, and **REFUSED**. This helps detect problems such as DNS server unavailability in a timely manner.
- Deselect all:

  If none of the preceding options is selected, the log plug-in is disabled.

> ⚠ **CAUTION**
>
> Disabling the log plug-in takes effect only for the resolution records of coredns. The logs of the coredns service process are still displayed, which are small in volume and do not affect performance.

If **Success log** and **Error log** are selected, the backend log plug-in configuration is as follows:

```
log . {
    class success denial
}
```

The ConfigMap corresponding to the created coredns is as follows:

```
apiVersion: v1
data:
  Corefile: |-
    .:5353 {
        cache 30
        errors
        log . {
          classes success denial
        }
        health
        kubernetes cluster.local in-addr.arpa ip6.arpa {
          pods insecure
          upstream /etc/resolv.conf
          fallthrough in-addr.arpa ip6.arpa
        }
        loadbalance round_robin
        prometheus 0.0.0.0:9153
        proxy . /etc/resolv.conf
        reload
    }
kind: ConfigMap
metadata:
```

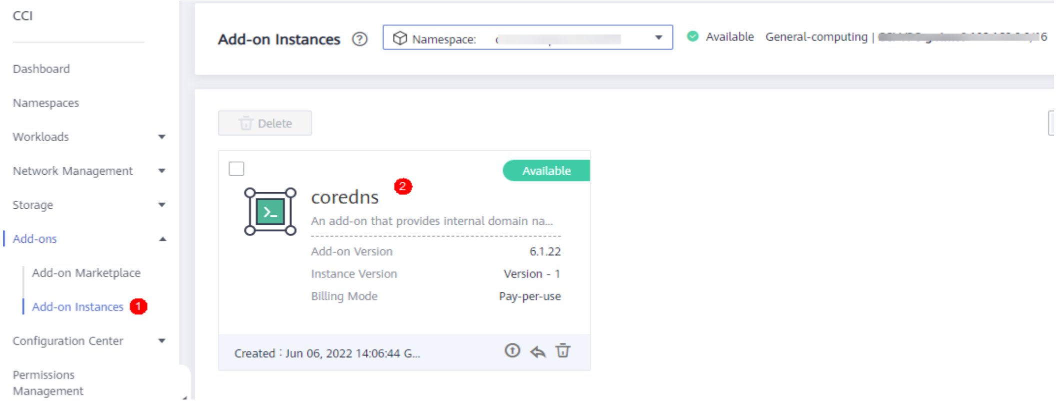
```
name: coredns
namespace: kube-system
```

Viewing Resolution Logs

After configuring the log plug-in, you can view resolution logs on the AOM console.

Step 1 Log in to the CCI console. In the navigation pane, choose **Add-ons** > **Add-on Instances**. Select coredns on the right to display the coredns page.

Figure 10-4 Add-on instances



Add-on Instances

Namespace: [redacted]

Available General-computing | [redacted]

coredns

An add-on that provides internal domain na...

Add-on Version 6.1.22

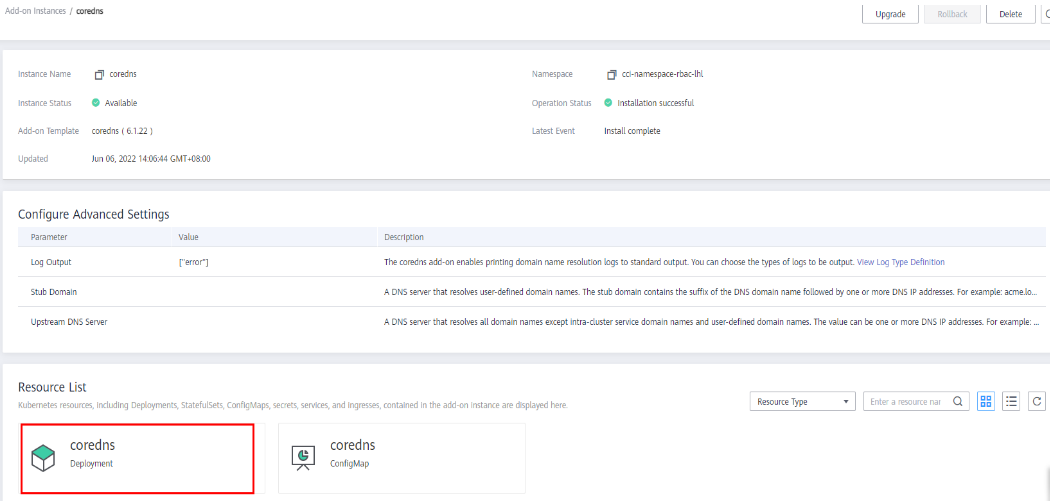
Instance Version Version - 1

Billing Mode Pay-per-use

Created : Jun 06, 2022 14:06:44 G...

Step 2 Click **coredns Deployment** in the resource list to go to the pod list.

Figure 10-5 coredns deployment



Instance Name coredns

Namespace cci-namespace-rbac-lhl

Instance Status Available

Operation Status Installation successful

Add-on Template coredns (6.1.22)

Latest Event Install complete

Updated Jun 06, 2022 14:06:44 GMT+08:00

Configure Advanced Settings

Parameter	Value	Description
Log Output	["error"]	The coredns add-on enables printing domain name resolution logs to standard output. You can choose the types of logs to be output. View Log Type Definition
Stub Domain		A DNS server that resolves user-defined domain names. The stub domain contains the suffix of the DNS domain name followed by one or more DNS IP addresses. For example: acme.lo...
Upstream DNS Server		A DNS server that resolves all domain names except intra-cluster service domain names and user-defined domain names. The value can be one or more DNS IP addresses. For example: ...

Resource List

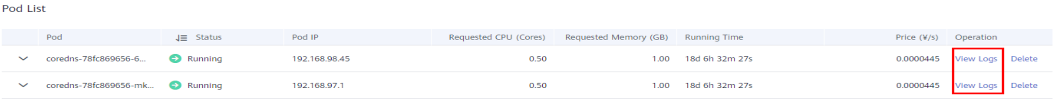
Kubernetes resources, including Deployments, StatefulSets, ConfigMaps, secrets, services, and ingresses, contained in the add-on instance are displayed here.

coredns
Deployment

coredns
ConfigMap

Step 3 Click **View Logs** in the **Operation** column in **Pod List** to access the AOM console to view the coredns logs.

Figure 10-6 Pod list



Pod List

	Pod	Status	Pod IP	Requested CPU (Cores)	Requested Memory (GB)	Running Time	Price (¥/s)	Operation
﹀	coredns-78fc869656-6...	Running	192.168.98.45	0.50	1.00	18d 6h 32m 27s	0.0000445	View Logs Delete
﹀	coredns-78fc869656-mk...	Running	192.168.97.1	0.50	1.00	18d 6h 32m 27s	0.0000445	View Logs Delete

----End

How Does Domain Name Resolution Work in Kubernetes?

DNS policies can be set on a per-pod basis. Kubernetes supports four types of DNS policies: **Default**, **ClusterFirst**, **ClusterFirstWithHostNet**, and **None**. For details, see **DNS for Services and Pods**. These policies are specified in the **dnsPolicy** field in the pod-specific.

- **Default**: Pods inherit the name resolution configuration from the node that runs the pods. The custom upstream DNS server and the stub domain cannot be used together with this policy.
- **ClusterFirst**: Any DNS query that does not match the configured cluster domain suffix, such as **www.kubernetes.io**, is forwarded to the upstream name server inherited from the node. Cluster administrators may have extra stub domains and upstream DNS servers configured.
- **ClusterFirstWithHostNet**: For pods running with hostNetwork, set its DNS policy **ClusterFirstWithHostNet**.
- **None**: It allows a pod to ignore DNS settings from the Kubernetes environment. All DNS settings are supposed to be provided using the **dnsPolicy** field in the pod-specific.

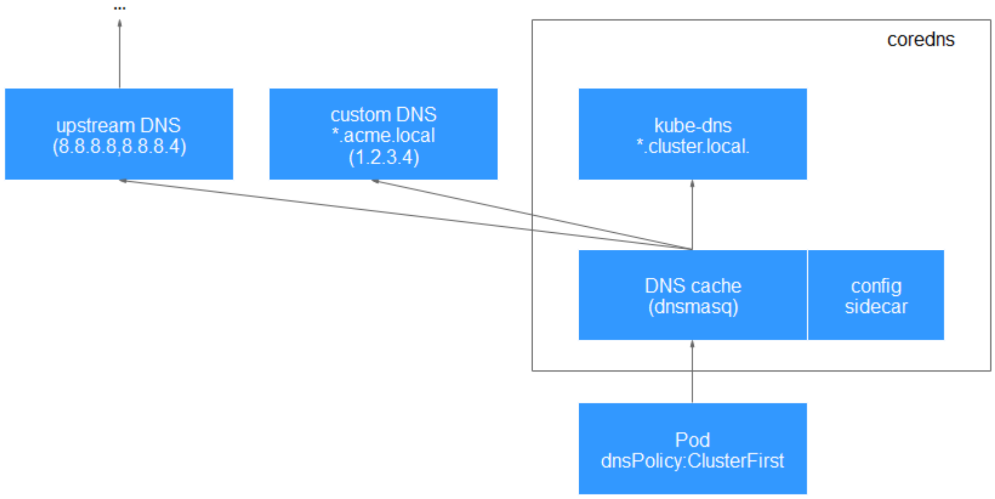
NOTE

- Clusters of Kubernetes v1.10 and later support **Default**, **ClusterFirst**, **ClusterFirstWithHostNet**, and **None**. Clusters earlier than Kubernetes v1.10 support only **Default**, **ClusterFirst**, and **ClusterFirstWithHostNet**.
- **Default** is not the default DNS policy. If **dnsPolicy** is not explicitly specified, **ClusterFirst** is used.

Routing

- Without stub domain configurations: Any query that does not match the configured cluster domain suffix, such as **www.kubernetes.io**, is forwarded to the upstream DNS server inherited from the node.
- With stub domain configurations: If you have configured stub domains and upstream DNS servers, DNS queries are routed according to the following flow:
  a. The query is first sent to the DNS caching layer in coredns.
  b. From the caching layer, the suffix of the request is examined and then forwarded to the appropriate DNS, based on the following cases:
     - Names with the cluster suffix, for example, **.cluster.local**: The request is sent to coredns.
     - Names with the stub domain suffix, for example, **.acme.local**: The request is sent to the configured custom DNS resolver, listening for example at 1.2.3.4.
     - Names that do not match the suffix (for example, **widget.com**): The request is forwarded to the upstream DNS.

Figure 10-7 Routing



Follow-Up Operations

After the add-on is installed, you can perform the following operations on the add-on.

Table 10-1 Other operations

Operation	Description
Upgrade	Click ⬆. Select the target version, and click **Next**. Then, confirm the new configuration information, and click **Submit**.
Rollback	Click ↩. Then, select the version to which the add-on is to be rolled back, and click **Submit**.
Deletion	Click 🗑 and then click **Yes**. **NOTICE** Deleted add-ons cannot be recovered. Exercise caution when performing this operation.

11 Auditing

11.1 CCI Operations Supported by CTS

Cloud Trace Service (CTS) records operations on cloud service resources, allowing you to query, audit, and backtrack the resource operation requests initiated from the CCI console or open APIs as well as responses to the requests.

Table 11-1 CCI operations that can be recorded by CTS

Operation	Trace Name
Creating a Service	createService
Deleting a Service	deleteService
Deleting all Services under a specified namespace	deleteServicesByNamespace
Replacing a Service	replaceService
Updating a Service	updateService
Deleting an Endpoints object	deleteEndpoint
Deleting all Endpoints objects under a specified namespace	deleteEndpointsByNamespace
Replacing an Endpoints object under a specified namespace	replaceEndpoint
Updating an Endpoints object under a specified namespace	updateEndpoint
Creating a Deployment	createDeployment
Deleting a Deployment	deleteDeployment
Deleting all Deployments under a specified namespace	deleteDeploymentsByNamespace

Operation	Trace Name
Replacing a Deployment under a specified namespace	replaceDeployment
Updating a Deployment under a specified namespace	updateDeployment
Creating a StatefulSet	createStatefulset
Deleting a StatefulSet	deleteStatefulset
Deleting all StatefulSets under a specified namespace	deleteStatefulsetsByNamespace
Replacing a StatefulSet under a specified namespace	replaceStatefulset
Updating a StatefulSet under a specified namespace	updateStatefulset
Creating a job	createJob
Deleting a job	deleteJob
Deleting all jobs under a specified namespace	deleteJobsByNamespace
Replacing the status of a job under a specified namespace	replaceJob
Updating the status of a job under a specified namespace	updateJob
Creating a cron job	createCronjob
Deleting a cron job	deleteCronjob
Deleting all cron jobs under a specified namespace	deleteCronjobsByNamespace
Replacing the status of a cron job under a specified namespace	replaceCronjob
Updating the status of a cron job under a specified namespace	updateCronjob
Creating an ingress	createIngress
Deleting an ingress	deleteIngress
Deleting all ingresses under a specified namespace	deleteIngressesByNamespace
Replacing an ingress under a specified namespace	replaceIngress
Updating the status of an ingress under a specified namespace	updateIngress

Operation	Trace Name
Creating a namespace	createNamespace
Deleting a namespace	deleteNamespace
Creating a pod	createPod
Updating a pod	updatePod
Replacing a pod	replacePod
Deleting a pod	deletePod
Deleting all pods under a specified namespace	deletePodsByNamespace
Deleting an event	deleteEvent
Creating a ConfigMap	createConfigmap
Updating a ConfigMap	updateConfigmap
Replacing a ConfigMap	replaceConfigmap
Deleting a ConfigMap	deleteConfigmap
Deleting all ConfigMaps under a specified namespace	deleteConfigmapsByNamespace
Creating a secret	createSecret
Updating a secret	updateSecret
Replacing a secret	replaceSecret
Deleting a secret	deleteSecret
Deleting all secrets under a specified namespace	deleteSecretsByNamespace
Deleting a network	deleteNetwork
Creating a network	createNetwork
Deleting all networks under a specified namespace	deleteNetworksByNamespace
Updating a network	updateNetwork
Replacing a network	replaceNetwork
Creating a network attachment definition	createNetworkAttachmentDefinition
Deleting all network attachment definitions under a specified namespace	deleteNetworkAttachmentDefinitionsByNamespace
Deleting a network attachment definition	deleteNetworkAttachmentDefinition

Operation	Trace Name
Creating a PV	createPersistentvolume
Deleting all PVs under a specified namespace	deletePersistentvolumesByNamespace
Replacing a PV	replacePersistentvolume
Updating a PV	updatePersistentvolume
Deleting a PV	deletePersistentvolume
Creating a PVC	createPersistentvolumeclaim
Importing an existing PVC	createPersistentvolumeclaimByStorageInfo
Deleting all PVCs under a specified namespace	deletePersistentvolumeclaimsByNamespace
Replacing a PVC	replacePersistentvolumeclaim
Updating a PVC	updatePersistentvolumeclaim
Deleting a PVC	deletePersistentvolumeclaim
Creating a Kubeflow job	createKubeflowJob
Deleting all Kubeflow jobs under a specified namespace	deleteKubeflowJobsByNamespace
Replacing a Kubeflow job	replaceKubeflowJob
Updating a Kubeflow job	updateKubeflowJob
Deleting a Kubeflow job	deleteKubeflowJob
Creating a Volcano job	createVolcanoJob
Deleting all Volcano jobs under a specified namespace	deleteVolcanoJobsByNamespace
Replacing a Volcano job	replaceVolcanoJob
Updating a Volcano job	updateVolcanoJob
Deleting a Volcano job	deleteVolcanoJob
Creating an agency	createAgency
Updating a quota	modifyQuota
Creating an ImageCache	createImagecache
Deleting an ImageCache	deleteImagecache
Replacing an ImageCache	replaceImagecache
Updating an ImageCache	updateImagecache

Operation	Trace Name
Uploading a chart	createChart
Updating a chart	updateChart
Deleting a chart	deleteChart
Uploading an add-on	createAddon
Updating an add-on	updateAddon
Deleting an add-on	deleteAddon
Creating a release	createRelease
Updating a release	updateRelease
Deleting a release	deleteRelease
Creating an add-on instance	createAddonInstance
Updating an add-on instance	updateAddonInstance
Deleting an add-on instance	deleteAddonInstance
Creating an add-on readme	createAddonReadme
Deleting an add-on readme	deleteAddonReadme

11.2 Viewing Logs in CTS

Scenarios

You can view operation records of the last seven days on the CTS console.

Procedure

Step 1 Log in to the management console.

Step 2 Click 📍 in the upper left corner to select the desired region and project.

Step 3 Click **Service List**, and choose **Management & Governance** > **Cloud Trace Service**.

Step 4 In the navigation pane, choose **Trace List**.

Step 5 Specify the filters used for querying traces. The following filters are available:

- **Trace Type**, **Trace Source**, **Resource Type**, and **Search By**

  Select the desired filter criterion from the drop-down lists. Select **CCI** from the **Trace Source** drop-down list.

  If you select **Trace name** for **Search By**, you need to select a trace name.

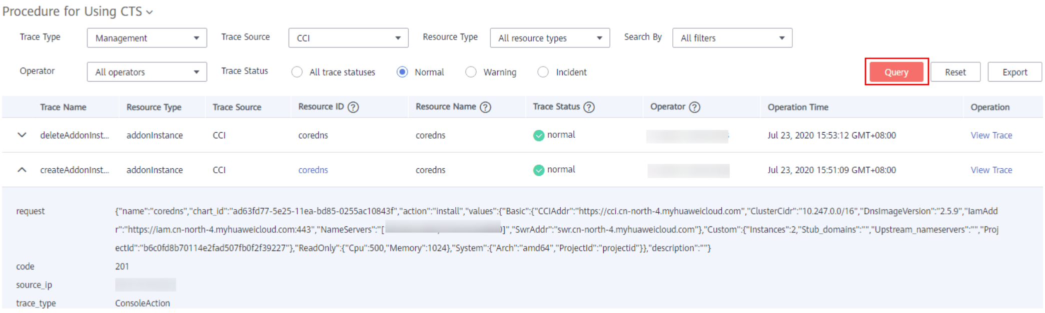
  If you select **Resource ID** for **Search By**, you need to select or enter a resource ID.

If you select **Resource name** for **Search By**, you need to select or enter a resource name.

- **Operator**: Select a specific operator (at the user level and not at the account level).
- **Trace Status**: Select one of **All trace statuses**, **Normal**, **Warning**, and **Incident**.
- **Start Date** and **End Date**: You can specify a time period to query traces.

Step 6 Click ⌄ on the left of a trace to expand its details.

Figure 11-1 Expanding trace details



Procedure for Using CTS ⌄

Trace Type: Management ▾ Trace Source: CCI ▾ Resource Type: All resource types ▾ Search By: All filters ▾

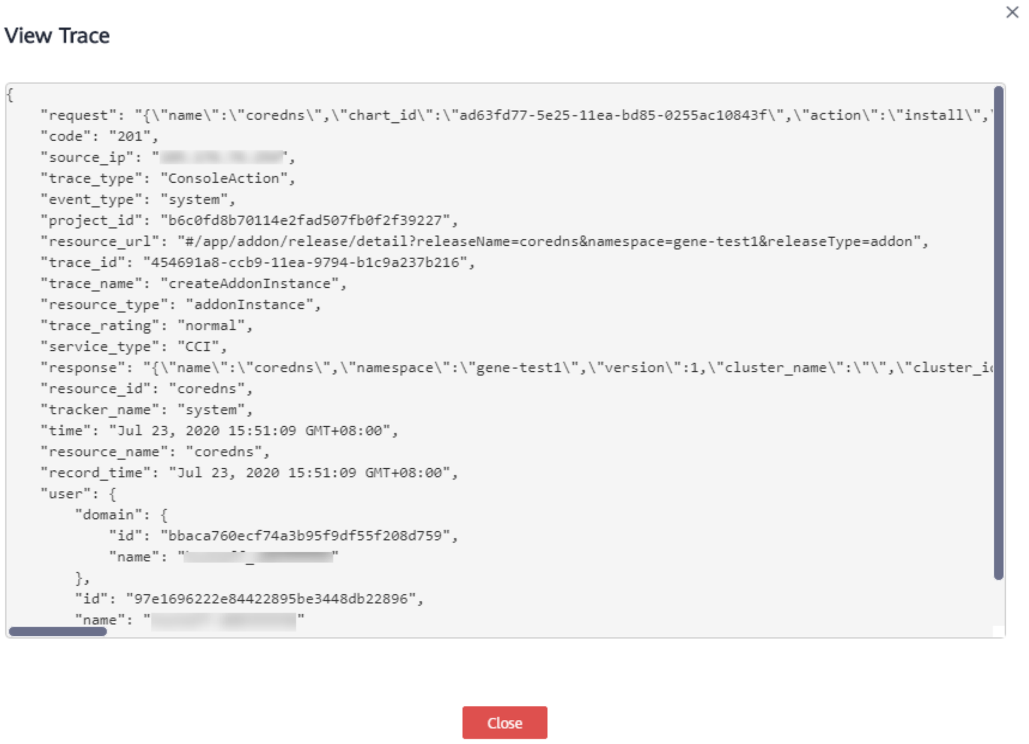
Operator: All operators ▾ Trace Status: ○ All trace statuses ● Normal ○ Warning ○ Incident

	Trace Name	Resource Type	Trace Source	Resource ID ⓘ	Resource Name ⓘ	Trace Status ⓘ	Operator ⓘ	Operation Time	Operation
⌄	deleteAddonInst...	addonInstance	CCI	coredns	coredns	✓ normal	[redacted]	Jul 23, 2020 15:53:12 GMT+08:00	View Trace
⌃	createAddonInst...	addonInstance	CCI	coredns	coredns	✓ normal	[redacted]	Jul 23, 2020 15:51:09 GMT+08:00	View Trace

```
request      {"name":"coredns","chart_id":"ad63fd77-5e25-11ea-bd85-0255ac10843f","action":"install","values":{"Basic":{"CCIAddr":"https://cci.cn-north-4.myhuaweicloud.com","ClusterCidr":"10.247.0.0/16","DnsImageVersion":"2.5.9","IamAdd
             r":"https://iam.cn-north-4.myhuaweicloud.com:443","NameServers":"[[redacted])]","SwrAddr":"swr.cn-north-4.myhuaweicloud.com"},"Custom":{"Instances":2,"Stub_domains":"","Upstream_nameservers":"","Proj
             ectId":"b6c0fd8b70114e2fad507fb0f2f39227"},"ReadOnly":{"Cpu":500,"Memory":1024},"System":{"Arch":"amd64","ProjectId":"projectid"}},"description":""}
code         201
source_ip    [redacted]
trace_type   ConsoleAction
```

Step 7 Click **View Trace** in the **Operation** column. In the displayed dialog box shown in **Figure 11-2**, the trace structure details are displayed.

Figure 11-2 Viewing trace details



View Trace

```
{
    "request": "{\"name\":\"coredns\",\"chart_id\":\"ad63fd77-5e25-11ea-bd85-0255ac10843f\",\"action\":\"install\",
    "code": "201",
    "source_ip": "[redacted]",
    "trace_type": "ConsoleAction",
    "event_type": "system",
    "project_id": "b6c0fd8b70114e2fad507fb0f2f39227",
    "resource_url": "#/app/addon/release/detail?releaseName=coredns&namespace=gene-test1&releaseType=addon",
    "trace_id": "454691a8-ccb9-11ea-9794-b1c9a237b216",
    "trace_name": "createAddonInstance",
    "resource_type": "addonInstance",
    "trace_rating": "normal",
    "service_type": "CCI",
    "response": "{\"name\":\"coredns\",\"namespace\":\"gene-test1\",\"version\":1,\"cluster_name\":\"\",\"cluster_i
    "resource_id": "coredns",
    "tracker_name": "system",
    "time": "Jul 23, 2020 15:51:09 GMT+08:00",
    "resource_name": "coredns",
    "record_time": "Jul 23, 2020 15:51:09 GMT+08:00",
    "user": {
        "domain": {
            "id": "bbaca760ecf74a3b95f9df55f208d759",
            "name": "[redacted]"
        },
        "id": "97e1696222e84422895be3448db22896",
        "name": "[redacted]"
```

----End

12 Security Vulnerability Responses

12.1 Notice on Fixing Linux Kernel SACK Vulnerabilities

- Pods that are not associated with an ELB or EIP are not affected by these vulnerabilities because they are not exposed to the public network. Therefore, no action is required.
- Deployments that were created after 00:00 on July 11 are not affected by these vulnerabilities. However, you are advised to recreate pods in the Deployments that were created before 00:00 on July 11, during off-peak hours. For details, see **Solution**.
- After existing job or cron jobs are completed, pods created by the next job or cron job will not be affected by these vulnerabilities. Therefore, no action is required.
- The coredns add-on is not affected by these vulnerabilities. Therefore, no action is required.

Vulnerability Details

On June 18, 2019, Red Hat released a security notice, stating that the TCP SACK module of the Linux kernel is exposed to three security vulnerabilities (CVE-2019-11477, CVE-2019-11478, and CVE-2019-11479). These vulnerabilities are related to the maximum segment size (MSS) and TCP Selective Acknowledgment (SACK) packets. Remote attackers can exploit these vulnerabilities to trigger a denial of service (DoS), resulting in server unavailability or breakdown.

Reference links:

https://www.suse.com/support/kb/doc/?id=7023928

https://access.redhat.com/security/vulnerabilities/tcpsack

https://www.debian.org/lts/security/2019/dla-1823

https://wiki.ubuntu.com/SecurityTeam/KnowledgeBase/SACKPanic?

https://lists.centos.org/pipermail/centos-announce/2019-June/023332.html

https://github.com/Netflix/security-bulletins/blob/master/advisories/third-party/2019-001.md

Table 12-1 Vulnerability information

Vulnerability Type	CVE-ID	Published	Fixed
Input validation flaw	**CVE-2019-11477**	2019-06-17	2019-07-11
Resource management flaw	**CVE-2019-11478**	2019-06-17	2019-07-11
Resource management flaw	**CVE-2019-11479**	2019-06-17	2019-07-11

Affected Products

Linux kernel version 2.6.29 and later

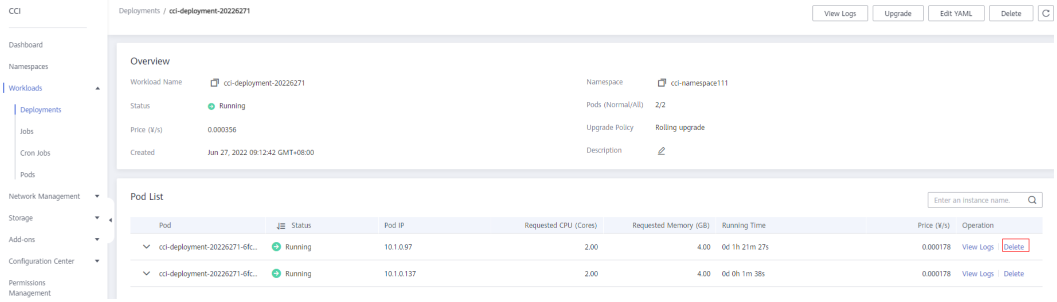
Solution

During off-peak hours, **delete and recreate pods** in the Deployments that were created before 00:00 on July 11.

Step 1 Log in to the CCI console. In the navigation pane, choose **Workloads** > **Deployments**. On the page displayed, click a Deployment name.

Step 2 In the **Pod List** area on the Deployment details page, click **Delete** in the row where the pod resides. In the dialog box that is displayed, click **Yes**.

Figure 12-1 Deleting a pod



Overview

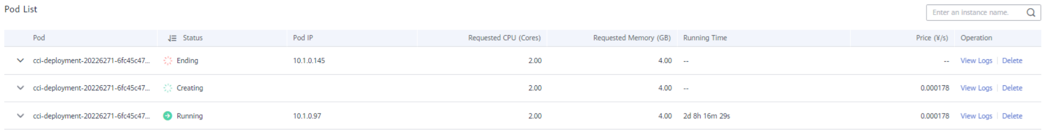
Workload Name: cci-deployment-20226271
Status: Running
Price (¥/s): 0.000356
Created: Jun 27, 2022 09:12:42 GMT+08:00
Namespace: cci-namespace111
Pods (Normal/All): 2/2
Upgrade Policy: Rolling upgrade
Description:

Pod List

Pod	Status	Pod IP	Requested CPU (Cores)	Requested Memory (GB)	Running Time	Price (¥/s)	Operation
cci-deployment-20226271-6fc...	Running	10.1.0.97	2.00	4.00	0d 1h 21m 27s	0.000178	View Logs | Delete
cci-deployment-20226271-6fc...	Running	10.1.0.137	2.00	4.00	0d 0h 1m 38s	0.000178	View Logs | Delete

After the pod is deleted, the Deployment automatically creates new pods, as shown in **Figure 12-2**.

Figure 12-2 Automatically creating pods



Pod List

Pod	Status	Pod IP	Requested CPU (Cores)	Requested Memory (GB)	Running Time	Price (¥/s)	Operation
cci-deployment-20226271-6fc45c47...	Ending	10.1.0.145	2.00	4.00	--	--	View Logs | Delete
cci-deployment-20226271-6fc45c47...	Creating		2.00	4.00	--	0.000178	View Logs | Delete
cci-deployment-20226271-6fc45c47...	Running	10.1.0.97	2.00	4.00	2d 8h 16m 29s	0.000178	View Logs | Delete

NOTICE

If there are multiple pods in a Deployment, delete them individually. That is, delete a pod only after the previous pod is successfully re-created. Otherwise, services will be affected.

----End

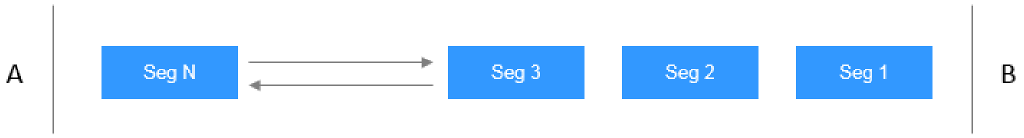
Appendix: Introduction to TCP SACKs

TCP is a connection oriented protocol. When two parties wish to communicate over a TCP connection, they establish a connection by exchanging certain information such as requesting to initiate (SYN) a connection, initial sequence number, acknowledgment number, maximum segment size (MSS) to use over this connection, and permission to send and process Selective Acknowledgements (SACKs). This connection establishment process is known as 3-way handshake.

TCP sends and receives user data by a unit called segment. A TCP segment consists of TCP Header, Options and user data. Each TCP segment has a Sequence Number (SEQ) and Acknowledgment Number (ACK).

These SEQ & ACK numbers are used to track which segments are successfully received by the receiver. ACK number indicates the next expected segment by the receiver.

Example:



User A sends 1 kilobyte of data through 13 segments of 100 bytes each. There are 13 segments in total because each segment has a TCP header of 20 bytes. On the receiving end, user B receives segments 1, 2, 4, 6, and 8-13. Segments 3, 5, and 7 are lost, and are not received by user B.

By using ACK numbers, user B will indicate that it is expecting segment 3, which user A understands as none of the segments after 2 were received by user B. Then user A will retransmit all the segments from 3 onwards, even though segments 4, 6, and 8-13 were successfully received by user B. User B has no way to indicate this to user A. This leads to an inefficient usage of the network.

12.2 CVE-2020-8558 Vulnerability Notice

The HUAWEI CLOUD CCI team fully noticed the kube-proxy security vulnerability CVE-2020-8558 on July 10. After detailed analysis, it is found that **the vulnerability has no impact on users and CCI services, and does not need to be handled**.

Vulnerability Details

Kubernetes officially released security vulnerability (CVE-2020-8558) which allows adjacent hosts to access Kubernetes nodes running on the local host.

For example, if a Kubernetes cluster runs a service on a node that listens on 127.0.0.1, because of this bug, the service will be potentially reachable by other hosts on the same LAN as the node, or by hosts on a Layer-2 network. In this way, the port information is obtained. If the example service on the port requires no additional authentication, the service is vulnerable to attacks.

Reference link:

https://github.com/kubernetes/kubernetes/issues/92315

Root Cause

This issue was originally raised by setting **net.ipv4.conf.all.route_localnet=1** for kube-proxy. This setting causes the system not to reject traffic that originates on other hosts to the local host.

How Do I Determine Whether a Vulnerability Is Involved?

- Affected cluster versions are used:
  - kubelet/kube-proxy v1.18.0-1.18.3
  - kubelet/kube-proxy v1.17.0-1.17.6
  - kubelet/kube-proxy v1.16.10 or earlier
- Your cluster nodes run in an environment where untrusted hosts share the same Layer-2 domain (for example, same LAN) as the cluster nodes.
- Your cluster allows untrusted pods to run containers with **CAP_NET_RAW** (Kubernetes clusters allow this capability by default).
- Your nodes (or pods that use the host network) run localhost services which do not require any further authentication.

For more information, see **Am I vulnerable?** in **https://github.com/kubernetes/kubernetes/issues/92315**.

Vulnerability Analysis Results

Based on the preceding analysis, **CCI is not affected by the vulnerability** because:

- CCI clusters are based on Kubernetes v1.15. However, the kube-proxy component uses self-developed code and does not involve the setting **net.ipv4.conf.all.route_localnet=1**.
- The network model of CCI clusters is different from that of common Kubernetes clusters. CCI uses secure containers and is deeply integrated with Huawei Cloud networking services. Your VPC network and the CCI host network are not in the same Layer-2 domain. **There is no information leakage risk at the CCI side.**
- By default, **net.ipv4.conf.all.route_localnet** is set to **0** in the service container kernel. The process bound to localhost cannot access other nodes in the same VPC. **There is no information leakage risk at the user side.**

12.3 CVE-2020-13401 Vulnerability Notice

The Huawei Cloud CCI team fully noticed the Kubernetes security vulnerability CVE-2020-13401 on July 22. After detailed analysis, it is found that **the vulnerability has no impact on users and CCI services, and does not need to be handled**.

Vulnerability Details

Kubernetes officially released security vulnerability CVE-2020-13401, with CVSS rating of **CVSS:3.1/AV:N/AC:H/PR:L/UI:N/S:C/C:L/I:L/A:L** (6.0 Medium).

Vulnerability brief: IPv6 address dynamic allocation can be implemented through Dynamic Host Configuration Protocol (DHCP) or Router Advertisement. This causes the CVE-2020-13401 vulnerability. Router Advertisement allows the router to periodically notify nodes of the network status, including routing records. The client configures the network through Neighbor Discovery Protocol (NDP).

A malicious attacker can tamper with the IPv6 routing records of other containers on the host or the host itself to initiate a man-in-the-middle attack. Even if there was no IPv6 traffic before, if the DNS returns A (IPv4) and AAAA (IPv6) records, many HTTP libraries will try to use the IPv6 record for connections first then fall back to the IPv4 record, giving an opportunity to the attacker to respond.

Reference link: **https://github.com/kubernetes/kubernetes/issues/91507**

How Do I Determine Whether a Vulnerability Is Involved?

Kubernetes is not affected by this vulnerability. However, the CNI plug-in (see **containernetworking / plugins#484** for details) used by Kubernetes is affected. The following kubelet versions contain the affected CNI plug-in:

- kubelet v1.18.0–v1.18.3
- kubelet v1.17.0–v1.17.6
- kubelet versions earlier than v1.16.11

Vulnerability Analysis Results

The CCI service is not affected by this vulnerability. The reason is as follows:

CCI workloads are deployed on clusters of Kubernetes v1.15 that do not have IPv6 enabled. Therefore, **CCI nodes will not be attacked.**

12.4 CVE-2020-8559 Vulnerability Notice

The Huawei Cloud CCI team noticed the Kubernetes security vulnerability CVE-2020-8559 on July 22. After detailed analysis, it is found that **the vulnerability has no impact on users and CCI services, and does not need to be handled**.

Vulnerability Details

Kubernetes recently disclosed the security vulnerability CVE-2020-8559 in the kube-apiserver component, with CVSS rating of Medium (6.4) **CVSS:3.1/AV:N/AC:H/PR:H/UI:R/S:U/C:H/I:H/A:H**.

Vulnerability brief: An attacker can intercept certain upgrade requests sent to kubelet of a node and forward the requests to other target nodes using the original access credentials in the requests. This can lead to permission escalation.

Reference link: **https://github.com/kubernetes/kubernetes/issues/92914**

How Do I Determine Whether a Vulnerability Is Involved?

Affected cluster versions are used:

- kube-apiserver v1.18.0–v1.18.5
- kube-apiserver v1.17.0–v1.17.8
- kube-apiserver v1.16.0–v1.16.12
- kube-apiserver versions earlier than v1.16.0

Vulnerability Analysis Results

The CCI service is not affected by this vulnerability. The reason is as follows:

CCI workloads are deployed on clusters of Kubernetes v1.15, and the container network is based on the user's VPC. No user can access nodes or intercept kubelet requests. Therefore, **nodes will not be attacked.**

12.5 CVE-2020-8557 Vulnerability Notice

The Huawei Cloud CCI team noticed the Kubernetes security vulnerability CVE-2020-8557 on July 22. After detailed analysis, it is found that **the vulnerability has no impact on users and CCI services, and does not need to be handled**.

Vulnerability Details

Kubernetes officially released the security vulnerability CVE-2020-8557, with CVSS rating of Medium (5.5) **CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H/CR:H/IR:H/AR:M**.

Vulnerability brief: The eviction manager of kubelet does not manage the temporary storage usage of the **/etc/hosts** file mounted to pods. Attackers can use this vulnerability to write a large amount of data to the **/etc/hosts** file, which fills the storage space of a node and causes denial of service.

Reference link: **https://github.com/kubernetes/kubernetes/issues/93032**

How Do I Determine Whether a Vulnerability Is Involved?

Affected cluster versions are used:

- kubelet v1.18.0–v1.18.5

- kubelet v1.17.0–v1.17.8
- kubelet versions earlier than v1.16.13

Vulnerability Analysis Results

The CCI service is not affected by this vulnerability. The reasons are as follows:

- CCI workloads are deployed on clusters of Kubernetes v1.15 and run Kata containers. The hosts file on the nodes is not directly mounted to the containers. Therefore, **nodes will not be attacked.**
- **Service containers of different tenants are completely isolated. Malicious users cannot access containers of other users.**