ServiceStage

User Guide

Issue 01

Date 2024-07-09

HUAWEI CLOUD COMPUTING TECHNOLOGIES CO., LTD.

Trademarks and Permissions


and other Huawei trademarks are the property of Huawei Technologies Co., Ltd.
All other trademarks and trade names mentioned in this document are the property of their respective holders.

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.

Huawei Cloud Computing Technologies Co., Ltd.

Address: Huawei Cloud Data Center Jiaoxinggong Road
Qianzhong Avenue
Gui'an New District
Gui Zhou 550029
People's Republic of China

Website: https://www.huaweicloud.com/intl/en-us/

Contents

1 Overview....1
2 Permissions Management....6
2.1 Creating a User and Granting ServiceStage Permissions....6
2.2 Creating a Custom Policy....7
2.3 Assigning Permissions to ServiceStage-Dependent Services....8
3 Environment Management....10
3.1 Environment Overview....10
3.2 Creating an Environment....10
3.3 CCE Resource Management....13
3.3.1 Binding a CCE Cluster....13
3.3.2 Unbinding a CCE Cluster....14
3.3.3 Managing Namespaces....15
3.3.4 Managing Configuration Items....18
3.3.5 Managing Secrets....22
3.4 Managing Resources....26
3.5 Removing Managed Resources....29
3.6 Upgrading a VM Agent....29
3.7 Restarting a VM Agent....30
3.8 Modifying an Environment....30
3.9 Deleting an Environment....31
3.10 Installing a VM Agent....32
4 Application Management....35
4.1 Creating an Application....35
4.2 Viewing Application Overview....36
4.3 Managing Application Environment Variables....37
4.4 Editing an Application....40
4.5 Deleting an Application....41
5 Component Management....43
5.1 Component Overview....43
5.2 Creating and Deploying a Component....54
5.3 Viewing Component Details....70
5.4 Managing Component Labels....71

5.5 Changing the Container Name of a CCE-deployed Component ... 73
5.6 Changing the Component Description ... 74
5.7 Managing Component Instances ... 75
5.8 Upgrading a Single Component ... 76
5.8.1 Single-batch Release ... 76
5.8.2 Rolling Release ... 80
5.8.3 Dark Launch (Canary) ... 84
5.9 Upgrading Components in Batches ... 89
5.10 Cloning Components in Batches ... 91
5.11 Synchronizing Component Configurations in Batches ... 96
5.12 Rolling Back a Component ... 96
5.13 Redeploying a Component ... 97
5.13.1 Single-batch Release ... 97
5.13.2 Rolling Release ... 100
5.13.3 Dark Launch (Canary) ... 104
5.14 Configuring the Component Access Mode ... 108
5.15 Changing the Component Access Domain Name ... 110
5.16 Configuring a Scaling Policy of a Component Instance ... 110
5.17 Component O&M ... 116
5.17.1 Viewing Component Running Metrics ... 116
5.17.2 Customizing Component Running Metrics ... 117
5.17.3 Managing Component Logs ... 117
5.17.3.1 Managing Component AOM Logs ... 117
5.17.3.2 Managing Component LTS Logs ... 118
5.17.3.2.1 LTS Log Overview ... 118
5.17.3.2.2 Associating an LTS Log Group ... 119
5.17.3.2.3 Searching for Running Logs ... 119
5.17.3.2.4 Quickly Querying Logs ... 120
5.17.3.2.5 Using Visualization to Analyze Logs ... 121
5.17.3.2.6 Viewing Real-Time Logs ... 122
5.17.3.2.7 Unbinding an LTS Log Group ... 123
5.17.3.3 Viewing Container Logs ... 123
5.17.4 Configuring Alarm Thresholds for Resource Monitoring ... 124
5.17.5 Viewing Component Running Events ... 127
5.18 Viewing the Component Running Environment ... 127
5.19 Starting and Stopping a Component Instance ... 128
5.20 Deleting a Component ... 128
5.21 Synchronizing Component Status ... 129
5.22 Component Advanced Setting ... 129
5.22.1 Configuring Environment Variables of a Component ... 129
5.22.2 Configuring the Lifecycle of a Component ... 131
5.22.3 Configuring Data Storage ... 132

5.22.4 Configuring Distributed Cache Service....143
5.22.5 Configuring Relational Databases....144
5.22.6 Configuring a Scheduling Policy of a Component Instance....145
5.22.7 Configuring a Log Policy of an Application....148
5.22.8 Configuring Custom Monitoring of a Component....149
5.22.9 Configuring Application Performance Management....150
5.22.10 Configuring Health Check....151
6 Deployment Source Management....154
6.1 Software Center....154
6.1.1 Managing Software Packages....154
6.1.2 Packaging Specifications of Software Packages....157
6.2 Image Repository....158
6.2.1 Uploading an Image....159
6.2.2 Managing Images....160
6.3 Organization Management....163
7 Continuous Delivery....166
7.1 Overview....166
7.2 Viewing Build Jobs....167
7.3 Creating a Source Code Job....168
7.4 Creating a Package Job....171
7.5 Maintaining Build Jobs....173
7.6 Managing Pipelines....176
7.7 Authorizing a Repository....180
8 Microservice Engine....182
8.1 Cloud Service Engine Overview....182
8.2 Creating a Microservice Engine....182
8.3 Managing Microservice Engines....186
8.3.1 Viewing Microservice Engine Information....186
8.3.2 Obtaining the Service Center Address of a Microservice Engine....187
8.3.3 Obtaining the Configuration Center Address of a Microservice Engine....188
8.3.4 Viewing the Instance Quota of a Microservice Engine....188
8.3.5 Viewing the Configuration Item Quota of a Microservice Engine....189
8.3.6 Configuring Backup and Restoration of a Microservice Engine....189
8.3.7 Managing Public Network Access for a Microservice Engine....191
8.3.7.1 Binding an EIP....191
8.3.7.2 Unbinding an EIP....192
8.3.8 Viewing Microservice Engine Operation Logs....193
8.3.9 Upgrading a Microservice Engine Version....193
8.3.10 Deleting a Microservice Engine....194
8.3.11 Managing Security Authentication for a Microservice Engine....195
8.4 Using Microservice Engines....197

8.4.1 Using the Microservice Dashboard 197
8.4.2 Managing Microservices 198
8.4.3 Microservice Governance 210
8.4.3.1 Overview 211
8.4.3.2 Governing Microservices 211
8.4.4 Configuration Management (Applicable to Engine 2.x) 221
8.4.5 Configuration Management (Applicable to Engine 1.x) 231
8.4.6 System Management 233
8.4.6.1 Overview 234
8.4.6.2 Accounts 235
8.4.6.3 Roles 241

9 Key Operations Recorded by CTS 247
9.1 ServiceStage Operations That Can Be Recorded by CTS 247
9.2 Querying Real-Time Traces 248

10 Viewing Monitoring Metrics and Alarms 252

1 Overview

ServiceStage is an application management and O&M platform that lets you deploy, roll out, monitor, and maintain applications all in one place. It supports technology stacks such as Java, PHP, Python, Node.js, Docker, and Tomcat, and supports microservice applications such as Apache ServiceComb Java Chassis (Java chassis) and Spring Cloud, making it easier to migrate enterprise applications to the cloud.

This document describes how to use ServiceStage to create, deploy, and maintain application components and perform service governance.

Prerequisites

1. You have **registered a Huawei account and enabled Huawei Cloud services**.
2. The login account has the permission to use ServiceStage. For details, see **Creating a User and Granting ServiceStage Permissions**.

Logging In to ServiceStage

Step 1 Log in to the **management console**.

Step 2 Click  and select a region.

Step 3 Click  in the upper left corner, and click **ServiceStage**.

- If you log in for the first time, click **Authorize** on the displayed service authorization page to authorize ServiceStage to use the services on which it depends. Then, the **ServiceStage** console is displayed.
- If this is not your first login, the **ServiceStage** console is displayed directly.

----End

Console Description

Table 1-1 describes ServiceStage console.

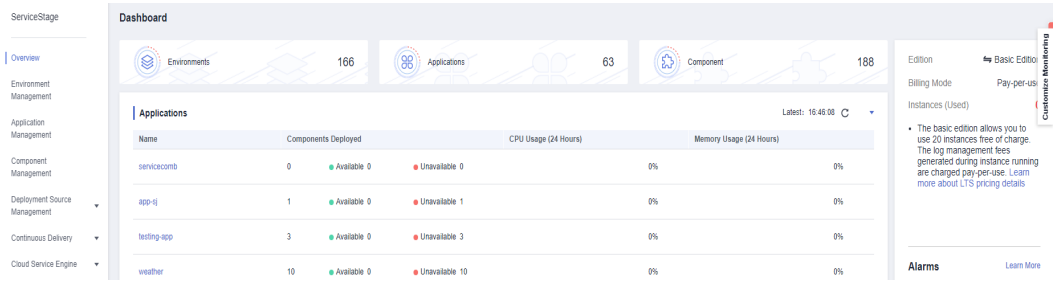
Table 1-1 ServiceStage console

Module	Description
Overview	The **Overview** page provides a dashboard, including ServiceStage edition selection, total number of environments, applications, and components, monitoring information, alarms, and documentation. • Edition selection: ServiceStage provides the basic and professional editions in pay-per-use billing mode. You can select an edition as required. For details, see **Upgrading Product Versions**. • Environments: displays the number of created environments. You can click **Environments** to go to the **Environment Management** page and view environment details. • Applications: displays the number of created applications. You can click **Applications** to go to the **Application Management** page and view application details. • Components: displays the number of deployed components. You can click **Components** to go to the **Component Management** page and view component details. • Customize Monitoring: Move the cursor to **Customize Monitoring** in the upper right corner and select the applications and environment to be displayed on the **Overview** page. A maximum of four monitoring information records can be displayed. Where, – Applications: displays the name of each application, number of deployed components (including available and unavailable components), and CPU and memory usage of the application. – Environments: displays the name of each environment, CPU and memory usage in the environment, number of components deployed in the environment, resource health, and instance health of deployed components. Click **CPU usage** or **Memory usage** on an environment card to enable or disable the information display. • Remove the monitoring information that does not need to be displayed: – Click ▼ in the upper left corner of a card to be removed and click **Remove**. – Move the cursor to **Customize Monitoring** in the upper right corner and deselect the monitoring information that does not need to be displayed. • Alarms: Click **Learn More** in the **Alarms** area to go to the AOM console and view ServiceStage alarm details. • Documentation: Click **Learn More** in the **Documentation** area to view ServiceStage documents.

Module	Description
Environment Management	An environment is a collection of compute, network, and middleware resources used for deploying and running a component. The **Environment Management** page allows you to create, edit, and delete environments, and configure resources (manage and remove resources). Created environments are displayed in a list.
Application Management	An application is a service system with complete functions and consists of one or more components related to features. The **Application Management** page allows you to create, edit, and delete applications. Created applications and the number of components created under them are displayed in a list, and entries for creating components under applications are available.
Component Management	A component is a service feature implementation of an application. It is carried by code or software packages and can be independently deployed and run. The **Component Management** page displays components of all applications in a list, and provides the component details page as well as the entries for component creation and O&M.
Deployment Source Management	Provides functions such as organization management, software repository, and image repository. • Organization management is used to isolate images and assign access permissions (read, write, and manage) to different users. • Image repositories are used to store and manage Docker images. • Software repositories are used to store, manage, and deploy software packages.

Module	Description
Continuous Delivery	Provides functions such as viewing build projects, releasing build projects, and authorizing repositories. • Build The software package or image package can be generated with a few clicks in a build job. In this way, the entire process of source code pull, compilation, packaging, and archiving is automatically implemented. • Pipeline One-click deployment can be achieved through pipeline. In this way, the entire process of source code pull, compilation, packaging, archiving, and deployment is automatically implemented. This unifies the integration environment and standardizes the delivery process. • Repository Authorization You can create repository authorization so that build projects and application components can use the authorization information to access the software repository.
Cloud Service Engine	Provides operation entries for engine instance management, dashboard usage, microservice catalog management, microservice governance, configuration management, and system management.

Figure 1-1 ServiceStage console



Product Versions

Log in to the ServiceStage console and select an edition on the **Overview** page. Currently, ServiceStage provides basic edition and professional edition.

Table 1-2 ServiceStage edition description

Edition	Package Description
Basic	20 instances are free of charge. A maximum of 100 instances are supported.
Professional	More than 100 instances are supported.

NOTE

For product pricing of each edition, see **Product Pricing Details**.

Upgrading Product Versions

Step 1 Log in to ServiceStage and go to the **Overview** page.

Step 2 On the right of the **Overview** page, click ⇌ next to **Edition**.

Step 3 Select a product version and click **OK**.

NOTE

Only the account administrator can upgrade a package.

For the definitions of an account and IAM user, see **Basic Concepts**.

----End

2 Permissions Management

2.1 Creating a User and Granting ServiceStage Permissions

You can use **Identity and Access Management (IAM)** for fine-grained permissions control for your ServiceStage. With IAM, you can:

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

If your Huawei Cloud account does not require individual IAM users, you may skip over this section.

This section describes the procedure for granting permissions. **Figure 2-1** shows the process flow.

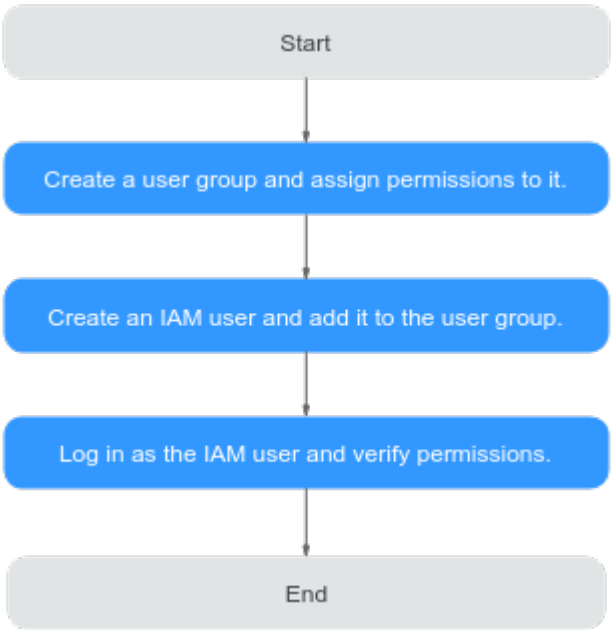
Prerequisites

Learn about the permissions supported by ServiceStage and choose permissions as required. For details about system permissions supported by ServiceStage, see **Permissions Management**.

For the permissions of other services, see **System-defined Permissions**.

Process Flow

Figure 2-1 Process for granting ServiceStage permissions



1. **create a user group and grant it permissions** .

   Create a user group on the IAM console and assign the **ServiceStage ReadOnlyAccess** permissions to the group.

2. **Create an IAM user and add it to the created user group**.

   Create a user on the IAM console and add it to the user group created in **1**.

3. **Log in as the IAM user** and verify permissions.

   Perform the following operations:

   a. Choose **Service List** > **ServiceStage**.

   b. Choose **Application Management** and click **Create Application**.

      If a message appears indicating that you have insufficient permissions to perform the operation, the **ServiceStage ReadOnlyAccess** policy is in effect.

2.2 Creating a Custom Policy

Custom policies can be created as a supplement to the system-defined policies of ServiceStage.

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

- Visual editor: Select cloud services, actions, resources, and request conditions. This does not require knowledge of policy syntax.
- JSON: Create a policy in the JSON format from scratch or based on an existing policy.

For details, see **Creating a Custom Policy**. The following section contains examples of common ServiceStage custom policies.

Example Custom Policy

This procedure creates a policy that an IAM user is prohibited to create and delete a microservice engine.

```
{
    "Version": "1.1",
    "Statement": [
        {
            "Action": [
                "cse:*:*"
            ],
            "Effect": "Allow"
        },
        {
            "Action": [
                "cse:engine:create",
                "cse:engine:delete"
            ],
            "Effect": "Deny"
        }
    ]
}
```

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

After authorization, users in the group can verify their permissions using the console or REST APIs.

The following uses the custom policy as an example to describe how to log in to the ServiceStage console to verify that a user is not allowed to create microservice engines.

1. Log in to Huawei Cloud as an IAM user.
   - Tenant name: Name of the Huawei Cloud account used to create the IAM user
   - IAM username and password: Username and password specified during the IAM user creation using the tenant name
2. On the **Cloud Service Engines** page, create a microservice engine. If error 403 is returned, the permissions are correct and have taken effect.

2.3 Assigning Permissions to ServiceStage-Dependent Services

Assigning CCE Namespace Permissions

You can assign only common operation permissions on CCE cluster resources to the ServiceStage user group using IAM, excluding the namespace permissions of the clusters with Kubernetes RBAC authentication enabled. Therefore, assign the namespace permissions to the clusters separately.

For details, see **Namespace Permissions**.

Assigning CTS Permissions

After the permissions are assigned for ServiceStage using IAM, they do not take effect for the CTS service on which ServiceStage depends. Therefore, assign the CTS service permissions separately.

For details, see **Permissions Management**.

3 Environment Management

3.1 Environment Overview

An environment is a collection of compute, network, and middleware resources used for deploying and running a component. ServiceStage combines the compute resources (such as CCE clusters and ECSs), network resources (such as ELB instances and EIPs), and middleware (such as DCS instances, RDS instances, and CSE engines) into an environment, such as a development environment, testing environment, pre-production environment, or production environment.

The resources within an environment can be networked together. Managing resources and deploying services by environment simplifies O&M.

A maximum of 300 environments can be created in a project.

3.2 Creating an Environment

Create an environment before deploying components.

Procedure

Step 1 Log in to ServiceStage.

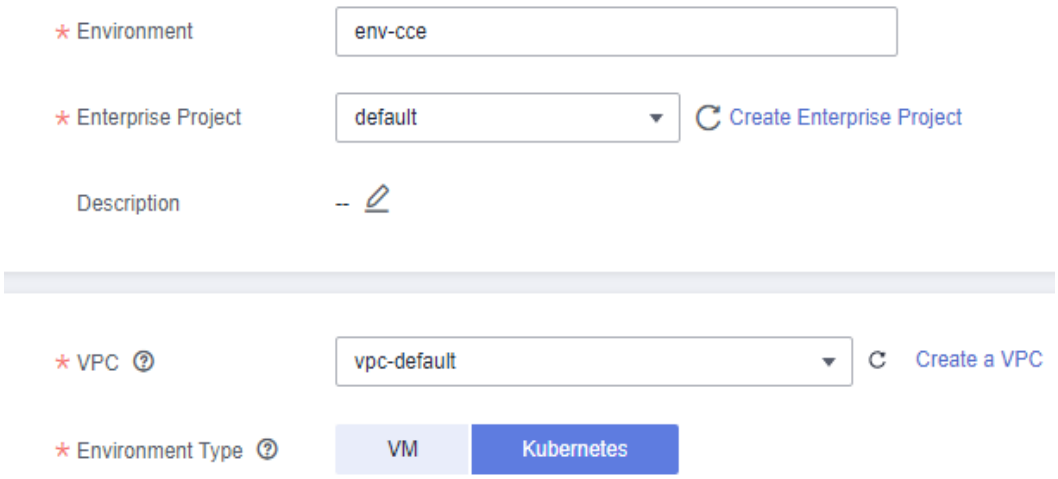
Step 2 Choose **Environment Management** > **Create Environment** and configure the environment. Parameters marked with an asterisk (*) are mandatory.

Parameter	Description
*Environment	Environment name.
*Enterprise Project	Enterprise projects let you manage cloud resources and users by project. It is available after you **enable the enterprise project function**.

Parameter	Description
Description	Environment description. 1. Click  and enter the environment description. 2. Click  to save the description.
*VPC	**NOTICE** After the environment is created, the VPC cannot be modified during **Modifying an Environment**. Select the VPC where the environment is located based on your service requirements. VPC where the environment resources are located. • To use a created VPC, select a VPC created under the current account from the drop-down list. • To use a shared VPC, select a VPC that another account shares with the current account from the drop-down list. VPC owners can share the subnets in a VPC with one or multiple accounts through **Resource Access Manager (RAM)**. Through VPC sharing, you can easily configure and manage multiple accounts' resources at low costs. For more information about VPC and subnet sharing, see **VPC Sharing**. • To use a new VPC, click **Create a VPC**. For details, see **Creating a VPC**.
*Environment Type	Environment type. • **VM**: applicable to VM-based deployment. Components are deployed on VMs using software packages. • **Kubernetes**: applicable to container-based deployment (CCE). Components are deployed using container images and scheduled by Kubernetes. **NOTE** • For details about the component deployment mode, see **Deploying a Component**. • If the environment type is Kubernetes and CCE is used for deployment, you are advised to run the following command on the CCE node to add a firewall: **iptables --append OUTPUT --proto tcp --destination 169.254.169.254 --match owner ! --uid-owner root -j REJECT** **169.254.169.254** is a special address in OpenStack and is used to provide instance metadata and user data services. This command prevents non-root users from obtaining sensitive information or performing unauthorized operations using this address.

Parameter	Description
*Tags **NOTE** Set tags in **CN East2**.	Tags are used to identify cloud resources. When you have multiple cloud resources of the same type, you can use tags to classify them based on usage, owner, or environment. If your organization has configured tag policies for ServiceStage, add tags to trackers based on the policies. If a tag does not comply with the tag policies, environment creation may fail. Contact your administrator to learn more about tag policies. Each environment can have a maximum of 20 tags. 1. Click **Add Tag**. The **Add Tag** dialog box is displayed. 2. Click **Add Tag**. 3. Enter the key and value of the tag. It is recommended that you **create a predefined tag** on TMS to add the same tag to different resources. 4. Click **OK**.

Figure 3-1 Configuring an environment



Step 3 Click **Create Now**.

After the environment is created, go to the environment details page to view the environment details and configure environment resources.

NOTE

If CCE clusters and VMs are managed in an earlier version, the environment type is **VM + Kubernetes** after the upgrade to the current version.

----End

Follow-Up Operations

- After a Kubernetes environment is created, bind a CCE cluster to the environment before using the environment to deploy components. For details, see **CCE Resource Management**.

- After an environment is created, the compute resources (excluding CCE clusters), network resources, and middleware need to be managed together to form an environment. For details, see **Managing Resources**.

3.3 CCE Resource Management

3.3.1 Binding a CCE Cluster

Before deploying components in a Kubernetes environment, bind only one CCE cluster to the environment so that components can be deployed and run in the Kubernetes environment.

Prerequisites

1. A CCE cluster to be bound has been created and is in the running state. The cluster must be in the same VPC as the target environment and cannot be managed by other environments.

   CCE clusters are used to deploy and run components in **Kubernetes** environments.

   For details about how to create a CCE cluster, see **Buying a Cluster**.

   For details about how to add a node to the CCE cluster, see **Creating a Node**.

   NOTE

   - ServiceStage supports **Elastic Cloud Server (VM)** and **BMS** nodes in CCE clusters.
   - For details about the OSs supported by CCE cluster nodes, see **Node OS**.
   - Container engines of CCE cluster nodes support **Docker** and **Containerd**. For details about relationship between node OSs and container engines, see **Container Engine**.

2. A Kubernetes environment has been created. For details, see **Creating an Environment**.

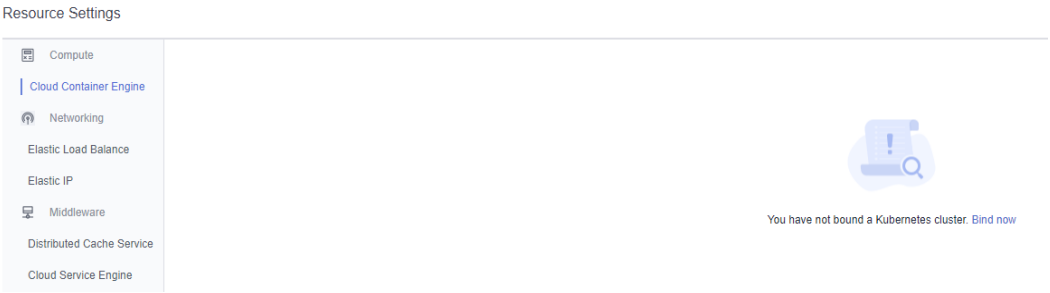
Procedure

Step 1 Log in to ServiceStage.

Step 2 On the **Environment Management** page, click the target environment.

Step 3 In the **Resource Settings** area, choose **Cloud Container Engine** from **Compute**.

Figure 3-2 Binding a CCE cluster



Step 4 Click **Bind now**.

- If the CCE cluster to be bound has been created, select the cluster from the cluster drop-down list and click **OK**.
- If no CCE cluster is created, go to the CCE console as prompted, create a CCE cluster by referring to **Prerequisites**, and bind the CCE cluster.

NOTE

In a Kubernetes environment, if IPv6 is enabled for the selected VPC and CCE clusters are managed, select a CCE cluster with IPv6 enabled. Otherwise, the Java chassis microservice registered on the exclusive microservice engine with security authentication enabled in the VPC fails to register the discovery address using IPv6.

For details, see **What Should I Do If the Service Registration Fails After IPv6 Is Enabled for the Exclusive Microservice Engine with Security Authentication Enabled?**

----End

Follow-Up Operations

- Click the **Node List** tab to view details about each node in the CCE cluster.
- Click **View Resource Details** to view CCE cluster details on the CCE console.

3.3.2 Unbinding a CCE Cluster

If a CCE cluster that has been bound to a Kubernetes environment is no longer used, you can remove it from the environment.

Prerequisites

A CCE cluster has been bound to the environment. For details, see **Binding a CCE Cluster**.

Procedure

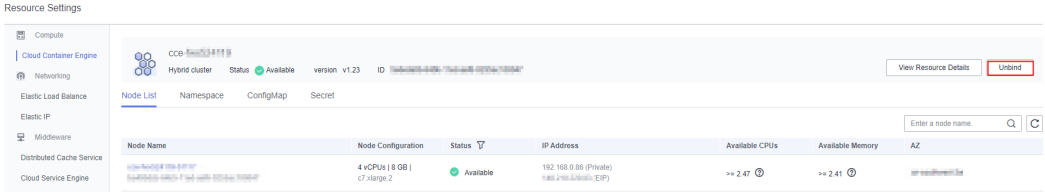
Step 1 Log in to ServiceStage.

Step 2 On the **Environment Management** page, click the target environment.

Step 3 In the **Resource Settings** area, choose **Cloud Container Engine** from **Compute**.

Step 4 Click **Unbind** to remove the CCE cluster from the environment.

Figure 3-3 Unbinding a CCE cluster



----End

3.3.3 Managing Namespaces

A namespace is a collection of resources and objects. Multiple namespaces can be created in a single CCE cluster with the data isolated from each other. This enables namespaces to share the services of the same cluster without affecting each other.

For example, you can deploy workloads in a development environment into one namespace, and deploy workloads in a testing environment into another namespace.

Table 3-1 describes the namespace types.

Table 3-1 Namespace types

Creation Type	Description
Created by a cluster by default	When a cluster is started, the **default**, **kube-public**, **kube-system**, and **kube-node-lease** namespaces are created by default. • **default**: All objects for which no namespace is specified are allocated to this namespace. • **kube-public**: Resources in this namespace can be accessed by all users (including unauthenticated users), such as public add-ons and container charts. • **kube-system**: All resources created by Kubernetes are in this namespace. • **kube-node-lease**: Each node has an associated Lease object in this namespace. The object is periodically updated by the node.
Created manually	You can create namespaces to serve separate purposes. For example, you can create three namespaces, one for a development environment, one for joint debugging environment, and one for testing environment. You can also create one namespace for login services and one for game services.

This section describes how to create and delete a namespace, and manage namespace resource quotas.

Prerequisites

A CCE cluster has been bound to the environment. For details, see **Binding a CCE Cluster**.

Creating a Namespace

Step 1 Log in to ServiceStage.

Step 2 On the **Environment Management** page, click the target environment.

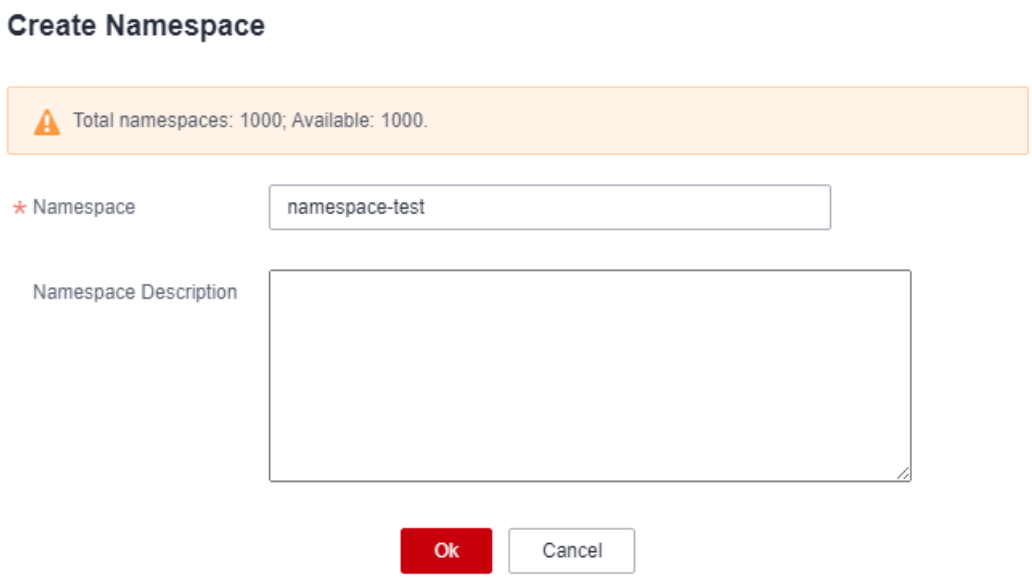
Step 3 In the **Resource Settings** area, choose **Cloud Container Engine** from **Compute**.

Step 4 Click **Namespace** > **Create Namespace**.

Step 5 Set the parameters by referring to the following table. Parameters marked with an asterisk (*) are mandatory.

Parameter	Description
*Namespace	Namespace name.
Namespace description	Description about the namespace.

Figure 3-4 Setting namespace parameters



Step 6 Click **OK**.

The created namespace is displayed in the namespace list.

----End

Deleting a Namespace

NOTICE

- If a namespace is deleted, all resources (such as workloads and configuration items) in this namespace will be also deleted. Exercise caution when deleting a namespace.
- The cluster-created namespace **default** cannot be deleted.

Step 1 Log in to ServiceStage.

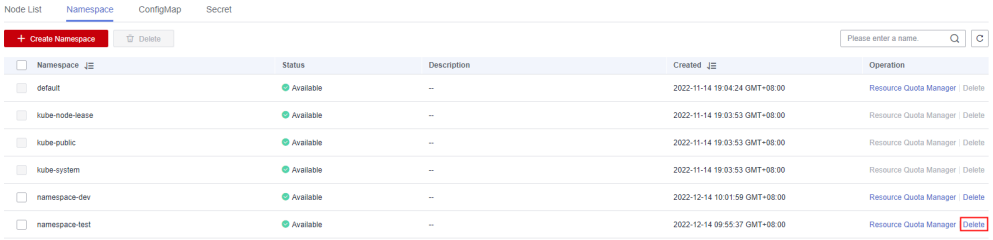
Step 2 On the **Environment Management** page, click the target environment.

Step 3 In the **Resource Settings** area, choose **Cloud Container Engine** from **Compute**.

Step 4 Click the **Namespace** tab.

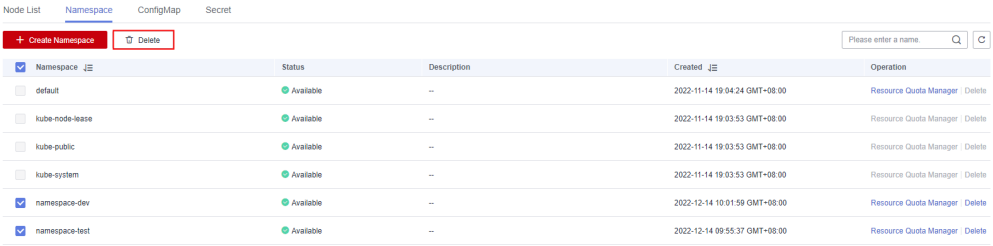
- To delete a single namespace, locate the target namespace and click **Delete** in the **Operation** column.

  Figure 3-5 Deleting a single namespace



- To delete namespaces in batches, select the target namespaces and click **Delete** above the namespaces.

  Figure 3-6 Deleting namespaces in batches



Step 5 In the displayed dialog box, enter **DELETE** and click **OK**.

----End

Managing Namespace Resource Quotas

By default, pods running in a CCE cluster can use the CPUs and memory of a node without restrictions. This means that pods in a namespace may exhaust all resources of the cluster.

Kubernetes provides namespaces for you to group workloads in a cluster. By setting resource quotas for each namespace, you can prevent resource exhaustion and ensure cluster reliability. You can configure quotas for resources such as CPU, memory, and the number of pods in a namespace. For more information, see **Resource Quotas**.

User-created namespaces and the cluster-created namespace **default** support resource quota management.

Step 1 Log in to ServiceStage.

Step 2 On the **Environment Management** page, click the target environment.

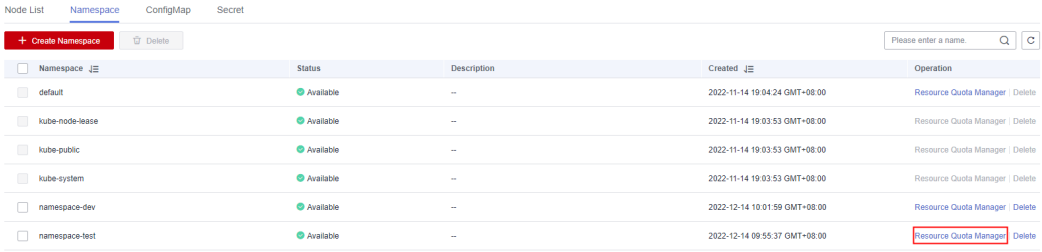
Step 3 In the **Resource Settings** area, choose **Cloud Container Engine** from **Compute**.

Step 4 Click the **Namespace** tab.

Step 5 Locate the target namespace and click **Resource Quota Manager** in the **Operation** column.

In the displayed **Resource Quota Manager** dialog box, view the resource types, total resource quotas, and accumulated quota usage in the namespace.

Figure 3-7 Accessing the Resource Quota Manager page



Step 6 Click **Edit Quota** and set the total quota of each resource type.

- If the usage of a resource type is not limited, leave it blank.
- If the usage of a resource type is limited, enter an expected integer ranging from 1 to 9,007,199,254,740,992.

NOTICE

- Accumulated quota usage includes the resources used by CCE to create default components, such as the Kubernetes Services (which can be viewed using kubectl) created under the **default** namespace. Therefore, you are advised to set a resource quota greater than expected to reserve resource for creating default components.
- If the total CPU or memory quota in a namespace is limited, you must set the maximum and minimum number of CPU cores and memory (GiB) that can be used by the component when setting resources for the Kubernetes component of this namespace in **Creating and Deploying a Component** and **Upgrading a Single Component**. Otherwise, the operation will fail.
- If the total quota of other resource types in a namespace is limited and the remaining usage of this resource type does not meet requirements, the Kubernetes component of this namespace will fail to be deployed.

Step 7 Click **OK**.

----End

3.3.4 Managing Configuration Items

Configuration items (ConfigMaps) are user-defined resources that store application configurations. They can be used as files or environment variables in applications.

Configuration items allow you to decouple configuration files from images to enhance the portability of applications.

Benefits of configuration items:

- Manage configurations for different environments and services.

- Deploy applications in different environments. You can maintain configuration files in multiple versions, which makes it easy to update and roll back applications.
- Quickly import configurations in the form of files to containers.

This section describes how to create, delete, view, and update configuration items.

Prerequisites

1. A CCE cluster has been bound to the environment. For details, see **Binding a CCE Cluster**.
2. The namespace to which the configuration item belongs has been created. For details, see **Creating a Namespace**.

Creating a Configuration Item

Step 1 Log in to ServiceStage.

Step 2 On the **Environment Management** page, click the target environment.

Step 3 In the **Resource Settings** area, choose **Cloud Container Engine** from **Compute**.

Step 4 Click **Configuration Item** > **Create Configuration Item**.

ServiceStage allows you to create configuration items in **Visualization** or **YAML** mode.

- Method 1: Visualization

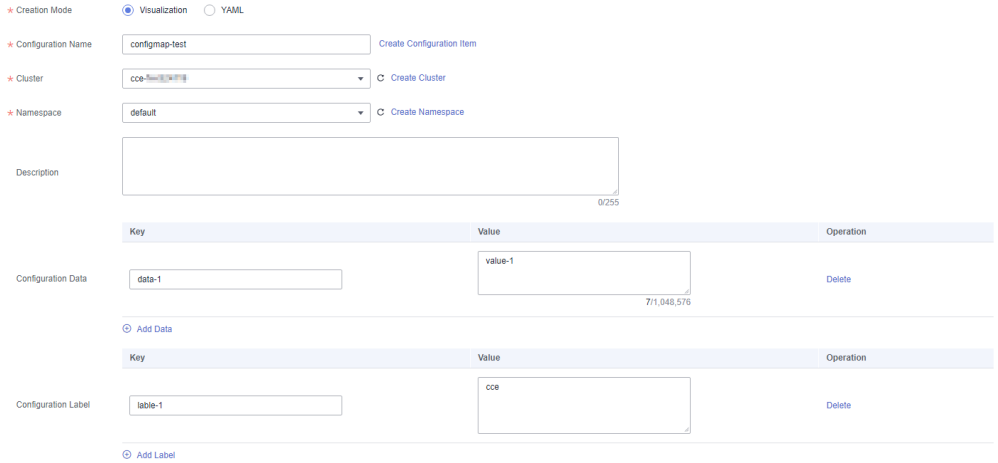
  Configure the configuration item by referring to **Table 3-2**. Parameters marked with an asterisk (*) are mandatory.

  Table 3-2 Parameters for creating a configuration item in visualization mode

Parameter	Description
*Configuration Name	Name of the configuration item, which must be unique in a namespace.
*Cluster	Cluster where the configuration item will be used.
*Namespace	Namespace to which the configuration item belongs. The default value is **default**.
Description	Description of the configuration item.
Configuration Data	Configuration data to be used in applications or used to store configuration data. **Key** indicates a file name, and **Value** indicates the content in the file. 1. Click **Add Data**. 2. Enter the key and value.

Parameter	Description
Configuration Labels	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 attributes of these objects and are used to manage and select the objects. 1. Click **Add**. 2. Enter the key and value.

Figure 3-8 Setting configuration item parameters in Visualization mode



- Method 2: YAML

  NOTE

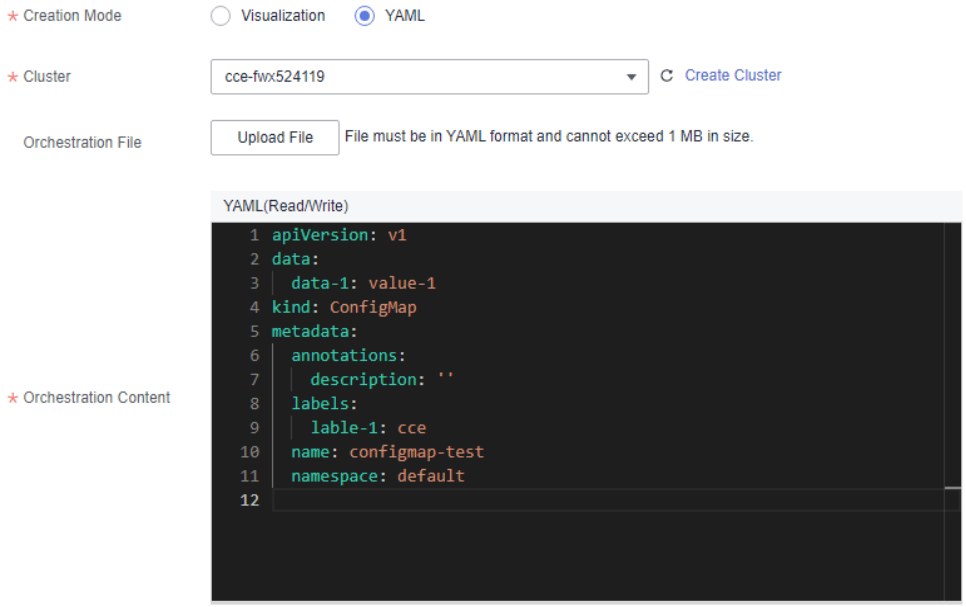
  To create a configuration item by uploading a file, ensure that a configuration item description file has been created. ServiceStage supports files in YAML format. For details, see **Configuration Item Requirements**.

  a. Select a cluster from the **Cluster** drop-down list.

  b. (Optional) Click **Upload File** and select the ConfigMap resource file created locally. Click **Open** and wait until the upload is successful.

  c. Write or modify the ConfigMap resource file in **Orchestration content**.

Figure 3-9 Setting configuration item parameters in YAML mode



Step 5 Click **Create ConfigMap**.

After the configuration item is created, it is displayed in the configuration item list.

----End

Follow-Up Operations

After a configuration item is created, you can search for, view, update, and delete the configuration item by referring to **Table 3-3**.

NOTE

- Deleted items cannot be restored. Exercise caution when performing this operation.
- The configuration item list contains system configuration items, which can only be viewed and cannot be modified or deleted.

Table 3-3 Configuration item management operations

Operation	Description
Searching for a configuration item	1. Select the namespace to which the configuration item belongs from the namespace drop-down list. 2. Enter a configuration item name in the search box.
Viewing a configuration item	Click **Show YAML** in the **Operation** column of the target configuration item to view the content of the YAML file of the configuration item.

Operation	Description
Modifying a configuration item	1. Click **Update** in the **Operation** column of the target configuration item. 2. Modify the information according to **Table 3-2**. 3. Click **Update Configuration Item**.
Deleting a configuration item	1. Click **Delete** in the **Operation** column of the target configuration item. 2. In the displayed dialog box, click **OK**.
Deleting configuration items in batches	1. Select the configuration items to be deleted. 2. Click **Delete Configuration Item**. 3. In the displayed dialog box, click **OK**.

Configuration Item Requirements

A configuration item resource file should be in YAML format, and the file size cannot exceed 1 MB.

Example:

```
apiVersion: v1
data: {}
kind: ConfigMap
metadata:
  annotations:
    description: ''
  labels: {}
  name: configmap-ww8qkl
  namespace: cse
```

3.3.5 Managing Secrets

Secrets are user-defined resources that store authentication and sensitive information such as application keys. They can be used as files or environment variables in applications.

This section describes how to create, delete, view, and update secrets.

Prerequisites

1. A CCE cluster has been bound to the environment. For details, see **Binding a CCE Cluster**.
2. The namespace to which the secret belongs has been created. For details, see **Creating a Namespace**.

Creating a Secret

Step 1 Log in to ServiceStage.

Step 2 On the **Environment Management** page, click the target environment.

Step 3 In the **Resource Settings** area, choose **Cloud Container Engine** from **Compute**.

Step 4 On the **Secret** page, click **Create Secret**.

ServiceStage allows you to create secrets in **Visualization** or **YAML** mode.

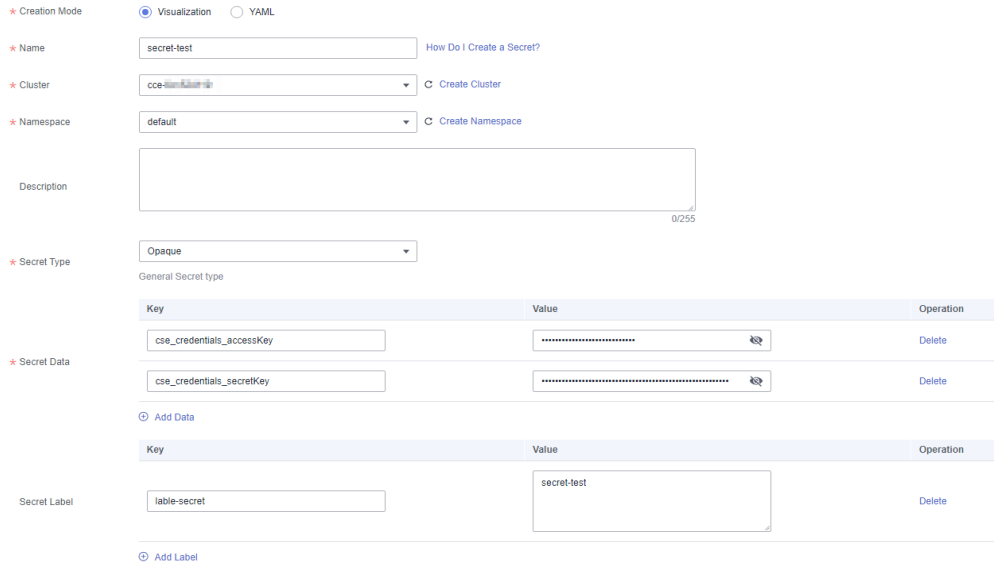
- Method 1: Visualization Configure the parameters by referring to **Table 3-4**. Parameters marked with an asterisk (*) are mandatory.

Table 3-4 Parameters for creating a secret in visualization mode

Parameter	Description
*Secret Name	Name of a secret, which must be unique in the same namespace.
*Cluster	Cluster where the secret will be used. Click **Create Cluster** to create a cluster.
*Namespace	Namespace to which the secret belongs. If you do not specify this parameter, the value **default** is used by default.
Description	Description of a secret.
*Secret Type	Select the type of the secret to be created based on service requirements. – **Opaque**: general secret type. If the secret type is not explicitly set in the secret configuration file, the default secret type is **Opaque**. – **kubernetes.io/dockerconfigjson**: a secret that stores the authentication information required for pulling images from a private repository. – **IngressTLS**: a secret that stores the certificate required by ingresses (layer-7 load balancing services). – **Other**: Enter a secret type that is none of the above.
*Repository Address	This parameter is valid only when **Secret Type** is set to **kubernetes.io/dockerconfigjson**. Enter the address of the image repository.
*Secret data	Value of the **data** field in the application secret file. – If the secret type is **Opaque**, enter the key and value. The value must be encoded using Base64. For more information, see **Base64 Encoding**. Click **Add Data** to add secret data. – If the secret type is **kubernetes.io/dockerconfigjson**, enter the username and password. – If the secret type is **IngressTLS**, upload the certificate file and private key file. – If the secret type is **Other**, enter the key and value.

Parameter	Description
Secret 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 attributes of these objects and are used to manage and select the objects. 1. Click **Add**. 2. Enter the key and value.

Figure 3-10 Setting secret parameters in Visualization mode



- Method 2: YAML

  NOTE

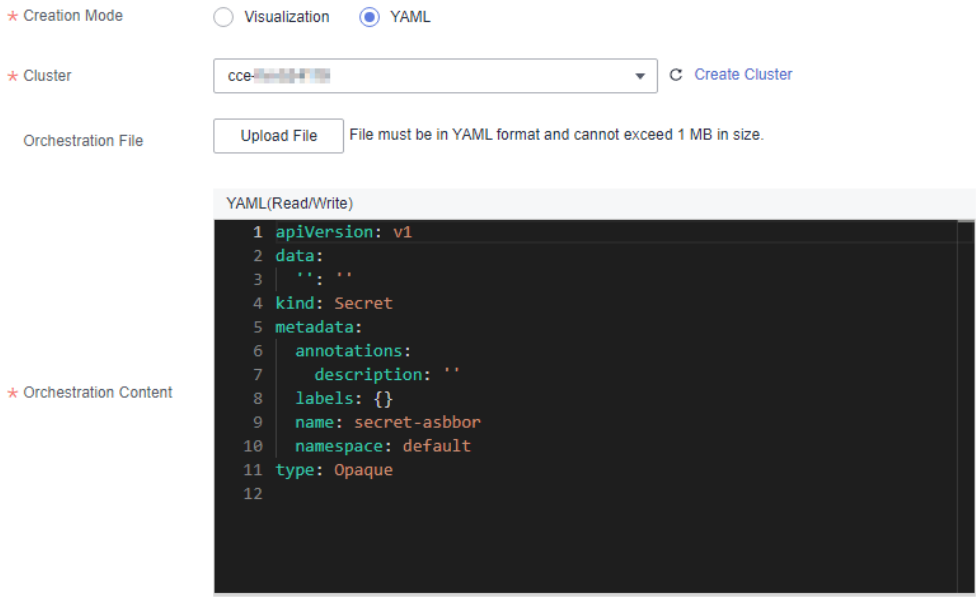
  To create a secret by uploading a file, ensure that the secret description file has been created. ServiceStage supports files in YAML format. For details, see **Secret Resource File Configuration**.

  a. Select a cluster from the **Cluster** drop-down list.

  b. (Optional) Click **Upload File**, select the created secret file, and click **Open**. Wait until the file is uploaded.

  c. Write or modify the secret resource file in **Orchestration content**.

Figure 3-11 Setting secret parameters in YAML mode



Step 5 Click **Create Secret**.

The new secret is displayed in the secret list.

----End

Follow-Up Operations

After a secret is created, you can search for, view, update, and delete the secret by referring to **Table 3-5**.

NOTE

- Deleted items cannot be restored. Exercise caution when performing this operation.
- The secret list contains system secrets, which can only be viewed and cannot be modified or deleted.

Table 3-5 Secret management operations

Operation	Description
Searching for a secret	1. Select the namespace to which the secret belongs from the namespace drop-down list. 2. Enter a secret name in the search box.
Viewing a secret	Click **Show YAML** in the **Operation** column of the target secret to view the content of the YAML file of the secret.
Updating a secret	1. Click **Update** in the **Operation** column of the target secret. 2. Modify the information according to **Table 3-4**. 3. Click **Modify Secret**.

Operation	Description
Deleting a secret	1. Click **Delete** in the **Operation** column of the target secret. 2. In the displayed dialog box, click **OK**.
Deleting secrets in batches	1. Select the secrets to be deleted. 2. Click **Delete Key**. 3. In the displayed dialog box, click **OK**.

Secret Resource File Configuration

This section provides examples of configuring secret resource description files. For example, you can retrieve the username and password for an application through a secret.

username: my-username

password: my-password

The following shows the content of a secret file. The value must be encoded using Base64. For more information, see **Base64 Encoding**.

```
apiVersion: v1
kind: Secret
metadata:
  name: mysecret          #Secret name.
  namespace: default      #Namespace. The default value is default.
data:
  username: ******    #The value must be Base64-encoded.
  password: ******  #The value must be Base64-encoded.
type: Opaque      #You are advised not to change this parameter value.
```

Base64 Encoding

To encrypt a string using Base64, run the **echo -n**'*Content to be encoded*' **| base64** command in the local Linux environment. Example:

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

Where,

- **3306** is the content to be encoded.
- **MzMwNg==** is the encoded content.

3.4 Managing Resources

After an environment is created, the compute resources (such as ECSs and CCE clusters), network resources (such as ELB instances and EIPs), and middleware (such as DCS instances, RDS instances, and CSE engines) need to be managed together to form an environment.

For details about how to manage CCE clusters in the Kubernetes environment, see **CCE Resource Management**.

Prerequisites

The following resources to be managed are required.

- The ECSs to be managed have been created and are in the running state. The ECSs must be in the same VPC as the target environment and cannot be managed by other environments.

  ECSs are used to deploy and run components on VMs. For details about how to create ECSs, see **Purchasing ECSs**.

- The AS groups to be managed have been created. The AS groups must be in the same VPC as the target environment and cannot be managed by other environments. In addition, the AS groups contain ECSs.

  AS groups are used to deploy and run components on VMs. For details about how to create AS groups, see **Creating an AS Group**.

  AS groups cannot be managed in LA-Sao Paulo1 and LA-Mexico City2.

- The ELBs to be managed have been created. The ELBs must be in the same VPC as the target environment.

  ELBs are used to access services provided by components in ELB mode. For details about how to create ELBs, see **Creating a Shared Load Balancer**.

- The EIPs to be managed have been created.

  EIPs are used to access services provided by a component through public network. For details about how to create EIPs, see **Assigning an EIP**.

- The DCSs to be managed have been created. The DCSs must be in the same VPC as the target environment.

  DCSs are used to read environment variables to obtain distributed cache information during application running. For details about how to create DCSs, see **Buying a DCS Redis Instance**.

- The RDS MySQL DB instances to be managed have been created. The RDSs must be in the same VPC as the target environment.

  RDSs are used for persistent storage of application data. For details about how to create RDSs, see **Step 1: Buy a DB Instance**.

- The CSEs to be managed have been created. If the CSEs and the environment are in different VPC, correctly configure the VPC connectivity.

  CSEs are used to connect microservices running in the environment to the engine to implement microservice registry and discovery, service governance, and configuration management. For details about how to create CSEs, see **Creating a Microservice Engine**.

Procedure

Step 1 Log in to ServiceStage.

Step 2 On the **Environment Management** page, click the target environment.

Step 3 In the left pane of the **Resource Settings** page, manage resources in the environment by referring to the following table.

Resource Type	Resource Name	Operation
Compute	Cloud Container Engine	See **Binding a CCE Cluster**. For details about how to manage CCE clusters in the Kubernetes environment, see **CCE Resource Management**.
	Elastic Cloud Server	1. Choose **Compute** > **Elastic Cloud Server**. 2. Click **Manage Resource**. 3. Select the ECS to be managed. 4. Click **OK**. **NOTE** • In the same VPC, ECSs that have been managed by other environments cannot be managed again. • In a VM environment, if **Agent Status** of the managed ECSs is **Install Agent**, install the agent by referring to **Installing a VM Agent**.
	Auto Scaling	1. Choose **Compute** > **Auto Scaling**. 2. Click **Manage Resource**. 3. Select the AS resource to be managed. 4. Click **OK**. **NOTE** In the same VPC, AS groups that have been managed by other environments cannot be managed again.
Networking	Elastic Load Balance	1. Choose **Networking** > **Elastic Load Balance**. 2. Click **Manage Resource**. 3. Select the load balancer to be managed. 4. Click **OK**.
	Elastic IP	1. Choose **Network** > **Elastic IP**. 2. Click **Manage Resource**. 3. Select the EIP to be managed. 4. Click **OK**.
Middleware	Distributed Cache Service	1. Choose **Middleware** > **Distributed Cache Service**. 2. Click **Manage Resource**. 3. Select the DCS instance to be managed. 4. Click **OK**.
	Cloud Service Engine	1. Choose **Middleware** > **Cloud Service Engine**. 2. Click **Manage Resource**. 3. Select the microservice to be managed. 4. Click **OK**.

Resource Type	Resource Name	Operation
	Relational Database Service	1. Choose **Middleware** > **Relational Database Service**. 2. Click **Manage Resource**. 3. Select the RDS instance to be managed. 4. Click **OK**.

----End

3.5 Removing Managed Resources

You can remove a managed resource that is no longer used.

Procedure

Step 1 Log in to ServiceStage.

Step 2 On the **Environment Management** page, click the target environment.

Step 3 In the **Resource Settings** area, choose the resources from **Compute**, **Networking**, or **Middleware**.

Step 4 In the managed resource list, perform the following operations:

- To remove resources in batches, select the resources to be deleted and click **Remove Resource**.
- To remove a single resource, locate the resource to be removed and click **Remove** in the **Operation** column.

----End

3.6 Upgrading a VM Agent

In a VM environment, if **Agent Status** of a managed ECS is **Agent online** and a new agent version is available, the VM agent can be upgraded.

Procedure

Step 1 Log in to ServiceStage.

Step 2 On the **Environment Management** page, click the target environment.

Step 3 In the **Resource Settings** area, choose **Elastic Cloud Server** from **Compute**.

Step 4 In the managed resource list, select the target resource and click **Upgrade Agent**.

Step 5 Click **OK**.

After **Agent Status** changes from **Agent upgrading** to **Agent online**, the VM agent has been upgraded.

----End

3.7 Restarting a VM Agent

In a VM environment, if **Agent Status** of a managed ECS is **Agent online**, the VM agent can be restarted.

Procedure

Step 1 Log in to ServiceStage.

Step 2 On the **Environment Management** page, click the target environment.

Step 3 In the **Resource Settings** area, choose **Elastic Cloud Server** from **Compute**.

Step 4 In the managed resource list, select the target resource and click **Restart Agent**.

Step 5 Click **OK**.

After **Agent Status** changes from **Agent restarting** to **Agent online**, the VM agent has been restarted.

----End

3.8 Modifying an Environment

This topic describes how to modify an environment.

Procedure

Step 1 Log in to ServiceStage.

Step 2 On the **Environment Management** page, use either of the following methods to go to the **Edit Environment** page:

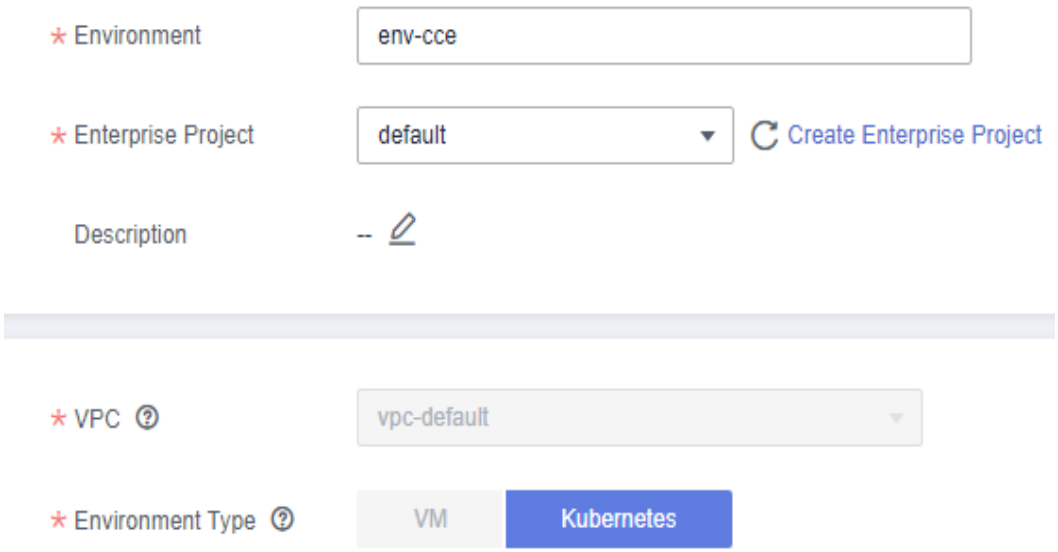
- Select the target environment and click **Edit** in the **Operation** column.
- Click the target environment. On the displayed environment details page, click **Edit**.

Step 3 Edit the environment information by referring to the following table.

Parameter	Description
Environment	Environment name.
Enterprise Project	Enterprise projects let you manage cloud resources and users by project. It is available after you **enable the enterprise project function**.

Parameter	Description
Description	Environment description. 1. Click and enter the environment description. 2. Click to save the description.

Figure 3-12 Editing an environment



Step 4 Click **Save**.

----End

3.9 Deleting an Environment

You can delete an environment that is no longer used.

NOTE

- Before deleting an environment, ensure that no component is deployed in the environment or the deployed components have been deleted. For details, see **Deleting a Component**.
- Deleting an environment does not delete managed resources in the environment.

Procedure

Step 1 Log in to ServiceStage.

Step 2 On the **Environment Management** page, use either of the following methods to delete an environment:

- Select the target environment and click **Delete** in the **Operation** column.
- Click the target environment. On the displayed environment details page, click **Delete**.

Step 3 Click **OK**.

----End

3.10 Installing a VM Agent

To deploy a component to a VM, you need to install the VM agent. After the host is managed, the backend can communicate with the host.

For details about the VM agent status and description, see **Table 3-6**.

Table 3-6 VM agent status description

Agent Status	Description
Agent uninstalled	The VM agent is not installed on the ECS node. You need to install the VM agent.
Agent online	The VM agent has been installed on the ECS node and is running properly.
Agent offline	The VM agent has been installed on the ECS node, but is offline and cannot work properly. For details about how to handle agent offline, see **What Should I Do If the VM Agent Is Offline?**
Agent upgrading	The VM agent has been installed on the ECS node and is being upgraded.
Agent upgrade failed	The VM agent has been installed on the ECS node and fails to be upgraded.
Agent restarting	The VM agent has been installed on the ECS node and is restarting.

The VM agent supports multiple OSs. You need to create an image by referring to **Table 3-7**, use the created image to create an ECS, and install the VM agent.

Table 3-7 OSs and versions supported by the VM agent

OS	Version	Description
EulerOS	• 2.2 64bit • 2.3 64bit • 2.5 64bit • 2.8 64bit	• For Linux x86_64 servers, all the listed OSs and versions are supported. • For Linux ARM servers, all the listed OSs and versions except CentOS 7.3 and earlier are supported.

OS	Version	Description
CentOS	• 6.5 64bit • 6.8 64bit • 6.9 64bit • 6.10 64bit • 7.2 64bit • 7.3 64bit • 7.4 64bit • 7.5 64bit • 7.6 64bit • 7.7 64bit • 7.8 64bit • 7.9 64bit	
Fedora	• 29 64bit • 30 64bit	
openEuler	20.03 64bit	

This section describes how to install the VM agent on a single VM.

Procedure

Step 1 Log in to ServiceStage.

Step 2 On the **Environment Management** page, click the target environment of the VM type.

Step 3 In the **Resource Settings** area, choose **Elastic Cloud Server** from **Compute**.

Step 4 In the managed resource list, locate the VM where the agent is to be installed and click **Install Agent** in the **Agent Status** column.

Step 5 Select an authorization mode.

Authorize the agent to use your authentication information to obtain the deployment, upgrade, start, and stop tasks of an application and execute the task.

You can use agency or AK/SK to perform authorization. Agency is recommended.

- Select **Agency** for **Authorization Model**:

  Click , select an agency, and click .

  For details about how to create an agency, see **Creating an Agency**.

  NOTE

  When creating an agency, you need to delegate the op_svc_ecs account to manage resources or ECS cloud service to access other cloud resources, and select the Tenant Administrator policy in the corresponding region.

- Select **AKSK** for **Authorization Model**:

  For security purposes, obtain and use the AK and SK with the ServiceStage Development permission. The account and the account used for installing the VM agent must belong to the same user.

  For details about how to obtain the AK/SK, see **Access Keys**.

Step 6 Copy the command automatically generated in the lower part of the window, that is, the agent installation command.

Example command for the **Agency** model:

```
export AGENT_INSTALL_URL=https://${Region_Name}-servicestage-vmapp.obs.${Region_Name}.${Domain_Name}/vmapp/agent/agent-install.sh;if [ -f `which curl` ];then curl -# -O -k ${AGENT_INSTALL_URL};else wget --no-check-certificate ${AGENT_INSTALL_URL};fi;bash agent-install.sh ${Project_ID} ${Version} ${Region_Name} ${Flag}
```

Example command for the **AKSK** model:

```
export AGENT_INSTALL_URL=https://${Region_Name}-servicestage-vmapp.obs.${Region_Name}.${Domain_Name}/vmapp/agent/agent-install.sh;if [ -f `which curl` ];then curl -# -O -k ${AGENT_INSTALL_URL};else wget --no-check-certificate ${AGENT_INSTALL_URL};fi;bash agent-install.sh ${AK}${SK} ${Project_ID} ${Version} ${Region_Name} ${Flag}
```

- **AGENT_INSTALL_URL** indicates the installation address of the agent.
- If the **Agency** model is used, the ECS node has the permission to obtain the temporary AK/SK of the user. In this case, you do not need to enter AK/SK in the command.
- **${AK}** and **${SK}** indicate access keys.
- **${Region_Name}** indicates a region name.
- **${Domain_Name}** indicates the global domain name.
- **${Project_ID}** indicates a project ID. For details about how to obtain a project ID, see **API Credentials**.
- **${Version}** is the version number. Use **latest** to automatically download the latest version.
- **${Flag}** is a Boolean value, indicating whether to automatically add the application access port. **true** indicates yes and **false** indicates no.

Step 7 Log in to the VM and run the installation command.

NOTE

If the VM agent fails to be installed, see **What Should I Do If I Don't See the VM Agent After Installing It?**

----End

4 Application Management

4.1 Creating an Application

An application is a service system with complete functions and consists of one or more components related to features.

For example, the weather forecast is an application that contains the weather and forecast components. ServiceStage organizes multiple components by application, and supports quick cloning of applications in different environments.

ServiceStage allows a single user to create a maximum of 1000 applications under the same project.

Procedure

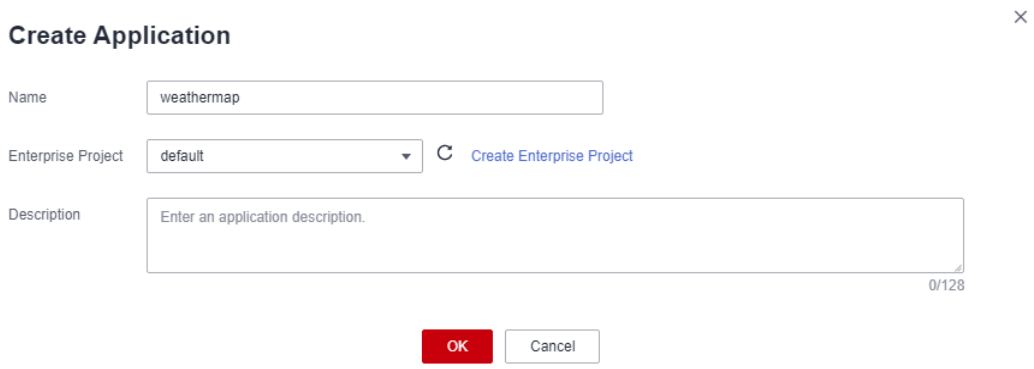
Step 1 Log in to ServiceStage.

Step 2 Choose **Application Management** > **Create Application** and configure the application. Parameters marked with an asterisk (*) are mandatory.

Parameter	Description
*Name	Enter an application name. The application name must be unique.
Enterprise Project	Select an enterprise project. Enterprise projects let you manage cloud resources and users by project. It is available after you **enable the enterprise project function**.

Parameter	Description
*Tags **NOTE** Set tags in **CN East2**.	Tags are used to identify cloud resources. When you have multiple cloud resources of the same type, you can use tags to classify them based on usage, owner, or environment. If your organization has configured tag policies for ServiceStage, add tags to trackers based on the policies. If a tag does not comply with the tag policies, application creation may fail. Contact your administrator to learn more about tag policies. Each application can have a maximum of 20 tags. 1. Enter the key and value of the tag. It is recommended that you **create a predefined tag** on TMS to add the same tag to different resources. 2. Click **OK**.
Description	Enter the application description.

Figure 4-1 Creating an application



Step 3 Click **OK**.

----End

4.2 Viewing Application Overview

After the application is created, you can go to the **Overview** page to view the application overview.

Procedure

Step 1 Log in to ServiceStage.

Step 2 Choose **Application Management**.

Step 3 Click the target application. On the displayed **Overview** page, view the application details.

If a component has been created and deployed under the application, you can view the component details in the component list.

----End

4.3 Managing Application Environment Variables

Environment variables are parameters set in the system or user applications. You can obtain the values of environment variables by calling APIs. During deployment, parameters are specified through environment variables instead of in the code, which makes the deployment flexible.

Environment variables added to an application are global environment variables and take effect for all components of the application.

For details about how to add environment variables for a specific component, see **Configuring Environment Variables of a Component**.

Manually Adding an Application Environment Variable

Step 1 Log in to ServiceStage.

Step 2 Choose **Application Management**.

Step 3 Click the target application. The **Overview** page is displayed.

Step 4 In the navigation pane on the left, click **Environment Variables**.

Step 5 Select a created environment from the drop-down list.

Step 6 Click **Add Environment Variable** and set **Variable Name** and **Variable/Variable Reference**.

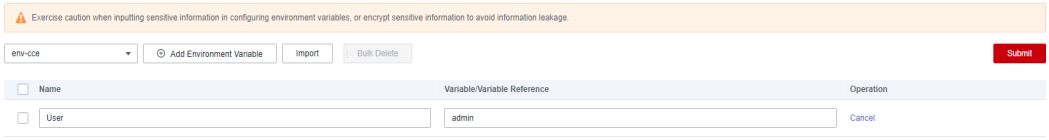
Where,

- **Variable Name** is the name of an application environment variable and must be unique.
- **Variable/Variable Reference** is the value of the application environment variable.

For example, set **Variable Name** to **User** and **Variable/Variable reference** to **admin**. That is, when the program code reads the **User** environment variable, **admin** is obtained. For example, you can start subprocesses as the admin user and read files as the admin user. The actual execution effect depends on the code.

Step 7 Click **Submit**.

Figure 4-2 Manually adding an application environment variable



----End

Importing the Application Environment Variable File

Step 1 Log in to ServiceStage.

Step 2 Choose **Application Management**.

Step 3 Click the target application. The **Overview** page is displayed.

Step 4 In the navigation pane on the left, click **Environment Variables**.

Step 5 Select a created environment from the **Environment** drop-down list.

Step 6 Click **Import** and select the environment variable file created locally.

The file to be imported must be a key-value pair mapping file in JSON or YAML format and in character string format. For example:

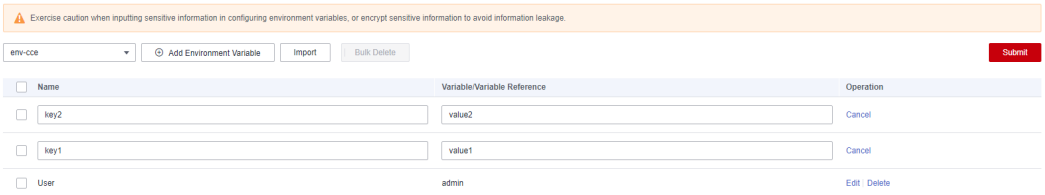
```
{"key1": "value1", "key2": "value2"}
```

Where,

- **key1** and **key2** are the names of application environment variables and must be unique.
- **value1** and **value2** are the values of application environment variables.

Step 7 Click **Submit**.

Figure 4-3 Importing the application environment variable file



----End

Editing an Application Environment Variable

Step 1 Log in to ServiceStage.

Step 2 Choose **Application Management**.

Step 3 Click the target application. The **Overview** page is displayed.

Step 4 In the navigation pane on the left, click **Environment Variables**.

Step 5 Select a created environment from the **Environment** drop-down list.

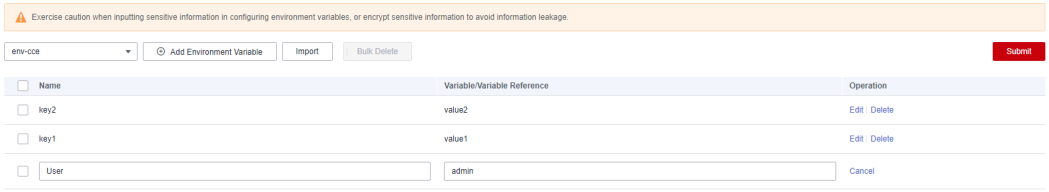
Step 6 Select the variable to be edited and click **Edit** in the **Operation** column.

Step 7 Reset **Variable Name** and **Variable/Variable Reference**.

- **Variable Name** is the name of an application environment variable and must be unique.
- **Variable/Variable Reference** is the value of the application environment variable.

Step 8 Click **Submit**.

Figure 4-4 Editing an application environment variable



----End

Deleting an Application Environment Variable

Step 1 Log in to ServiceStage.

Step 2 Choose **Application Management**.

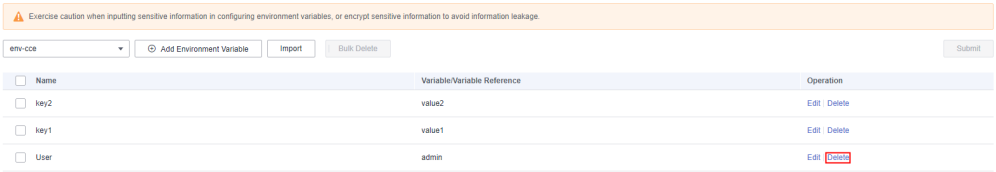
Step 3 Click the target application. The **Overview** page is displayed.

Step 4 In the navigation pane on the left, click **Environment Variables**.

Step 5 Select a created environment from the **Environment** drop-down list.

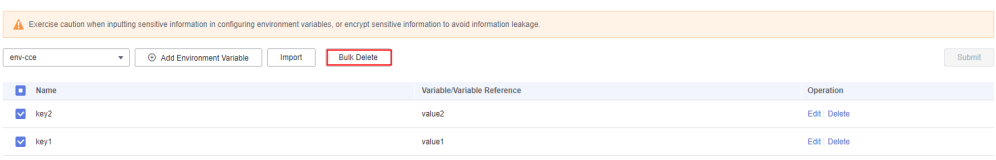
- To delete a single application environment variable, select the target variable and click **Delete** in the **Operation** column.

  Figure 4-5 Deleting a single application environment variable

- To delete application environment variables in batches, select the target variables and click **Bulk Delete**.

  Figure 4-6 Deleting application environment variables in batches

Step 6 In the displayed dialog box, click **OK**.

----End

Follow-Up Operations

After the application environment variables are changed, you can:

- Make the changed application environment variables take effect for a specified component of the application by **Upgrading a Single Component**.
- Make the changed application environment variables take effect for multiple or all components of the application by **Upgrading Components in Batches**.

4.4 Editing an Application

You can modify the application name and description.

Procedure

Step 1 Log in to ServiceStage.

Step 2 Choose **Application Management**.

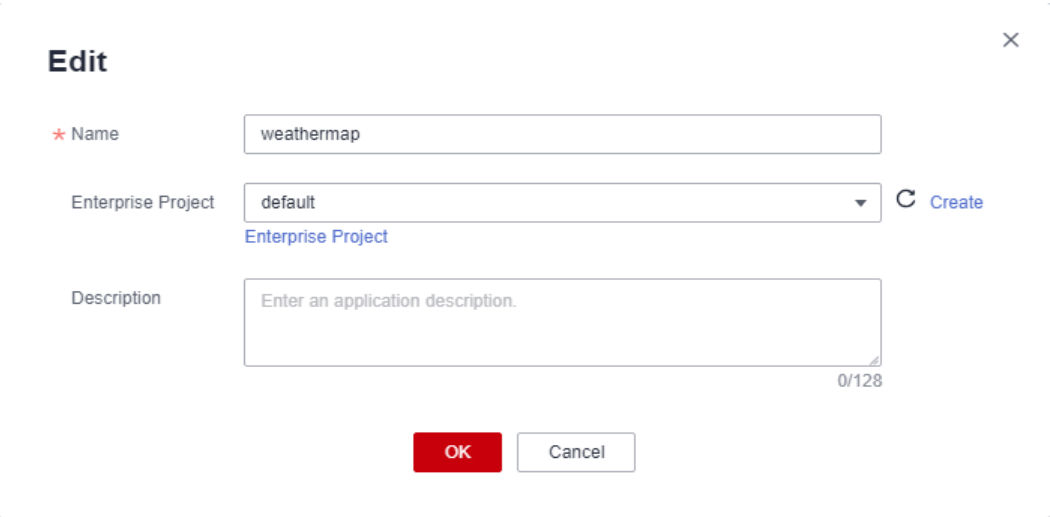
Step 3 Use either of the following methods to edit an application:

- Select the target application and click **Edit** in the **Operation** column.
- On the **Application Management** page, click the target application. On the displayed **Overview** page, click **Edit** in the upper part of the page.

Step 4 Configure the application again by referring to the following table.

Parameter	Description
Name	Enter an application name. The application name must be unique.
Enterprise Project	Select an enterprise project. Enterprise projects let you manage cloud resources and users by project. It is available after you **enable the enterprise project function**.
Description	Enter the application description.

Figure 4-7 Editing an application



Step 5 Click **OK**.

----End

4.5 Deleting an Application

You can delete an application that is no longer used.

NOTICE

Deleted applications cannot be restored. Exercise caution when performing this operation.

Prerequisites

Before deleting an application, delete all components of the application. For details, see **Deleting a Component**.

Procedure

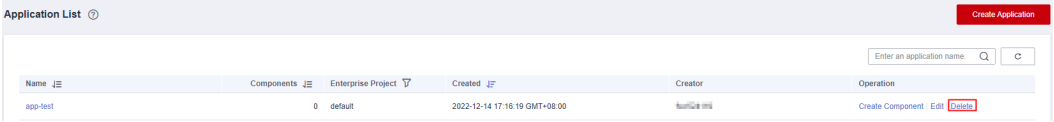
Step 1 Log in to ServiceStage.

Step 2 Choose **Application Management**.

Step 3 Use either of the following methods to delete an application:

- Select the target application and click **Delete** in the **Operation** column.
- On the **Application Management** page, click the target application. On the displayed **Overview** page, click **Delete** in the upper part of the page.

Figure 4-8 Deleting an application



Step 4 In the displayed dialog box, click **OK**.

----End

5 Component Management

5.1 Component Overview

A component is a service feature implementation of an application. It is carried by code or software packages and can be independently deployed and run.

After creating an application on ServiceStage, you can create and deploy components in the application. A maximum of 1000 components can be created for an application.

You can set the component technology stack and component source based on service requirements to create and deploy components.

Technology Stack

A technology stack includes the operating system, framework, and runtime on which component running depends. It consists of attributes such as the stack name, type, status, and version. The version number complies with the **semantic versioning specifications**.

ServiceStage provides and manages the stack lifecycle. You only need to focus on service development to improve application hosting experience.

The lifecycle phases of the technology stack are defined as follows:

- Preview: The beta version is released.
- General Availability (GA): The official version is released.
- End of Life (EOL): The lifecycle ends.

The technology stack status is defined as follows:

- Preview: The stack is in the Preview phase.
- Supported: The stack is in the GA phase.
- Deprecated: The stack is in the GA phase but the EOL announcement has been released, or the stack is not recommended by ServiceStage.

For details about the technology stack, see **Table 5-1**.

Table 5-1 Technology stack information

Technology Stack	Type	Status	Release Description	Component Source and Deployment Mode
OpenJDK8	Java	Supported	• OpenJDK-8u312b07: **Release Note** • Image OS: EulerOS 2.9.8 OpenJDK-8u312b07: **Release Note**	• The component source is source code or JAR package, and container-based deployment is supported. For details, see **Deploying a Component**. • The component source is JAR package and VM-based deployment is supported. For details, see **Deploying a Component**.

Technology Stack	Type	Status	Release Description	Component Source and Deployment Mode
OpenJDK11	Java	Supported	• **BiSheng JDK 11.0.19**: for the AArch64 architecture • **OpenJDK 11.0.2**: for the x86_64 architecture • Image OS: EulerOS 2.9.8	• The component source is source code or JAR package, and container-based deployment is supported. For details, see **Deploying a Component**. • The component source is JAR package and VM-based deployment is supported. For details, see **Deploying a Component**.
			• **BiSheng JDK 11.0.19**: for the AArch64 architecture • **OpenJDK 11.0.2**: for the x86_64 architecture	

Technology Stack	Type	Status	Release Description	Component Source and Deployment Mode
OpenJDK17	Java	Supported	• OpenJDK 17.0.2 • Image OS: EulerOS 2.9.8 OpenJDK 17.0.2	• The component source is source code or JAR package, and container-based deployment is supported. For details, see Deploying a Component. • The component source is JAR package and VM-based deployment is supported. For details, see Deploying a Component.

Technology Stack	Type	Status	Release Description	Component Source and Deployment Mode
Tomcat8/ OpenJDK8	Tomcat	Supported	• OpenJDK-8u312 b07: **Release Note** • Tomcat-8.5.75: **Release Note** • Image OS: EulerOS 2.9.8	• The component source is source code or WAR package, and container-based deployment is supported. For details, see **Deploying a Component**. • The component source is WAR package and VM-based deployment is supported. For details, see **Deploying a Component**.
			• OpenJDK-8u312 b07: **Release Note** • Tomcat-8.5.75: **Release Note**	

Technology Stack	Type	Status	Release Description	Component Source and Deployment Mode
Tomcat9/ OpenJDK8	Tomcat	Supported	• OpenJDK-8u312 b07: **Release Note** • Tomcat-9.0.58: **Release Note** • Image OS: EulerOS 2.9.8	• The component source is source code or WAR package, and container-based deployment is supported. For details, see **Deploying a Component**. • The component source is WAR package and VM-based deployment is supported. For details, see **Deploying a Component**.
			• OpenJDK-8u312 b07: **Release Note** • Tomcat-9.0.58: **Release Note**	

Technology Stack	Type	Status	Release Description	Component Source and Deployment Mode
Node.js8	Node.js	Supported	• Nodejs-v8.11.3: **Release Note** • Image OS: EulerOS 2.9.8 Nodejs-v8.11.3: **Release Note**	• The component source is source code or ZIP package, and container-based deployment is supported. For details, see **Deploying a Component**. • The component source is ZIP package and VM-based deployment is supported. For details, see **Deploying a Component**.

Technology Stack	Type	Status	Release Description	Component Source and Deployment Mode
Node.js14	Node.js	Supported	• Nodejs-v14.18.1: **Release Note** • Image OS: EulerOS 2.9.8 Nodejs-v14.18.1: **Release Note**	• The component source is source code or ZIP package, and container-based deployment is supported. For details, see **Deploying a Component**. • The component source is ZIP package and VM-based deployment is supported. For details, see **Deploying a Component**.

Technology Stack	Type	Status	Release Description	Component Source and Deployment Mode
Node.js16	Node.js	Supported	• Nodejs-v16.13.0: **Release Note** • Image OS: EulerOS 2.9.8 Nodejs-v16.13.0: **Release Note**	• The component source is source code or ZIP package, and container-based deployment is supported. For details, see **Deploying a Component**. • The component source is ZIP package and VM-based deployment is supported. For details, see **Deploying a Component**.
Docker	Docker	-	Supported by CCE. For details, see **Kubernetes Release Notes**.	The component source is image package, and container-based deployment is supported. For details, see **Deploying a Component**.
Python3	Python	-	-	The component source is source code or ZIP package, and container-based deployment is supported. For details, see **Deploying a Component**.

Technology Stack	Type	Status	Release Description	Component Source and Deployment Mode
Php7	Php	-	-	The component source is source code or ZIP package, and container-based deployment is supported. For details, see **Deploying a Component**.

Component Source

Component Source	Description
Source Code Repository	Create authorization by referring to **Authorizing a Repository** and set the code source.
JAR package	The following upload modes are supported: 1. Select the corresponding software package from the SWR software repository. Upload the software package to the software repository in advance. For details, see **Uploading the Software Package**. 2. Select the corresponding software package from OBS. Upload the software package to the OBS bucket in advance. For details, see **Uploading an Object**.

Component Source	Description
WAR package	The following upload modes are supported: 1. Select the corresponding software package from the SWR software repository. Upload the software package to the software repository in advance. For details, see **Uploading the Software Package**. 2. Select the corresponding software package from OBS. Upload the software package to the OBS bucket in advance. For details, see **Uploading an Object**.
ZIP package	The following upload modes are supported: 1. Select the corresponding software package from the SWR software repository. Upload the software package to the software repository in advance. For details, see **Uploading the Software Package**. 2. Select the corresponding software package from OBS. Upload the software package to the OBS bucket in advance. For details, see **Uploading an Object**.

Component Source	Description
Image package	Containerized applications need to be created based on images. **My Images** (private images), **Open Source Images**, **Shared Images**, and **Third-party Images** are supported. • If you select **My Images**, upload the image to the image repository in advance. For details, see **Uploading an Image**. • If you select **Third-party Images**, ensure that you have obtained the address of the third-party image. The format of the image address is as follows: *{IP address of the third-party image repository}:{Port number for accessing the third-party image repository}/{Image storage path}/{Image name}:{Image tag}* Alternatively: *{Image name}:{Image tag}* If the image tag is not specified, the latest version is used by default. Currently, only third-party public images can be obtained.

Deploying a Component

Deployment Mode	Description
Container-based deployment	CCE is a highly scalable, enterprise-class hosted Kubernetes service for you to run containers and applications. With CCE, you can easily deploy, manage, and scale containerized applications on the cloud platform.
VM-based deployment	A VM, or an HECS, is a basic computing unit that consists of vCPUs, memory, OS, and EVS disks. After creating an ECS, you can use it like using your local computer or physical server to deploy components.

5.2 Creating and Deploying a Component

This section describes how to create and deploy a component on ServiceStage.

ServiceStage allows you to create components with the same name in the same application. During component deployment, for components with the same name:

- Components with the same name in the same application cannot be deployed in the same environment.
- Components with the same name in different applications can be deployed in the same environment.

Prerequisites

1. An application has been created because components can only be added to applications. For details, see **Creating an Application**.
2. An environment has been created and resources have been managed because components need to be deployed in a specified environment. For details, see **Environment Management**.
3. An organization has been created because the image generated by the component deployment needs to be managed based on an organization. For details, see **Creating an Organization**.
4. (Optional) A namespace has been created, if you want to create and deploy a component in a Kubernetes environment. For details, see **Creating a Namespace**.
5. If you create a component based on a source code repository, create repository authorization first. For details, see **Authorizing a Repository**.
6. If you create a component based on a software package, the software package has been uploaded to the SWR software repository or OBS bucket.
   - Upload the software package to the software repository. For details, see **Uploading the Software Package**.
   - Upload the software package to the OBS bucket. For details, see **Uploading an Object**.

   NOTE

   If the software package fails to be uploaded, see **What If a Software Package Fails to Be Uploaded?**

Procedure

Step 1 Log in to ServiceStage.

Step 2 Use any of the following methods to go to the **Create Component** page:

- Choose **Component Management** > **Create Component**.
- On the **Application Management** page, select the application for which you want to create a component, and click **Create Component** in the **Operation** column.
- On the **Application Management** page, click the application for which you want to create a component. On the displayed **Overview** page, click **Create Component**.

Step 3 In the **Basic Information** area, configure the component by referring to the following table. Parameters marked with an asterisk (*) are mandatory.

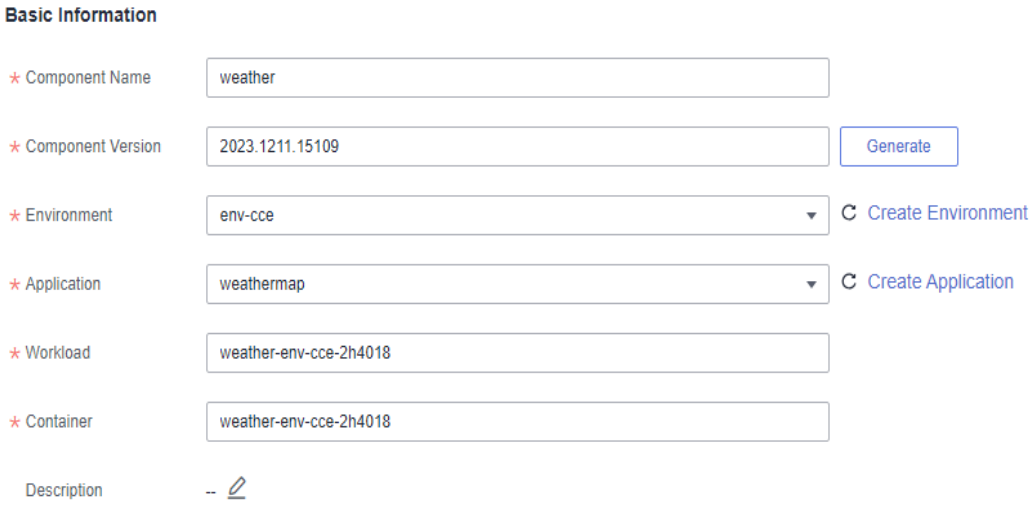
Parameter	Description
*Component Name	Name of a component, which cannot be changed after the component is created and deployed. Kubernetes environment: ● Components with the same name in different applications can be deployed in the same environment. ● Components with the same name in the same application can be deployed in different environments. VM environment: ● Components with the same name in different applications can be deployed in the same environment. ● Components with the same name in the same application can be deployed in different environments.
*Component Version	Component version number, which can be automatically generated or customized. ● Automatically-generated: Click **Generate**. By default, the version number is the time when you click **Generate**. The format is yyyy.mmdd.hhmms, where **s** is the ones place of the second in the timestamp. For example, if the timestamp is 2022.0803.104321, the version number is 2022.0803.10431. ● Customized: Enter a value in the format of A.B.C or A.B.C.D. A, B, C, and D are natural numbers. For example, 1.0.0 or 1.0.0.0.
*Environment	Component deployment environment.
*Deployment Mode	Component deployment mode. This parameter is mandatory when **Environment** is set to **VM + Kubernetes**. ● Container-based deployment: CCE is a highly scalable, enterprise-class hosted Kubernetes service for you to run containers and applications. With CCE, you can easily deploy, manage, and scale containerized applications on the cloud platform. ● VM-based deployment: A VM, or an HECS, is a basic computing unit that consists of vCPUs, memory, OS, and EVS disks. After creating an ECS, you can use it like using your local computer or physical server to deploy components. **NOTE** If CCE clusters and VMs are managed in an earlier version, the environment type is **VM + Kubernetes** after the upgrade to the current version.
*Application	Application to which the component belongs.

Parameter	Description

*Workload	This parameter is mandatory when the component is deployed based on CCE.
	The workload name is automatically generated by default, and can also be customized.
	The default workload name consists of the **Component Name** and **Environment** you set, and six random characters generated by the system. The total length cannot exceed 62 characters.
	● If the **Component Name** contains 55 characters or fewer, the default workload name is:
	`All of the component name-All or part of the environment name-Six random characters generated by the system`
	Underscores (_) in the name will be replaced with hyphens (-), and uppercase letters will be converted to lowercase letters.
	For example, if the component name is **Com_calc** and the environment is **env-cce**, the default workload name is:
	`com-calc-env-cce-1uw8g0`
	● If the **Component Name** contains more than 55 characters, the default workload name is:
	`First 55 characters of the component name-Six random characters generated by the system`
	Underscores (_) in the name will be replaced with hyphens (-), and uppercase letters will be converted to lowercase letters.
	For example, if the component name is **C_aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa**, the default workload name is:
	`c-aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa-1uw8g0`
	You can also customize the workload name. The name contains 1 to 63 characters, including lowercase letters, digits, and hyphens (-), and must start with a lowercase letter and end with a lowercase letter or digit.
	The workload name cannot be changed after the component is created and deployed.

*Container	This parameter is mandatory when the component is deployed based on CCE. By default, the container name is the same as the workload name. You can also customize the container name. The name contains 1 to 63 characters, including lowercase letters, digits, and hyphens (-), and must start with a lowercase letter and end with a lowercase letter or digit.
Description	Component description. 1. Click to enter the component description. 2. Click to save the component description. Click to cancel the setting.

Figure 5-1 Setting the basic component information



Step 4 In the **Component Package** area, configure the component package by referring to the following table. Parameters marked with an asterisk (*) are mandatory.

Parameter	Description
*Stack	1. Select a component technology stack type based on the component deployment mode. For details, see **Table 5-1**. 2. Select a technology stack from the **Name** drop-down list. 3. (Optional) Set **JVM** to configure the memory parameter size during Java code running. This parameter is available when a Java or Tomcat technology stack is selected. Click **Stack Settings** and set **JVM**, for example, **-Xms256m -Xmx1024m**. Multiple parameters are separated by spaces. 4. (Optional) Set **Tomcat** parameters to configure the parameters such as Tomcat request path and port number. This parameter is available when a Tomcat technology stack is selected. a. Click **Stack Settings** and select **Tomcat**. The **Tomcat** dialog box is displayed. b. Click **Use Sample Code** and edit the template file based on service requirements. **NOTE** In Tomcat configuration, the default **server.xml** configuration is used. The context path is **/**, and no application path is specified. If you need to customize an application path, customize the Tomcat context path by referring to **How Do I Customize a Tomcat Context Path?** c. Click **OK**.
*Source Code/ Software Package	If you select **Source code repository**, create authorization by referring to **Authorizing a Repository** and set the code source. If you select a software package, the software package type supported by the component source is determined by the selected technology stack type. For details, see **Table 5-1**.

Parameter	Description
*Upload Method	If the component source is software package or image package, select an uploaded software package or image package. For details about the upload method, see **Component Source**.

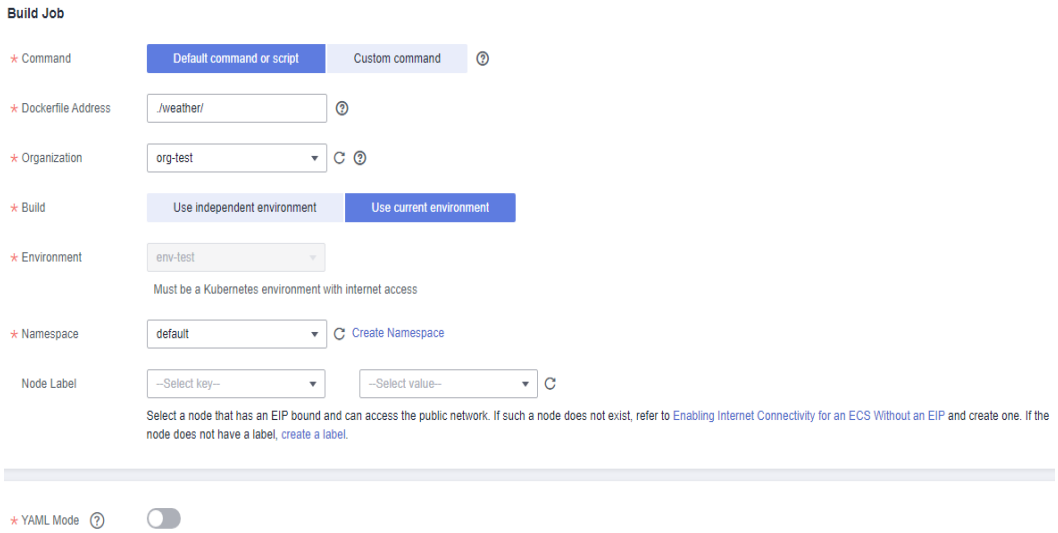
Step 5 In the **Build Job** area, set build parameters by referring to the following table. Parameters marked with an asterisk (*) are mandatory.

This area is mandatory when the component is deployed based on CCE and the selected technology stack type is Java, Tomcat, Node.js, Python, or PHP.

Parameter	Description
*Command	If the component source is **Source code repository**, set **Command** based on service requirements. • **Default command or script**: preferentially executes **build.sh** in the **root** directory. If **build.sh** does not exist, the code will be compiled using the common method of the selected language. Example: **mvn clean package** for Java. • **Custom command**: Commands are customized using the selected language. Alternatively, the default command or script is used after **build.sh** is modified. **NOTICE** – If **Custom command** is selected, exercise caution when inputting sensitive information in the **echo**, **cat**, or **debug** command, or encrypt sensitive information to avoid information leakage. – To run the compilation command in the project subdirectory, you need to go to the project subdirectory and then run other script commands. For example: **cd ./weather/** **mvn clean package**
*Dockerfile Address	If the component source is **Source code repository**, set **Dockerfile Address** based on service requirements. **Dockerfile Address** is the directory where the Dockerfile is located relative to the root directory (./) of the project. The Dockerfile is used to build an image. If **Dockerfile Address** is not specified, the system searches for the Dockerfile in the root directory of the project by default. If the Dockerfile does not exist in the root directory, the system automatically generates the Dockerfile based on the selected operating environment.
*Organization	An organization is used to manage images generated during component build.

Parameter	Description
*Build	Select the type of the environment used to build an image. The build environment must be a Kubernetes environment, and can access the Internet. You are advised to select **Use current environment**. If the CCE cluster in the current environment cannot access the Internet and you have planned an independent build environment, you can select **Use independent environment**. • Use independent environment: Use an independent build environment to build an image. The CCE clusters in the independent build environment and in the current component deployment environment must have the same CPU architecture. Otherwise, the component deployment fails. • Use current environment: Use the deployment environment to which the component belongs to build an image. In the current environment, masters and nodes in the CCE cluster must have the same CPU architecture. Otherwise, the component build fails.
*Environment	• **Use independent environment**: Select an independent build environment different from that to which the component belongs. • **Use current environment**: Select the deployment environment to which the component belongs.
*Namespace	Select the namespace of the CCE cluster in the environment where the build is executed, which is used to isolate build data. For details, see **Managing Namespaces**.
Node Label	You can use a node label to deliver the build job to a fixed node bound with an EIP. For details about how to add a label, see **Managing Node Labels**.
YAML Mode	• Disabled: The GUI configurations are used to deploy components. • Enabled: The YAML configurations are used to deploy components.

Figure 5-2 Configuring build parameters



Step 6 Click **Next**.

- If the component is deployed based on CCE and **YAML Mode** is enabled in **Step 5**, perform **Step 7** to **Step 9**.
- If the component is deployed based on VM, perform **Step 10** to **Step 14**.
- If the component is deployed based on CCE and **YAML Mode** is disabled in **Step 5**, perform **Step 10** to **Step 14**.

Step 7 (Optional) In the **Access Mode** area, click  to enable **Public Network Access**.

This operation is supported when the component is deployed based on CCE.

NOTE

- After public network access is enabled for a component, you can use a public network domain name to access the component through an ELB bound with an EIP to use services provided by the component.
- If a component is upgraded and maintained in ELB dark launch mode after being created and deployed, you need to enable public network access for the component. For details about ELB dark launch, see **Dark Launch (Canary)**.
- After a component with public network access enabled is created and deployed, you can change the configured component access domain name by referring to **Changing the Component Access Domain Name**.
- By default, public network access is disabled for a component. After a component is created and deployed, you can also configure the component access mode. For details, see **Configuring the Component Access Mode**.

1. Set **Public Network Load Balancer**.
   - Select an Elastic Load Balance (ELB) resource that has been bound to an elastic IP address (EIP) in the selected environment.
   - If no ELB resource exists, click **Add One**. On the **Edit Environment** page that is displayed, click **Add Optional Resource** to add created ELB resources to the environment.
   - For details about how to create an ELB resource, see **Creating a Shared Load Balancer**.

NOTE

- An ELB needs to be bound to an EIP, and must be in the same VPC and subnet as the compute resources managed in the current component deployment environment.
- Components must be bound to different ELBs in different deployment environments to avoid route errors.

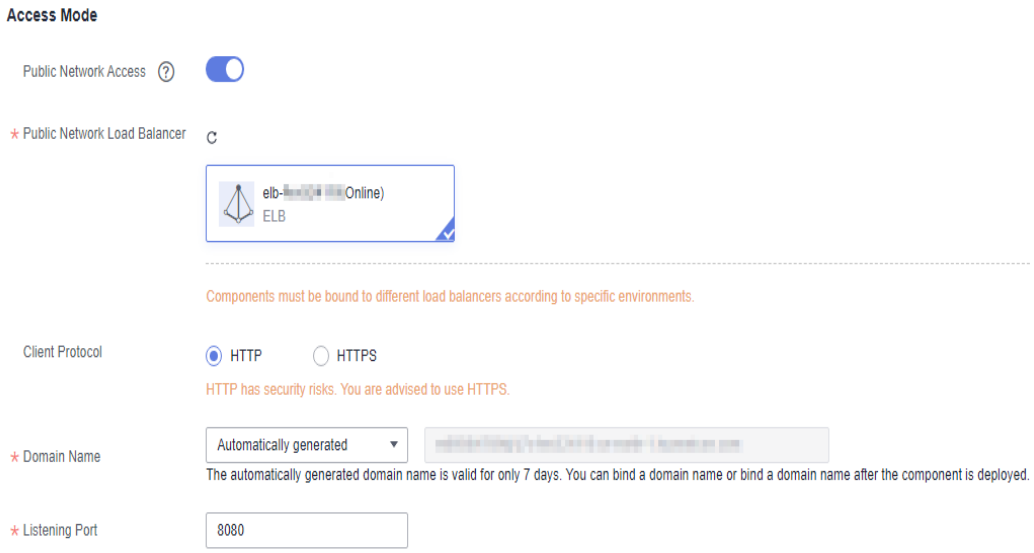
2. (Optional) Set **Client Protocol**.
   - **HTTP** has security risks. You are advised to select **HTTPS**.
   - If **HTTPS** is selected, click **Use existing** to select an existing certificate.

     If no certificate exists, click **Create new** to create a server certificate. For details, see **Creating a Certificate**.
3. Set **Domain Name**.
   - If **Automatically generated** is selected, the automatically generated domain name is valid only for seven days.

     Domain names cannot be automatically generated in LA-Sao Paulo1 and LA-Mexico City2.
   - If **Bound** is selected, enter a domain name.
4. Set **Listening Port**.

   Set the listening port of the application process.

Figure 5-3 Configuring public access



Step 8 Import or edit the YAML configuration file of the component.

This operation is supported when the component is deployed based on CCE and **YAML Mode** is enabled in **Step 5**. For details about the parameters in the YAML configuration file, see **Deployment**.

- Click **Import YAML File** to import the edited YAML configuration file.
- Edit the configuration parameters as required.

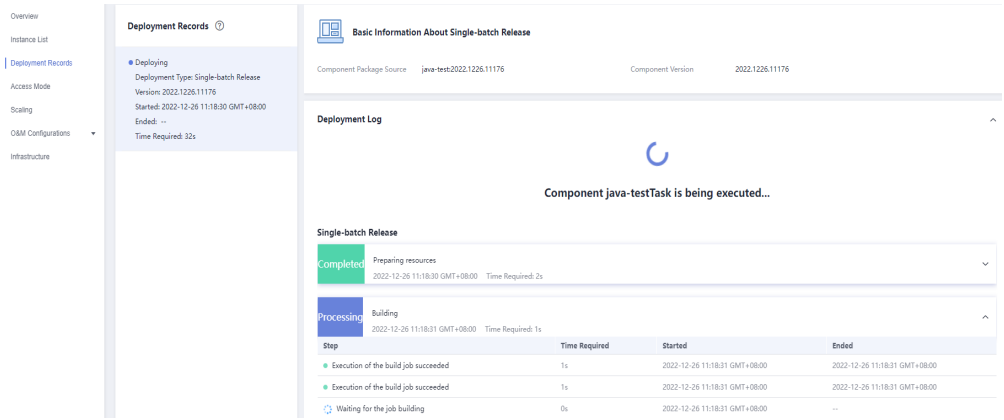
You can change **name** in the YAML configuration by referring to the description of **Workload Name** in **Step 3**. The workload name cannot be changed after the component is created and deployed.

```
1 apiVersion: apps/v1
2 kind: Deployment
3 metadata:
4   name: c-aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa-1uw8g0
5   namespace: default
```

Step 9 Click **Create and Deploy**.

On the **Deployment Records** page, view the deployment logs and wait until the component deployment is complete.

Figure 5-4 Viewing component deployment logs

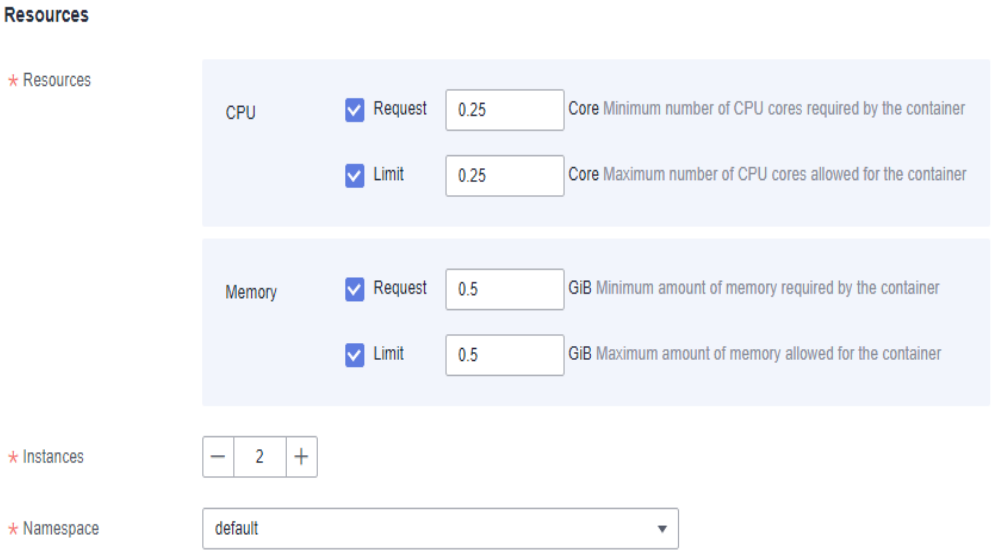


Step 10 In the **Resources** area, set the resources required by the component.

- If the component is deployed based on CCE, set the parameters by referring to the following table. Parameters marked with an asterisk (*) are mandatory.

Parameter	Description
*Resources	Components cannot be scheduled to nodes whose residual resources are fewer than the requested amount. For details about how to configure the request and limit parameters, see **Resource Management**. You can customize **CPU** and **Memory** to set their quota, and set the maximum/minimum number of CPU cores and memory size (GiB) that can be used by components. To make modifications, select the item to be changed and enter a new value. Unselected parameters indicate no restriction.
*Instances	Set the number of component instances running in the environment. The value range is 1–200.
*Namespace	Select the namespace of the container where the component instance is located.

Figure 5-5 Deploying resources of a Kubernetes component



- For components deployed based on VM, if **Resource Type** is set to **ECS**, select the ECS that has been managed in the component deployment environment; if **Resource Type** is set to **AS**, select the AS group to be used from the **Resources** drop-down list, and then select the ECS in the AS group to deploy the component.

  NOTE

  - The selected ECS must have the VM agent installed. For details, see **Installing a VM Agent**.
  - AS groups are not supported in LA-Sao Paulo1 and LA-Mexico City2.

Step 11 (Optional) In the **Access Mode** area, enable **Public Network Access**.

NOTE

- After public network access is enabled for a component, you can use a public network domain name to access the component through an ELB bound with an EIP to use services provided by the component.
- After a component with public network access enabled is created and deployed, you can change the configured component access domain name by referring to **Changing the Component Access Domain Name**.
- By default, public network access is disabled for a component. After a component is created and deployed, you can also configure the component access mode. For details, see **Configuring the Component Access Mode**.

Click  to enable public access and set the following parameters:

1. Set **Public Network Load Balancer**.
   - Select an Elastic Load Balance (ELB) resource that has been bound to an elastic IP address (EIP) in the selected environment.
   - If no ELB resource exists, click **Add One**. On the **Edit Environment** page that is displayed, click **Add Optional Resource** to add created ELB resources to the environment.
   - For details about how to create an ELB resource, see **Creating a Shared Load Balancer**.

NOTE

- An ELB needs to be bound to an EIP, and must be in the same VPC and subnet as the compute resources managed in the current component deployment environment.
- Components must be bound to different ELBs in different deployment environments to avoid route errors.

2. (Optional) Set **Client Protocol**.
   - **HTTP** has security risks. You are advised to select **HTTPS**.
   - If **HTTPS** is selected, click **Use existing** to select an existing certificate.

     If no certificate exists, click **Create new** to create a server certificate. For details, see **Creating a Certificate**.
3. Set **Domain Name**.
   - If **Automatically generated** is selected, the automatically generated domain name is valid only for seven days.

     Domain names cannot be automatically generated in LA-Sao Paulo1 and LA-Mexico City2.
   - If **Bound** is selected, enter a domain name.

Step 12 (Optional) In the **Local Time** area, set the time zone of the container.

This parameter is available when the component is deployed based on CCE.

By default, the time zone is the same as that of the region where the container node is located.

Step 13 (Optional) Set **Advanced Settings**.

- If the component is deployed based on CCE, refer to the following table.

Operation Type	Operation	Description
Microservice Engine	Binding a Microservice Engine	Bind the microservice engine and connect the component to it. **NOTE** After a component developed based on ServiceComb 2.7.8 or later or Spring Cloud Huawei 1.10.4-2021.0.x or later is connected to a microservice engine, the following attributes are injected into the **properties** parameter of the **MicroServiceInstance** parameter when a microservice instance is created in the microservice engine: 1. **CAS_APPLICATION_ID**: ID of the application to which the component belongs. 2. **CAS_COMPONENT_NAME**: component name. 3. **CAS_ENVIRONMENT_ID**: ID of the component deployment environment. 4. **CAS_INSTANCE_ID**: component instance ID. 5. **CAS_INSTANCE_VERSION**: component instance version. For details about the **MicroServiceInstance** parameter, see **MicroServiceInstance**. 1. Choose **Advanced Settings** > **Microservice Engine**. 2. Click **Bind Microservice Engine**. 3. Select a microservice engine instance that has been bound in the environment. 4. Click **OK**. **NOTE** Move the cursor to a bound microservice engine and perform the following operations: ▪ Bind the microservice engine again: Click  , select the target microservice engine again, and click **OK**. ▪ Delete a bound microservice engine: Click  . 5. (Optional) If an exclusive microservice engine is bound, specify **Addons**. ▪ Mesher: Enter the listening port number of the application process to enable multi-language access to service mesh. You can use Mesher to connect components that are not developed in the microservice framework to the microservice engine.

Operation Type	Operation	Description
		NOTE ■ For non-microservice components developed using Java, Tomcat, or Docker, you can enable Mesher and use Mesher to connect the components to CSE for microservice registry and discovery. ■ For components developed using Python, PHP, or Node.js, forcibly enable Mesher and connect the components to CSE for microservice registry and discovery.
Distributed Cache Service	Binding a Distributed Cache	In a distributed system, the distributed cache service provides scalable and reliable user session management. Choose **Advanced Settings** > **Distributed Cache Service** and bind a DCS instance. For details, see **Configuring Distributed Cache Service**.
Cloud Database	Binding a Cloud Database	The cloud database is required for persistent storage of application data. Expand **Advanced Settings** > **Cloud Database** and bind cloud database. For details, see **Configuring Relational Databases**.
Component Configurations	Configuring Environment Variables	Environment variables are set in the container running environment and can be modified after component deployment, ensuring the flexibility of applications. Environment variables set for a component are local environment variables and take effect only for this component. Choose **Advanced Settings** > **Component Configuration** and set environment variables. For details, see **Configuring Environment Variables of a Component**.
Deployment Configurations	Setting Startup Commands	A startup command is used to start a container. Choose **Advanced Settings** > **Deployment Configuration** and set the startup command. For details, see **Configuring the Lifecycle of a Component**.
	Configuring Data Storage	Container storage is a component that provides storage for applications. Multiple types of storage are supported. A component can use any amount of storage. Choose **Advanced Settings** > **Deployment Configuration** and configure data storage. For details, see **Configuring Data Storage**.

Operation Type	Operation	Description
	Configuring the Lifecycle	For container-deployed components, ServiceStage provides callback functions for the lifecycle management of applications. For example, if you want an application component to perform a certain operation before stopping, you can register a hook function. Choose **Advanced Settings** > **Deployment Configuration** and configure the lifecycle. For details, see **Configuring the Lifecycle of a Component**.
	Configuring the Scheduling Policy	For container-based deployment components, ServiceStage splits the components into minimum deployment instances based on the deployment features of the components. The application scheduler monitors application instance information in real time. When detecting that a new pod needs to be scheduled, the application scheduler calculates all remaining resources (compute, storage, and network resources) in the cluster to obtain the most appropriate scheduling target node. Choose **Advanced Settings** > **Deployment Configuration** and configure the scheduling policy. For details, see **Configuring a Scheduling Policy of a Component Instance**.
O&M and Monitoring	Configuring Log Collection	For container-based deployment components, ServiceStage supports setting of application log policies. You can view related logs on the AOM console. You can configure a log policy during or after component deployment. If no configuration is performed, the system collects standard application output logs by default. Choose **Advanced Settings** > **O&M Monitoring** and configure log collection. For details, see **Configuring a Log Policy of an Application**.
	Configuring Health Check	Health check periodically checks application health status during running of container-based deployment components. Choose **Advanced Settings** > **O&M Monitoring** and configure health check. For details, see **Configuring Health Check**.

Operation Type	Operation	Description
	Configuring Performance Management	Performance management helps you quickly locate problems and identify performance bottlenecks to improve your experience. ServiceStage allows you to configure application performance management during or after component deployment. APM can be configured for components whose technology stack type is Java, Tomcat, or Docker. Choose **Advanced Settings** > **O&M Monitoring** and configure performance management. For details, see **Configuring Application Performance Management**.
	Configuring Custom Monitoring	ServiceStage allows you to obtain monitoring data based on custom metrics. You can set custom metric monitoring during or after component deployment. This section applies to components deployed using CCE. Choose **Advanced Settings** > **O&M Monitoring** and configure custom monitoring. For details, see **Configuring Custom Monitoring of a Component**.

Figure 5-6 Setting advanced settings of a Kubernetes component

- If the component is deployed based on VM, refer to the following table.

Operation	Description
Configuring Environment Variables	Environment variables are set in the container running environment and can be modified after component deployment, ensuring the flexibility of applications. Environment variables set for a component are local environment variables and take effect only for this component. For details about how to set environment variables, see **Configuring Environment Variables of a Component**.

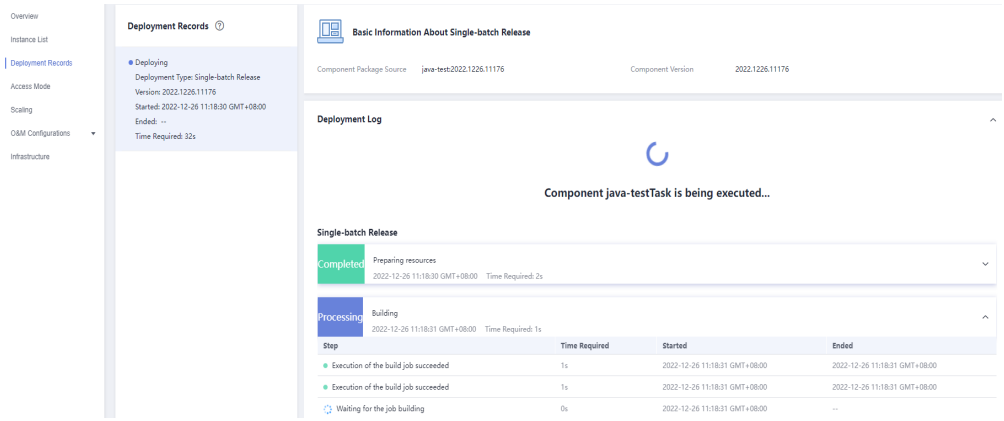
Operation	Description
Binding a Microservice Engine	Components whose technology stack type is Java or Tomcat can be bound to microservice engines for microservice governance. 1. Click **Bind Microservice Engine**. 2. Select a microservice engine instance that has been bound in the environment and click **OK**.

Figure 5-7 Setting advanced settings of a VM component

Step 14 Click **Create and Deploy**.

On the **Deployment Records** page, view the deployment logs and wait until the component deployment is complete.

Figure 5-8 Viewing component deployment logs



----End

5.3 Viewing Component Details

After the component is created and deployed, you can view the component details on its **Overview** page.

Prerequisites

- To view the YAML configurations of a CCE-deployed component, enable **View YAML Mode**. The parameters in the YAML configuration file are described in **Deployment**.

- Click **Synchronize Workload** to synchronize latest configurations after CCE-based component workload changes (such as workloads upgraded using CCE).

Procedure

Step 1 Log in to ServiceStage.

Step 2 Use either of the following methods to go to the component **Overview** page.

- On the **Application Management** page, click the application to which the component belongs, and click the target component in **Component List**. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu.
- On the **Component Management** page, click the target component. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu.

----End

5.4 Managing Component Labels

Labels are key-value pairs and can be attached to workloads. Workload labels are often used for affinity and anti-affinity scheduling. You can add labels to multiple workloads or a specified workload.

You can manage the labels of stateless workloads, stateful workloads, and Daemon sets based on service requirements. This section uses Deployments as an example to describe how to manage labels.

In the following figure, three labels (release, env, and role) are defined for workload APP 1, APP 2, and APP 3. The values of these labels vary with workload.

- APP 1: [release:alpha;env:development;role:frontend]
- APP 2: [release:beta;env:testing;role:frontend]
- APP 3: [release:alpha;env:production;role:backend]

If you set **key** to **role** and **value** to **frontend** when using workload scheduling or another function, APP 1 and APP 2 will be selected.

Figure 5-9 Label example

NOTE

Labels cannot be added to components that are abnormal or deployed based on VM.

Adding Component Labels

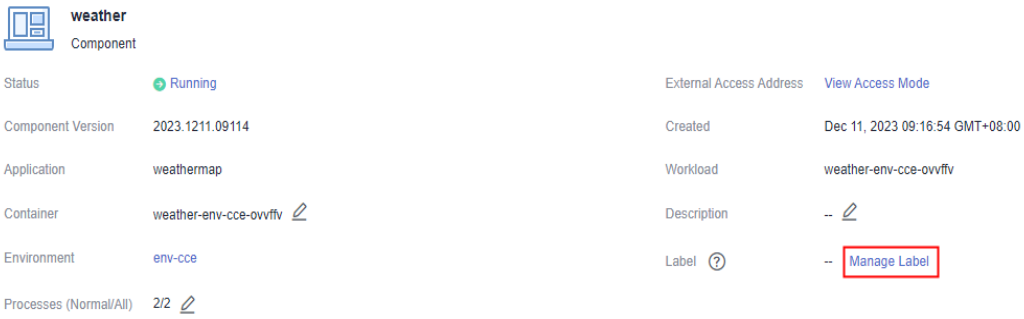
Step 1 Log in to ServiceStage.

Step 2 Use either of the following methods to go to the component **Overview** page.

- On the **Application Management** page, click the application to which the component belongs, and click the target component in **Component List**. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu.
- On the **Component Management** page, click the target component. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu.

Step 3 Click **Manage Label**.

Figure 5-10 Managing labels



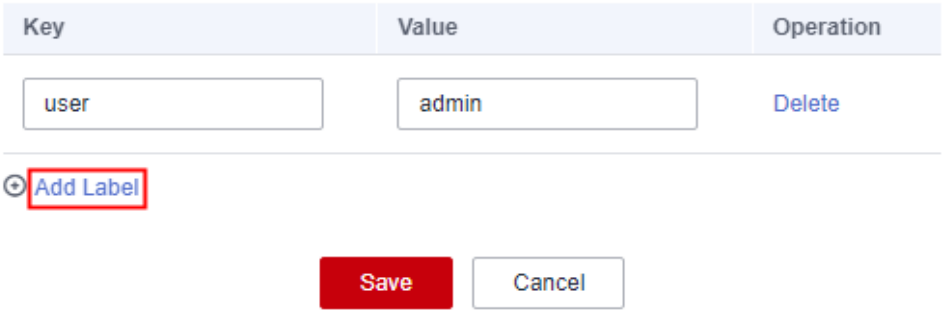
Step 4 Click **Add Label**.

1. Enter the **key** and **value**.

   The key must be unique.

2. Click **Save**.

Figure 5-11 Adding a label



----End

Deleting a Component Label

Step 1 Log in to ServiceStage.

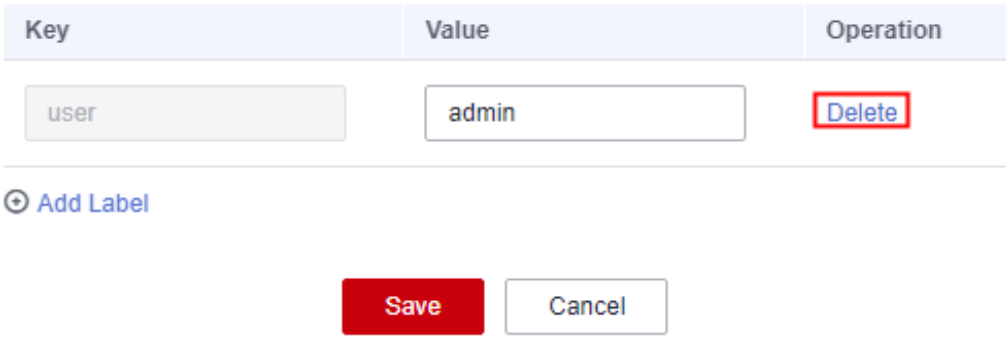
Step 2 Use either of the following methods to go to the component **Overview** page.

- On the **Application Management** page, click the application to which the component belongs, and click the target component in **Component List**. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu.
- On the **Component Management** page, click the target component. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu.

Step 3 Click **Manage Label**.

Step 4 Select the label to be deleted and click **Delete** in the **Operation** column.

Figure 5-12 Deleting a label



Step 5 Click **Save**.

----End

5.5 Changing the Container Name of a CCE-deployed Component

For components created and deployed based on CCE, you can change the container name of a **Running** or **Not ready** component.

Procedure

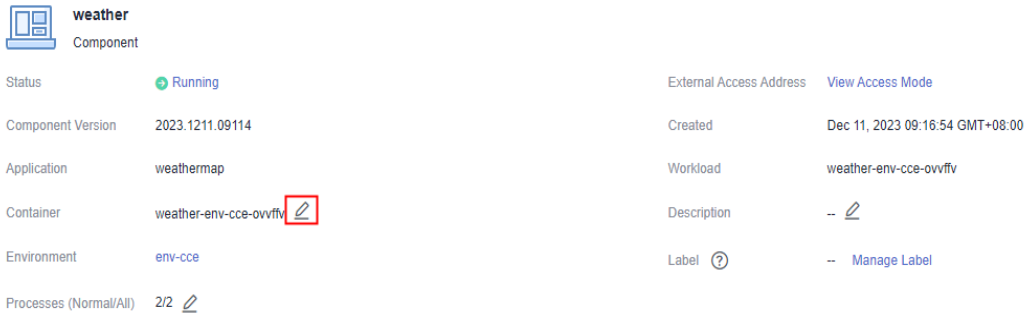
Step 1 Log in to ServiceStage.

Step 2 Use either of the following methods to go to the component **Overview** page.

- On the **Application Management** page, click the application to which the component belongs, and click the target component in **Component List**. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu.
- On the **Component Management** page, click the target component. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu.

Step 3 Change the container name.

1. Click next to the container name.



2. Enter a new container name.
3. Click .

----End

5.6 Changing the Component Description

After a component is created and deployed, you can change the description of a **Running** or **Not ready** component.

Procedure

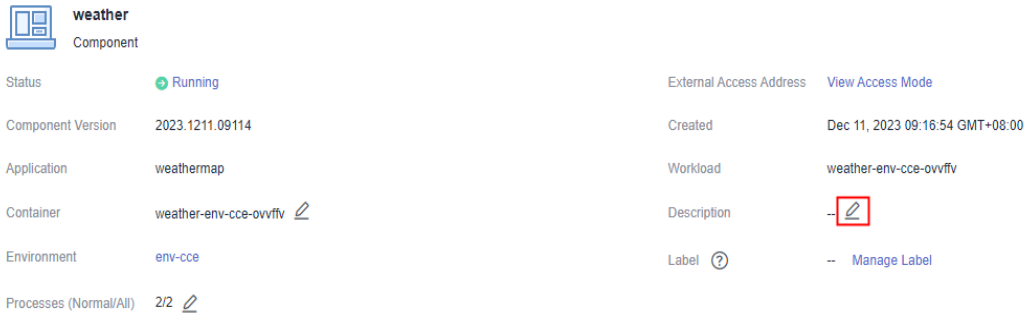
Step 1 Log in to ServiceStage.

Step 2 Use either of the following methods to go to the component **Overview** page.

- On the **Application Management** page, click the application to which the component belongs, and click the target component in **Component List**. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu.
- On the **Component Management** page, click the target component. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu.

Step 3 Change the component description.

1. Click next to the description.



2. Enter the component description.
3. Click .

----End

5.7 Managing Component Instances

In a VM environment, a component instance is a running process of a component on a VM. In a CCE environment, a component instance is a pod, which is the minimum basic unit for CCE to deploy applications or services.

The number of VM-deployed component instances is the number of VMs selected during component deployment. The number of CCE-deployed component instances is the number of instances selected during component deployment, that is, the number of pods.

After a component is created and deployed, you can manage component instances on the component **Instance List** page.

Procedure

Step 1 Log in to ServiceStage.

Step 2 Use either of the following methods to go to the **Instance List** page.

- On the **Application Management** page, click the application to which the component belongs, and click the target component in **Component List**. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu, and choose **Instance List** in the left navigation pane.
- On the **Component Management** page, click the target component. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu, and choose **Instance List** in the left navigation pane.

Step 3 On the **Instance List** page, you can perform the following operations.

Operation	Description
Restart a single instance	If an instance of a component deployed in the Kubernetes environment is abnormal, you can delete the instance to restart it. 1. Select the instance to be deleted and click **Delete** in the **Operation** column. 2. In the displayed dialog box, click **OK**.
View instance running monitoring information	By viewing the instance running monitoring information, you can learn about the CPU and memory usage of a single running instance. 1. In the instance list, click ⌄ next to the target instance. 2. Click the **Monitor** tab to view the running monitoring information about the instance.

Operation	Description
View instance running events	ServiceStage allows you to view details about events that occur during the running of a specified instance. 1. In the instance list, click next to the target instance. 2. Click the **Event** tab to view events that occur during instance running.
View running instance containers	For components deployed in the Kubernetes environment, ServiceStage allows you to view information about the container where a specified instance runs, including the container name, running status, and mounted image. 1. In the instance list, click next to the target instance. 2. Click the **Container** tab to view the information about the container where the instance runs.

----End

5.8 Upgrading a Single Component

5.8.1 Single-batch Release

After a component is created and deployed, you can upgrade a **Running** or **Not ready** component in single-batch release mode.

In single-batch release mode, all instances are upgraded at a time. During the upgrade, component services will be interrupted. This is applicable to the test upgrade scenario or the upgrade scenario where services are to be stopped. The upgrade takes a short time.

NOTE

Only components deployed in the Kubernetes environment can be upgraded in single-batch release mode.

For details about how to upgrade multiple component versions of the same application in batches, see **Upgrading Components in Batches**.

Prerequisites

You have created and deployed a component. For details, see **Creating and Deploying a Component**.

Procedure

Step 1 Log in to ServiceStage.

Step 2 Use either of the following methods to go to the component **Overview** page.

- On the **Application Management** page, click the application to which the component belongs, and click the target component in **Component List**.

Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu.

- On the **Component Management** page, click the target component. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu.

Step 3 Click **Upgrade** in the upper right corner of the page.

Step 4 Select **Single-batch Release** for **Upgrade Type**.

Step 5 Click **Next** and set the component version configuration information by referring to the following table. Parameters marked with an asterisk (*) are mandatory.

Parameter	Description
Stack	The value is fixed to the technology stack selected during component creation and deployment.
*YAML Mode	Uses YAML configurations to upgrade components. • Disabled: The GUI configurations are used to upgrade components. • Enabled: The YAML configurations are used to upgrade components. The latest load information of the component is automatically synchronized from CCE where the target component is deployed for component upgrade after modification. Alternatively, click **Import YAML File** to import the edited YAML configuration file of the target component. **NOTE** If YAML configurations are used to upgrade components, the parameters in the YAML configuration file are described in **Deployment**.
*Software Package/ Image	The value is fixed to the component source selected during component creation and deployment. • **YAML Mode** disabled: If you select **Source code repository**, create authorization by referring to **Authorizing a Repository** and set the code source. If you select a software package or image package, the option is fixed to the software package type (JAR, WAR, or ZIP) or image package type selected during component creation and deployment. It is determined by the selected technology stack type. For details, see **Table 5-1**. • **YAML Mode** enabled: If you select **Source code repository**, create authorization by referring to **Authorizing a Repository** and set the code source. If you select a software package, the option is fixed to the software package type (JAR, WAR, or ZIP) selected during component creation and deployment. It is determined by the selected technology stack type. For details, see **Table 5-1**.

Parameter	Description
*Upload Method	• **YAML Mode** disabled: If the component source is software package or image package, select an uploaded software package or image package. For details about the upload method, see **Component Source**. • **YAML Mode** enabled: If the component source is software package, select an uploaded software package. For details about the upload method, see **Component Source**.
*Command	This parameter is mandatory when **YAML Mode** is disabled, the component source is **Source code repository**, the component is deployed in the Kubernetes environment, and the selected technology stack type is Java, Tomcat, Node.js, Python, or PHP. • **Default command or script**: preferentially executes **build.sh** in the **root** directory. If **build.sh** does not exist, the code will be compiled using the common method of the selected language. Example: **mvn clean package** for Java. • **Custom command**: Commands are customized using the selected language. Alternatively, the default command or script is used after **build.sh** is modified. **NOTICE** – If **Custom command** is selected, exercise caution when inputting sensitive information in the **echo**, **cat**, or **debug** command, or encrypt sensitive information to avoid information leakage. – To run the compilation command in the project subdirectory, you need to go to the project subdirectory and then run other script commands. For example: **cd ./weather/** **mvn clean package**
*Dockerfile Address	This parameter is mandatory when **YAML Mode** is disabled, the component source is **Source code repository**, the component is deployed in the Kubernetes environment, and the selected technology stack type is Java, Tomcat, Node.js, Python, or PHP. **Dockerfile Address** is the directory where the Dockerfile is located relative to the root directory (./) of the project. The Dockerfile is used to build an image. If **Dockerfile Address** is not specified, the system searches for the Dockerfile in the root directory of the project by default. If the Dockerfile does not exist in the root directory, the system automatically generates the Dockerfile based on the selected operating environment.

Parameter	Description
*Component Version	Component version number, which can be automatically generated or customized. • Automatically-generated: Click **Generate**. By default, the version number is the timestamp when you click **Generate**. The format is yyyy.mmdd.hhmms, where **s** is the ones place of the second in the timestamp. For example, if the timestamp is 2022.0803.104321, the version number is 2022.0803.10431. • Customized: Enter a value in the format of A.B.C or A.B.C.D. A, B, C, and D are natural numbers. For example, 1.0.0 or 1.0.0.0. **NOTICE** The customized version number must be unique and cannot be the same as any historical version number of the component.
Resources	This parameter is available when **YAML Mode** is disabled. Components cannot be scheduled to nodes whose residual resources are fewer than the requested amount. For details about how to configure the request and limit parameters, see **Managing Resources for Containers**. You can customize **CPU** and **Memory** to set their quota, and change the maximum/minimum number of CPU cores and memory size (GiB) that can be used by components. To make modifications, select the item to be changed and enter a new value. Unselected parameters indicate no restriction.
JVM Parameters	This parameter is available when **YAML Mode** is disabled and the technology stack type is Java or Tomcat. It configures the memory size during Java code running. Enter the JVM parameter, for example, **-Xms256m -Xmx1024m**. Multiple parameters are separated by spaces. If the parameter is left blank, the value is empty.
Tomcat	This parameter is available when **YAML Mode** is disabled and the technology stack type is Tomcat. It configures parameters such as the request path and port number of Tomcat. 1. Select **Tomcat**. The **Tomcat** dialog box is displayed. 2. Click **Use Sample Code** and edit the template file based on service requirements. **NOTE** In Tomcat configuration, the default **server.xml** configuration is used. The context path is **/**, and no application path is specified. If you need to customize an application path, customize the Tomcat context path by referring to **How Do I Customize a Tomcat Context Path?** 3. Click **OK**.

Parameter	Description
Advanced Settings	This parameter is available when **YAML Mode** is disabled. Set **Microservice Engine**, **Component Configuration**, **Deployment Configuration**, and **O&M Monitoring** by referring to **Step 13**.

Step 6 Click **Upgrade**.

Wait until the component status changes from **Upgrading/Rolling back** to **Running**, indicating that the component version configuration is successfully upgraded.

----End

Follow-Up Operations

Operation	Description
Redeploying a Component	You can select a historical version configuration from the deployment record list and use the version configuration as a template to redeploy components. For details, see **Redeploying a Component**.

5.8.2 Rolling Release

After a component is created and deployed, you can upgrade a **Running** or **Not ready** component in rolling release mode.

In rolling release mode, only one or more instances are upgraded at a time and then added to the production environment. This process repeats until all old instances are upgraded. Services will not be interrupted during the upgrade.

For details about how to upgrade multiple component versions of the same application in batches, see **Upgrading Components in Batches**.

Prerequisites

You have created and deployed a component. For details, see **Creating and Deploying a Component**.

Procedure

Step 1 Log in to ServiceStage.

Step 2 Use either of the following methods to go to the component **Overview** page.

- On the **Application Management** page, click the application to which the component belongs, and click the target component in **Component List**. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu.

- On the **Component Management** page, click the target component. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu.

Step 3 Click **Upgrade** in the upper right corner of the page.

Step 4 Select **Rolling Release** for **Upgrade Type**.

Step 5 Click **Next** and set the component version configuration information by referring to the following table. Parameters marked with an asterisk (*) are mandatory.

Parameter	Description
Stack	The value is fixed to the technology stack selected during component creation and deployment.
*YAML Mode	Uses YAML configurations to upgrade components if the component is deployed in the Kubernetes environment. • Disabled: The GUI configurations are used to upgrade components. • Enabled: The YAML configurations are used to upgrade components. The latest load information of the component is automatically synchronized from CCE where the target component is deployed for component upgrade after modification. Alternatively, click **Import YAML File** to import the edited YAML configuration file of the target component. **NOTE** If YAML configurations are used to upgrade components, the parameters in the YAML configuration file are described in **Deployment**.
*Software Package/ Image	The value is fixed to the component source selected during component creation and deployment. • **YAML Mode** disabled: If you select **Source code repository**, create authorization by referring to **Authorizing a Repository** and set the code source. If you select a software package or image package, the option is fixed to the software package type (JAR, WAR, or ZIP) or image package type selected during component creation and deployment. It is determined by the selected technology stack type. For details, see **Table 5-1**. • **YAML Mode** enabled: If you select **Source code repository**, create authorization by referring to **Authorizing a Repository** and set the code source. If you select a software package, the option is fixed to the software package type (JAR, WAR, or ZIP) selected during component creation and deployment. It is determined by the selected technology stack type. For details, see **Table 5-1**.
*Upload Method	• **YAML Mode** disabled: If the component source is software package or image package, select an uploaded software package or image package. For details about the upload method, see **Component Source**. • **YAML Mode** enabled: If the component source is software package, select an uploaded software package. For details about the upload method, see **Component Source**.

Parameter	Description
*Command	This parameter is mandatory when **YAML Mode** is disabled, the component source is **Source code repository**, the component is deployed in the Kubernetes environment, and the selected technology stack type is Java, Tomcat, Node.js, Python, or PHP. • **Default command or script**: preferentially executes **build.sh** in the **root** directory. If **build.sh** does not exist, the code will be compiled using the common method of the selected language. Example: **mvn clean package** for Java. • **Custom command**: Commands are customized using the selected language. Alternatively, the default command or script is used after **build.sh** is modified. **NOTICE** – If **Custom command** is selected, exercise caution when inputting sensitive information in the **echo**, **cat**, or **debug** command, or encrypt sensitive information to avoid information leakage. – To run the compilation command in the project subdirectory, you need to go to the project subdirectory and then run other script commands. For example: **cd ./weather**/ **mvn clean package**
*Dockerfile Address	This parameter is mandatory when **YAML Mode** is disabled, the component source is **Source code repository**, the component is deployed in the Kubernetes environment, and the selected technology stack type is Java, Tomcat, Node.js, Python, or PHP. **Dockerfile Address** is the directory where the Dockerfile is located relative to the root directory (./) of the project. The Dockerfile is used to build an image. If **Dockerfile Address** is not specified, the system searches for the Dockerfile in the root directory of the project by default. If the Dockerfile does not exist in the root directory, the system automatically generates the Dockerfile based on the selected operating environment.
*Component Version	Component version number, which can be automatically generated or customized. • Automatically-generated: Click **Generate**. By default, the version number is the time when you click **Generate**. The format is yyyy.mmdd.hhmms, where **s** is the ones place of the second in the timestamp. For example, if the timestamp is 2022.0803.104321, the version number is 2022.0803.10431. • Customized: Enter a value in the format of A.B.C or A.B.C.D. A, B, C, and D are natural numbers. For example, 1.0.0 or 1.0.0.0. **NOTICE** The customized version number must be unique and cannot be the same as any historical version number of the component.

Parameter	Description
Resources	This parameter is available when **YAML Mode** is disabled and the component is deployed based on CCE. Components cannot be scheduled to nodes whose residual resources are fewer than the requested amount. For details about how to configure the request and limit parameters, see **Managing Resources for Containers**. You can customize **CPU** and **Memory** to set their quota, and change the maximum/minimum number of CPU cores and memory size (GiB) that can be used by components. To make modifications, select the item to be changed and enter a new value. Unselected parameters indicate no restriction.
Environment variables	This parameter is available when the component is deployed based on VM. For details, see **Configuring Environment Variables of a Component**.
JVM Parameters	This parameter is available when **YAML Mode** is disabled or the component is deployed based on VM, and the technology stack type is Java or Tomcat. It configures the memory size during Java code running. Enter the JVM parameter, for example, **-Xms256m -Xmx1024m**. Multiple parameters are separated by spaces. If the parameter is left blank, the value is empty.
Tomcat	This parameter is available when **YAML Mode** is disabled or the component is deployed based on VM, and the technology stack type is Tomcat. It configures parameters such as the request path and port number of Tomcat. 1. Select **Tomcat**. The **Tomcat** dialog box is displayed. 2. Click **Use Sample Code** and edit the template file based on service requirements. **NOTE** In Tomcat configuration, the default **server.xml** configuration is used. The context path is **/**, and no application path is specified. If you need to customize an application path, customize the Tomcat context path by referring to **How Do I Customize a Tomcat Context Path?** 3. Click **OK**.
Advanced Settings	This parameter is available when **YAML Mode** is disabled and the component is deployed based on CCE. Set **Microservice Engine**, **Component Configuration**, **Deployment Configuration**, and **O&M Monitoring** by referring to **Step 13**.

Parameter	Description
*Deployment Batches	This parameter is available when the component is deployed based on CCE. Number of batches in which component instances are upgraded. The value range is [1, Total number of instances]. Total number of instances refers to the number of running instances of the component. For example, if there are 4 component instances and **Deployment Batches** is set to **2**, these component instances are upgraded in two batches, and each batch involves two component instances.

Step 6 Click **Upgrade**.

Wait until the component status changes from **Upgrading/Rolling back** to **Running**, indicating that the component version configuration is successfully upgraded.

----End

Follow-Up Operations

Operation	Description
Redeploying a Component	You can select a historical version configuration from the deployment record list and use the version configuration as a template to redeploy components. For details, see **Redeploying a Component**.

5.8.3 Dark Launch (Canary)

After a component is created and deployed, you can upgrade a **Running** or **Not ready** component in dark launch (canary) mode.

In dark launch (canary) mode, a certain proportion of instances are upgraded, and traffic is directed to the new version to verify whether functions of the new version are normal. Then, the remaining instances will be upgraded in rolling mode. Dark launch ensures stability of the entire system. During initial dark launch, problems can be detected and fixed.

For details about dark launch (canary) types and details, see **Table 5-2**.

Table 5-2 Dark launch (canary) types and description

Type	Description
Microservice Dark Launch	Applies to ServiceComb and Spring Cloud applications. Dark launch tasks function on microservices. Multiple microservices can work together to roll out new features. 1. The Java, Tomcat, or Docker technology stack must be selected for the component. 2. The component must be bound to a microservice engine with security authentication disabled and multi-language access to service mesh disabled. 3. ServiceComb 2.7.8 or later is required. Spring Cloud Huawei 1.10.4-2021.0.x or later is required.
ELB Dark Launch	Applies to ELB traffic-based components. Dark launch tasks function on ELB. When creating and deploying a component, enable public network access for the component. For details, see **Creating and Deploying a Component**.

NOTE

Upgrade in dark launch (canary) mode is supported only when the deployment environment is Kubernetes and there are two or more component instances.

For details about how to upgrade multiple component versions of the same application in batches, see **Upgrading Components in Batches**.

Prerequisites

You have created and deployed a component. For details, see **Creating and Deploying a Component**.

Procedure

Step 1 Log in to ServiceStage.

Step 2 Use either of the following methods to go to the component **Overview** page.

- On the **Application Management** page, click the application to which the component belongs, and click the target component in **Component List**. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu.
- On the **Component Management** page, click the target component. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu.

Step 3 Click **Upgrade** in the upper right corner of the page.

Step 4 Select **Dark Launch (Canary)** for **Upgrade Type**.

Step 5 Click **Next** and set the component version configuration information by referring to the following table. Parameters marked with an asterisk (*) are mandatory.

NOTICE

During the dark launch upgrade of a component microservice in this operation, do not use CSE to perform dark launch of the component microservice at the same time. Otherwise, this operation fails.

For details about how to perform dark launch of a component microservice through CSE, see **Dark Launch**.

Parameter	Description
Stack	The value is fixed to the technology stack selected during component creation and deployment.
*Software Package/ Image	The value is fixed to the component source selected during component creation and deployment. If you select **Source code repository**, create authorization by referring to **Authorizing a Repository** and set the code source. If you select a software package or image package, the option is fixed to the software package type (JAR, WAR, or ZIP) or image package type selected during component creation and deployment. It is determined by the selected technology stack type. For details, see **Table 5-1**.
*Upload Method	If the component source is software package or image package, select an uploaded software package or image package. For details about the upload method, see **Component Source**.
*Command	This parameter is mandatory when the component source is **Source code repository**, the component is deployed in the Kubernetes environment, and the selected technology stack type is Java, Tomcat, Node.js, Python, or PHP. • **Default command or script**: preferentially executes **build.sh** in the **root** directory. If **build.sh** does not exist, the code will be compiled using the common method of the selected language. Example: **mvn clean package** for Java. • **Custom command**: Commands are customized using the selected language. Alternatively, the default command or script is used after **build.sh** is modified. **NOTICE** – If **Custom command** is selected, exercise caution when inputting sensitive information in the **echo**, **cat**, or **debug** command, or encrypt sensitive information to avoid information leakage. – To run the compilation command in the project subdirectory, you need to go to the project subdirectory and then run other script commands. For example: **cd ./weather/** **mvn clean package**

Parameter	Description
*Dockerfile Address	This parameter is mandatory when the component source is **Source code repository**, the component is deployed in the Kubernetes environment, and the selected technology stack type is Java, Tomcat, Node.js, Python, or PHP. **Dockerfile Address** is the directory where the Dockerfile is located relative to the root directory (./) of the project. The Dockerfile is used to build an image. If **Dockerfile Address** is not specified, the system searches for the Dockerfile in the root directory of the project by default. If the Dockerfile does not exist in the root directory, the system automatically generates the Dockerfile based on the selected operating environment.
*Component Version	Component version number, which can be automatically generated or customized. • Automatically-generated: Click **Generate**. By default, the version number is the time when you click **Generate**. The format is yyyy.mmdd.hhmms, where **s** is the ones place of the second in the timestamp. For example, if the timestamp is 2022.0803.104321, the version number is 2022.0803.10431. • Customized: Enter a value in the format of A.B.C or A.B.C.D. A, B, C, and D are natural numbers. For example, 1.0.0 or 1.0.0.0. **NOTICE** The customized version number must be unique and cannot be the same as any historical version number of the component.
Resources	Components cannot be scheduled to nodes whose residual resources are fewer than the requested amount. For details about how to configure the request and limit parameters, see **Managing Resources for Containers**. You can customize **CPU** and **Memory** to set their quota, and change the maximum/minimum number of CPU cores and memory size (GiB) that can be used by components. To make modifications, select the item to be changed and enter a new value. Unselected parameters indicate no restriction.
JVM Parameters	This parameter is available when the technology stack type is Java or Tomcat. It configures the memory parameter size during Java code running. Enter the JVM parameter, for example, **-Xms256m -Xmx1024m**. Multiple parameters are separated by spaces. If the parameter is left blank, the value is empty.

Parameter	Description
Tomcat	This parameter is available when the technology stack type is Tomcat. It configures parameters such as the request path and port number of Tomcat. 1. Select **Tomcat**. The **Tomcat** dialog box is displayed. 2. Click **Use Sample Code** and edit the template file based on service requirements. **NOTE** In Tomcat configuration, the default **server.xml** configuration is used. The context path is **/**, and no application path is specified. If you need to customize an application path, customize the Tomcat context path by referring to **How Do I Customize a Tomcat Context Path?** 3. Click **OK**.
Advanced Settings	Set **Component Configuration**, **Deployment Configuration**, and **O&M Monitoring** by referring to **Step 13**.
Dark Launch Policy	• **Traffic Ratio**: percentage of traffic directed to the new version. • **Current Traffic Ratio**: percentage of traffic directed to the current version.
*First Batch of Dark Launch Instances	Number of instances for dark launch in the first batch. The value range is [1, Total number of instances – 1]. Total number of instances refers to the number of running instances of the component. For example, if there are 6 component instances and **First Batch of Dark Launch Instances** is set to **1**, 1 instance will be upgraded in the first batch.
Deployment Batch with Remaining Instances	Number of batches whose remaining instances will be upgraded. For example, if there are 6 component instances, **First Batch of Dark Launch Instances** is set to **1**, and **Deployment Batch with Remaining Instances** is set to **3**, there are 5 instances remaining to be deployed in 3 batches, and these 5 instances will be upgraded in the sequence 2:2:1

Step 6 Click **Upgrade**.

Wait until the component status changes from **Upgrading/Rolling back** to **Running**, indicating that the component version configuration is successfully upgraded.

----End

Follow-Up Operations

Operation	Description
View System Monitoring	If the component is upgraded through dark launch (canary), choose **System Monitoring** to monitor the CPU and memory usage of instances of the dark launch version and the current version after the first batch of dark launch is complete.
Upgrade Remaining Instances in Rolling Mode	To upgrade the remaining component instances to the new version after the first batch of dark launch is successful and the functions of the new version are normal, perform the following operations: 1. Select the deployment record of the **Dark Launch (Canary)** type. 2. Click **Upgrade Remaining Instances in Rolling Mode**. 3. In the displayed dialog box, click **OK**. The remaining instances are upgraded to the new version based on the upgrade policy set in **Step 5**.
Rolling Back a Component	The component version configuration can be rolled back in the following scenarios: • After the first batch of dark launch release is complete when the component version is upgraded through dark launch (canary). • After the version configuration of all component instances is upgraded to the new version. To roll back the component configuration to the source version, refer to **Rolling Back a Component**.
Redeploying a Component	You can select a historical version configuration from the deployment record list and use the version configuration as a template to redeploy components. For details, see **Redeploying a Component**.

5.9 Upgrading Components in Batches

After components are created and deployed, you can reconfigure and deploy multiple **Running** and **Not ready** components of the same application in rolling release mode. In rolling release mode, only one or more instances are upgraded at a time and then added to the production environment. This process repeats until all old instances are upgraded. Services will not be interrupted during the upgrade.

For details about how to upgrade a single component, see **Upgrading a Single Component**.

NOTICE

If the component is deployed based on CCE, it is recommended that the total number of component instances to be upgraded in batches be less than or equal to 30. Otherwise, CCE will limit the traffic and the upgrade will take a long time.

Procedure

Step 1 Log in to ServiceStage.

Step 2 Choose **Application Management**.

Step 3 Click the application where the target components are located. The **Overview** page of the application is displayed.

Step 4 Select the components to be upgraded in batches in **Component List** and click **Bulk Upgrade**.

Step 5 Set the version configuration information of the components to be upgraded by referring to the following table.

Parameter	Description
Component Version	Target version of the upgraded component. • By default, the version number is the time when you start to upgrade the component. The format is yyyy.mmdd.hhmms, where **s** is the ones place of the second in the timestamp. For example, if the timestamp is 2022.0803.104321, the version number is 2022.0803.10431. • You can also customize the version number in the format of A.B.C, or A.B.C.D. A. B, C, and D are natural numbers, for example, 1.0.0 or 1.0.0.0. **NOTICE** The customized version number must be unique and cannot be the same as any historical version number of the component.
Software Package/ Image Package/ Source Code Repository	Click  and select the software package, source code repository, or image package again. For details, see **Component Source**.
Deployment Batches	Number of batches in which component instances are upgraded. The value range is [1, Total number of instances]. Total number of instances refers to the number of running instances of the component. For example, if there are 4 component instances and **Deployment Batches** is set to **2**, these component instances are upgraded in two batches, and each batch involves two component instances.

Parameter	Description
Operation	• Click in the **Operation** column of a component to cancel the upgrade. • Click **Advanced Settings** in the **Operation** column of a CCE-deployed component to configure advanced settings. Set **Microservice Engine**, **Component Configuration**, **Deployment Configuration**, and **O&M Monitoring** by referring to **Step 13**.

Step 6 (Optional) Click **Pre-check Advanced Settings** to check whether the advanced settings of each component are correct.

Step 7 Click **Complete and Execute**.

Wait until the component status changes from **Upgrading/Rolling back** to **Running**, indicating that the component version configuration is successfully upgraded.

----End

Follow-Up Operations

Operation	Description
Rolling Back a Component	After the version configuration of all component instances is upgraded to the new version, if you need to roll back the component to the source version, see **Rolling Back a Component**.
Redeploying a Component	You can select a historical version configuration from the deployment record list and use the version configuration as a template to redeploy components. For details, see **Redeploying a Component**.

5.10 Cloning Components in Batches

This avoids complex and repeated configurations because multiple microservice components may have same configurations, for example, scheduling policy, AS policy, data storage, log storage, and lifecycle configurations, and only minor differences need to be modified. ServiceStage provides the function of cloning and deploying components in batches to improve component deployment efficiency and user experience.

Only components in the **Running** and **Not ready** states can be cloned.

Procedure

Step 1 Log in to ServiceStage.

Step 2 Choose **Application Management**.

Step 3 Click the application where the target components are located. The **Overview** page of the application is displayed.

Step 4 Select the components to be cloned in batches in **Component List** and click **Bulk Clone**.

Step 5 Set the version configuration information of the components to be cloned by referring to the following table.

Parameter	Description
Component	Name of a component, which cannot be changed after the component is deployed. Kubernetes environment: ● Components with the same name in different applications can be deployed in the same environment. ● Components with the same name in the same application can be deployed in different environments. VM environment: ● Components with the same name in different applications can be deployed in the same environment. ● Components with the same name in the same application can be deployed in different environments.
Component Version	Version number of a component. ● By default, the version number is the time when you start to clone components in batches. The format is yyyy.mmdd.hhmms, where **s** is the ones place of the second in the timestamp. For example, if the timestamp is 2022.0803.104321, the version number is 2022.0803.10431. ● You can also customize the version number in the format of A.B.C, or A.B.C.D. A. B, C, and D are natural numbers, for example, 1.0.0 or 1.0.0.0. **NOTICE** The customized version number must be unique and cannot be the same as any historical version number of the component. You can perform the following operations to synchronize component versions in batches: 1. Move the cursor to the **Component Version** text box of the specified component and click  . 2. Select other components whose versions need to be synchronized. 3. Click **OK**.

Parameter	Description
Application	Select the application to which the component belongs. You can perform the following operations to synchronize the applications to which components belong in batches: 1. Move the cursor to the **Application** drop-down list of the specified component and click  . 2. Select other components whose applications need to be synchronized. 3. Click **OK**.
Environment	Select the component deployment environment. Only the Kubernetes environment can be selected. You can perform the following operations to synchronize the environments to which components belong in batches: 1. Move the cursor to the **Environment** drop-down list of the specified component and click  . 2. Select other components whose environments need to be synchronized. 3. Click **OK**.
Namespace	Select the namespace of the CCE cluster in the environment where the build is executed, which is used to isolate build data. For details, see **Managing Namespaces**. You can perform the following operations to synchronize the namespaces to which components belong in batches: 1. Move the cursor to the **Namespace** drop-down list of the specified component and click  . 2. Select other components whose namespaces need to be synchronized. 3. Click **OK**.
Software Package/ Image Package/ Source Code Repository	Click  and select the software package, source code repository, or image package again. For details, see **Component Source**.

Parameter	Description
Instances	Set the number of component instances running in the environment. The value range is 1–200. You can perform the following operations to synchronize the number of component instances in batches: 1. Move the cursor to the **Instances** text box of the specified component and click  . 2. Select other components whose number of component instances need to be synchronized. 3. Click **OK**.

Parameter	Description
Operation	• Click in the **Operation** column of a component to cancel the clone. • Click **Advanced Settings** in the **Operation** column of a component to configure advanced settings. – Set public network access parameters by referring to **Step 11**. – Set **Microservice Engine**, **Distributed Cache Service**, **Relational Database Service**, **Component Configuration**, **Deployment Configuration**, and **O&M Monitoring** by referring to **Step 13**. **NOTE** • You can perform the following operations to synchronize the bound microservice engines, distributed caches, or cloud databases of components in batches: 1. Click to expand **Microservice Engine**, **Distributed Cache Service**, or **Relational Database Service** under **Advanced Settings**. 2. Move the cursor to the microservice engine, distributed cache, or cloud database bound to the specified component, and click . 3. Select other components whose configurations need to be synchronized. 4. Click **OK**. • You can perform the following operations to delete microservice engines, distributed caches, or cloud databases for components in batches: 1. Click to expand **Microservice Engine**, **Distributed Cache Service**, or **Relational Database Service** under **Advanced Settings**. 2. Move the cursor to the microservice engine, distributed cache, or cloud database bound to the specified component, and click . 3. Click **Delete**, select other components whose configurations need to be synchronized, and click **OK**. To delete the microservice engine, distributed cache, or cloud database only for the current component, click **Cancel**. • You can perform the following operations to rebind a microservice engine, distributed cache, or cloud database to a component: 1. Click to expand **Microservice Engine**, **Distributed Cache Service**, or **Relational Database Service** under **Advanced Settings**. 2. Move the cursor to the microservice engine, distributed cache, or cloud database bound to the specified component, and click . 3. Select a managed microservice engine, distributed cache, or cloud database in the current environment, and click **OK**.

Step 6 (Optional) Click **Pre-check Advanced Settings** to check whether the advanced settings of each component are correct.

Step 7 Click **Complete and Execute**.

Wait until the component statuses change from **Initializing** to **Running**, indicating that the components have been cloned.

----End

5.11 Synchronizing Component Configurations in Batches

You can synchronize latest configurations after CCE-based component workload changes (such as workloads upgraded using CCE).

Procedure

Step 1 Log in to ServiceStage.

Step 2 Choose **Application Management**.

Step 3 Click the application where the target components are located. The **Overview** page of the application is displayed.

Step 4 Select the components whose configurations are to be synchronized in batches in **Component List** and click **Batch Sync**.

----End

5.12 Rolling Back a Component

You can roll back a component from the latest version to the version before the upgrade or redeployment.

A component that has been rolled back cannot be rolled back again.

Procedure

Step 1 Log in to ServiceStage.

Step 2 Use either of the following methods to go to the **Deployment Records** page.

- On the **Application Management** page, click the application to which the component belongs, and click the target component in **Component List**. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu, and choose **Deployment Records** in the left navigation pane.
- On the **Component Management** page, click the target component. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu, and choose **Deployment Records** in the left navigation pane.

Step 3 In the **Deployment Records** list, select the deployment record of the latest version.

Step 4 Click **Roll Back**.

Step 5 In the displayed dialog box, click **OK**.

After the rollback is complete, the component will be rolled back to the source version.

----End

5.13 Redeploying a Component

5.13.1 Single-batch Release

You can select a historical version configuration from the deployment record list and use the version configuration as a template to redeploy components in single-batch release mode.

In single-batch release mode, all instances are redeployed at a time. During the deployment, component services will be interrupted. This is applicable to the test deployment scenario or the deployment scenario where services are to be stopped. The deployment takes a short time.

NOTE

Only components deployed in the Kubernetes environment can be redeployed in single-batch release mode.

The component version configuration that has been rolled back by referring to **Rolling Back a Component** cannot be used as a template to redeploy the component.

Prerequisites

You have upgraded a component. For details, see **Upgrading a Single Component** or **Upgrading Components in Batches**.

Procedure

Step 1 Log in to ServiceStage.

Step 2 Use either of the following methods to go to the **Deployment Records** page.

- On the **Application Management** page, click the application to which the component belongs, and click the target component in **Component List**. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu, and choose **Deployment Records** in the left navigation pane.
- On the **Component Management** page, click the target component. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu, and choose **Deployment Records** in the left navigation pane.

Step 3 In the **Deployment Records** list, select the deployment record of the historical version to be used as the configuration template.

Step 4 Click **Redeploy** in the upper right corner of the page. The **Redeploy** dialog box is displayed.

Step 5 Select **Single-batch Release** for **Deployment Type** and click **OK**.

Step 6 Configure the component version by referring to the following table. Parameters marked with an asterisk (*) are mandatory.

Parameter	Description
Stack	The value is fixed to the configuration of the selected historical version and cannot be changed.
*YAML Mode	Uses YAML configurations to redeploy components. • Disabled: The GUI configurations are used to redeploy components. • Enabled: The YAML configurations are used to redeploy components. The latest load information of the component is automatically synchronized from CCE where the target component is deployed for component redeployment after modification. Alternatively, click **Import YAML File** to import the edited YAML configuration file of the target component. **NOTE** If YAML configurations are used to redeploy components, the parameters in the YAML configuration file are described in **Deployment**.
*Software Package/ Image	The value is fixed to the component source selected during component creation and deployment. • **YAML Mode** disabled: If you select **Source code repository**, create authorization by referring to **Authorizing a Repository** and set the code source. If you select a software package or image package, the option is fixed to the software package type (JAR, WAR, or ZIP) or image package type selected during component creation and deployment. It is determined by the selected technology stack type. For details, see **Table 5-1**. • **YAML Mode** enabled: If you select **Source code repository**, create authorization by referring to **Authorizing a Repository** and set the code source. If you select a software package, the option is fixed to the software package type (JAR, WAR, or ZIP) selected during component creation and deployment. It is determined by the selected technology stack type. For details, see **Table 5-1**.
*Upload Method	• **YAML Mode** disabled: If the component source is software package or image package, select an uploaded software package or image package. For details about the upload method, see **Component Source**. • **YAML Mode** enabled: If the component source is software package, select an uploaded software package. For details about the upload method, see **Component Source**.

Parameter	Description
*Command	This parameter is mandatory when **YAML Mode** is disabled, the component source is **Source code repository**, the component is deployed in the Kubernetes environment, and the selected technology stack type is Java, Tomcat, Node.js, Python, or PHP. • **Default command or script**: preferentially executes **build.sh** in the **root** directory. If **build.sh** does not exist, the code will be compiled using the common method of the selected language. Example: **mvn clean package** for Java. • **Custom command**: Commands are customized using the selected language. Alternatively, the default command or script is used after **build.sh** is modified. **NOTICE** – If **Custom command** is selected, exercise caution when inputting sensitive information in the **echo**, **cat**, or **debug** command, or encrypt sensitive information to avoid information leakage. – To run the compilation command in the project subdirectory, you need to go to the project subdirectory and then run other script commands. For example: **cd ./weather/** **mvn clean package**
*Dockerfile Address	This parameter is mandatory when **YAML Mode** is disabled, the component source is **Source code repository**, the component is deployed in the Kubernetes environment, and the selected technology stack type is Java, Tomcat, Node.js, Python, or PHP. **Dockerfile Address** is the directory where the Dockerfile is located relative to the root directory (./) of the project. The Dockerfile is used to build an image. If **Dockerfile Address** is not specified, the system searches for the Dockerfile in the root directory of the project by default. If the Dockerfile does not exist in the root directory, the system automatically generates the Dockerfile based on the selected operating environment.
*Component Version	Component version number, which can be automatically generated or customized. • Automatically-generated: Click **Generate**. By default, the version number is the time when you click **Generate**. The format is yyyy.mmdd.hhmms, where **s** is the ones place of the second in the timestamp. For example, if the timestamp is 2022.0803.104321, the version number is 2022.0803.10431. • Customized: Enter a value in the format of A.B.C or A.B.C.D. A, B, C, and D are natural numbers. For example, 1.0.0 or 1.0.0.0. **NOTICE** The customized version number must be unique and cannot be the same as any historical version number of the component.

Parameter	Description
Resources	The value is fixed to the configuration of the selected historical version and cannot be changed. The configured resources are displayed when **YAML Mode** is disabled.
JVM Parameters	The value is fixed to the configuration of the selected historical version and cannot be changed. This parameter is available when **YAML Mode** is disabled and the technology stack type is Java or Tomcat. It configures the memory size during Java code running.
Tomcat	The value is fixed to the configuration of the selected historical version and cannot be changed. This parameter is available when **YAML Mode** is disabled and the technology stack type is Tomcat. It configures parameters such as the request path and port number of Tomcat.
Advanced Settings	The value is fixed to the configuration of the selected historical version and cannot be changed. The configured advanced settings are displayed when **YAML Mode** is disabled.

Step 7 Click **Upgrade**.

In the **Deployment Records** area, you can view the deployment progress and wait until the deployment is complete.

----End

5.13.2 Rolling Release

You can select a historical version configuration from the deployment record list and use the version configuration as a template to redeploy components in rolling release mode.

In rolling release mode, only one or more instances are deployed at a time and then added to the production environment. This process repeats until all old instances are upgraded. Services will not be interrupted during the deployment.

The component version configuration that has been rolled back by referring to **Rolling Back a Component** cannot be used as a template to redeploy the component.

Prerequisites

You have upgraded a component. For details, see **Upgrading a Single Component** or **Upgrading Components in Batches**.

Procedure

Step 1 Log in to ServiceStage.

Step 2 Use either of the following methods to go to the **Deployment Records** page.

- On the **Application Management** page, click the application to which the component belongs, and click the target component in **Component List**. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu, and choose **Deployment Records** in the left navigation pane.
- On the **Component Management** page, click the target component. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu, and choose **Deployment Records** in the left navigation pane.

Step 3 In the **Deployment Records** list, select the deployment record of the historical version to be used as the configuration template.

Step 4 Click **Redeploy** in the upper right corner of the page. The **Redeploy** dialog box is displayed.

Step 5 Select **Rolling Release** for **Deployment Type** and click **OK**.

Step 6 Configure the component version by referring to the following table. Parameters marked with an asterisk (*) are mandatory.

Parameter	Description
Stack	The value is fixed to the configuration of the selected historical version and cannot be changed.
*YAML Mode	Uses YAML configurations to redeploy components if the component is deployed in the Kubernetes environment. • Disabled: The GUI configurations are used to redeploy components. • Enabled: The YAML configurations are used to redeploy components. The latest load information of the component is automatically synchronized from CCE where the target component is deployed for component redeployment after modification. Alternatively, click **Import YAML File** to import the edited YAML configuration file of the target component. **NOTE** If YAML configurations are used to redeploy components, the parameters in the YAML configuration file are described in **Deployment**.

Parameter	Description
*Software Package/ Image	The value is fixed to the component source selected during component creation and deployment. • **YAML Mode** disabled: If you select **Source code repository**, create authorization by referring to **Authorizing a Repository** and set the code source. If you select a software package or image package, the option is fixed to the software package type (JAR, WAR, or ZIP) or image package type selected during component creation and deployment. It is determined by the selected technology stack type. For details, see **Table 5-1**. • **YAML Mode** enabled: If you select **Source code repository**, create authorization by referring to **Authorizing a Repository** and set the code source. If you select a software package, the option is fixed to the software package type (JAR, WAR, or ZIP) selected during component creation and deployment. It is determined by the selected technology stack type. For details, see **Table 5-1**.
*Upload Method	• **YAML Mode** disabled: If the component source is software package or image package, select an uploaded software package or image package. For details about the upload method, see **Component Source**. • **YAML Mode** enabled: If the component source is software package, select an uploaded software package. For details about the upload method, see **Component Source**.
*Command	This parameter is mandatory when **YAML Mode** is disabled, the component source is **Source code repository**, the component is deployed in the Kubernetes environment, and the selected technology stack type is Java, Tomcat, Node.js, Python, or PHP. • **Default command or script**: preferentially executes **build.sh** in the **root** directory. If **build.sh** does not exist, the code will be compiled using the common method of the selected language. Example: **mvn clean package** for Java. • **Custom command**: Commands are customized using the selected language. Alternatively, the default command or script is used after **build.sh** is modified. **NOTICE** – If **Custom command** is selected, exercise caution when inputting sensitive information in the **echo**, **cat**, or **debug** command, or encrypt sensitive information to avoid information leakage. – To run the compilation command in the project subdirectory, you need to go to the project subdirectory and then run other script commands. For example: **cd ./weather/** **mvn clean package**

Parameter	Description
*Dockerfile Address	This parameter is mandatory when **YAML Mode** is disabled, the component source is **Source code repository**, the component is deployed in the Kubernetes environment, and the selected technology stack type is Java, Tomcat, Node.js, Python, or PHP. **Dockerfile Address** is the directory where the Dockerfile is located relative to the root directory (./) of the project. The Dockerfile is used to build an image. If **Dockerfile Address** is not specified, the system searches for the Dockerfile in the root directory of the project by default. If the Dockerfile does not exist in the root directory, the system automatically generates the Dockerfile based on the selected operating environment.
*Component Version	Component version number, which can be automatically generated or customized. ● Automatically-generated: Click **Generate**. By default, the version number is the time when you click **Generate**. The format is yyyy.mmdd.hhmms, where **s** is the ones place of the second in the timestamp. For example, if the timestamp is 2022.0803.104321, the version number is 2022.0803.10431. ● Customized: Enter a value in the format of A.B.C or A.B.C.D. A, B, C, and D are natural numbers. For example, 1.0.0 or 1.0.0.0. **NOTICE** The customized version number must be unique and cannot be the same as any historical version number of the component.
Environment variables	The value is fixed to the configuration of the selected historical version and cannot be changed. This parameter is available when the component is deployed based on VM.
Resources	The value is fixed to the configuration of the selected historical version and cannot be changed. This parameter is available when **YAML Mode** is disabled and the component is deployed based on CCE.
JVM Parameters	The value is fixed to the configuration of the selected historical version and cannot be changed. This parameter is available when **YAML Mode** is disabled and the technology stack type is Java or Tomcat. It configures the memory size during Java code running.
Tomcat	The value is fixed to the configuration of the selected historical version and cannot be changed. This parameter is available when **YAML Mode** is disabled and the technology stack type is Tomcat. It configures parameters such as the request path and port number of Tomcat.

Parameter	Description
Advanced Settings	The value is fixed to the configuration of the selected historical version and cannot be changed. This parameter is available when **YAML Mode** is disabled and the component is deployed based on CCE.
*Deployment Batches	Number of batches in which component instances are upgraded. The value range is [1, Total number of instances]. Total number of instances refers to the number of running instances of the component. For example, if there are 4 component instances and **Deployment Batches** is set to **2**, these component instances are upgraded in two batches, and each batch involves two component instances. This parameter is available when the component is deployed based on CCE.

Step 7 Click **Upgrade**.

In the **Deployment Records** area, you can view the deployment progress and wait until the deployment is complete.

----End

5.13.3 Dark Launch (Canary)

You can select a historical version configuration from the deployment record list and use the version configuration as a template to redeploy components in dark launch (canary) mode.

In dark launch (canary) mode, a certain proportion of instances are upgraded, and traffic is directed to the new version to verify whether functions of the new version are normal. Then, the remaining instances will be upgraded in rolling mode. Dark launch ensures stability of the entire system. During initial dark launch, problems can be detected and fixed.

For details about dark launch (canary) types and details, see **Table 5-3**.

Table 5-3 Dark launch (canary) types and description

Type	Description
Microservice Dark Launch	Applies to ServiceComb and Spring Cloud applications. Dark launch tasks function on microservices. Multiple microservices can work together to roll out new features. 1. The Java, Tomcat, or Docker technology stack must be selected for the component. 2. The component must be bound to a microservice engine with security authentication disabled and multi-language access to service mesh disabled. 3. ServiceComb 2.7.8 or later is required. Spring Cloud Huawei 1.10.4-2021.0.x or later is required.
ELB Dark Launch	Applies to ELB traffic-based components. Dark launch tasks function on ELB. The component must be accessible from the public network and bound to an ELB.

NOTE

Redeployment in dark launch (canary) mode is supported only when the deployment environment is Kubernetes and there are two or more component instances.

The component version configuration that has been rolled back by referring to **Rolling Back a Component** cannot be used as a template to redeploy the component.

Procedure

Step 1 Log in to ServiceStage.

Step 2 Use either of the following methods to go to the **Deployment Records** page.

- On the **Application Management** page, click the application to which the component belongs, and click the target component in **Component List**. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu, and choose **Deployment Records** in the left navigation pane.
- On the **Component Management** page, click the target component. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu, and choose **Deployment Records** in the left navigation pane.

Step 3 In the **Deployment Records** list, select the deployment record of the historical version to be used as the configuration template.

Step 4 Click **Redeploy** in the upper right corner of the page. The **Redeploy** dialog box is displayed.

Step 5 Select **Dark Launch (Canary)** for **Deployment Type** and click **OK**.

Step 6 Configure the component version by referring to the following table. Parameters marked with an asterisk (*) are mandatory.

NOTICE

During the dark launch upgrade of a component microservice in this operation, do not use CSE to perform dark launch of the component microservice at the same time. Otherwise, this operation fails.

For details about how to perform dark launch of a component microservice through CSE, see **Dark Launch**.

Parameter	Description
Stack	The value is fixed to the configuration of the selected historical version and cannot be changed.
*Software Package/ Image	The value is fixed to the component source selected during component creation and deployment. If you select **Source code repository**, create authorization by referring to **Authorizing a Repository** and set the code source. If you select a software package or image package, the option is fixed to the software package type (JAR, WAR, or ZIP) or image package type selected during component creation and deployment. It is determined by the selected technology stack type. For details, see **Table 5-1**.
*Upload Method	If the component source is software package or image package, select an uploaded software package or image package. For details about the upload method, see **Component Source**.
*Command	This parameter is mandatory when the component source is **Source code repository**, the component is deployed in the Kubernetes environment, and the selected technology stack type is Java, Tomcat, Node.js, Python, or PHP. • **Default command or script**: preferentially executes **build.sh** in the **root** directory. If **build.sh** does not exist, the code will be compiled using the common method of the selected language. Example: **mvn clean package** for Java. • **Custom command**: Commands are customized using the selected language. Alternatively, the default command or script is used after **build.sh** is modified. **NOTICE** – If **Custom command** is selected, exercise caution when inputting sensitive information in the **echo**, **cat**, or **debug** command, or encrypt sensitive information to avoid information leakage. – To run the compilation command in the project subdirectory, you need to go to the project subdirectory and then run other script commands. For example: **cd ./weather/** **mvn clean package**

Parameter	Description
*Dockerfile Address	This parameter is mandatory when the component source is **Source code repository**, the component is deployed in the Kubernetes environment, and the selected technology stack type is Java, Tomcat, Node.js, Python, or PHP. **Dockerfile Address** is the directory where the Dockerfile is located relative to the root directory (./) of the project. The Dockerfile is used to build an image. If **Dockerfile Address** is not specified, the system searches for the Dockerfile in the root directory of the project by default. If the Dockerfile does not exist in the root directory, the system automatically generates the Dockerfile based on the selected operating environment.
*Component Version	Component version number, which can be automatically generated or customized. ● Automatically-generated: Click **Generate**. By default, the version number is the time when you click **Generate**. The format is yyyy.mmdd.hhmms, where **s** is the ones place of the second in the timestamp. For example, if the timestamp is 2022.0803.104321, the version number is 2022.0803.10431. ● Customized: Enter a value in the format of A.B.C or A.B.C.D. A, B, C, and D are natural numbers. For example, 1.0.0 or 1.0.0.0. **NOTICE** The customized version number must be unique and cannot be the same as any historical version number of the component.
Resources	The value is fixed to the configuration of the selected historical version and cannot be changed.
JVM Parameters	The value is fixed to the configuration of the selected historical version and cannot be changed. This parameter is available when the technology stack type is Java or Tomcat. It configures the memory size during Java code running.
Tomcat	The value is fixed to the configuration of the selected historical version and cannot be changed. This parameter is available when the technology stack type is Tomcat. It configures parameters such as the request path and port number of Tomcat.
Advanced Settings	The value is fixed to the configuration of the selected historical version and cannot be changed.
Dark Launch Policy	● **Traffic Ratio**: percentage of traffic directed to the new version. ● **Current Traffic Ratio**: percentage of traffic directed to the current version.

Parameter	Description
*First Batch of Dark Launch Instances	Number of instances for dark launch in the first batch. The value range is [1, Total number of instances – 1]. Total number of instances refers to the number of running instances of the component. For example, if there are 6 component instances and **First Batch of Dark Launch Instances** is set to **1**, 1 instance will be upgraded in the first batch.
Deployment Batch with Remaining Instances	Number of batches whose remaining instances will be upgraded. For example, if there are 6 component instances, **First Batch of Dark Launch Instances** is set to **1**, and **Deployment Batch with Remaining Instances** is set to **3**, there are 5 instances remaining to be deployed in 3 batches, and these 5 instances will be upgraded in the sequence 2:2:1

Step 7 Click **Upgrade**.

In the **Deployment Records** area, you can view the deployment progress and wait until the deployment is complete.

----End

5.14 Configuring the Component Access Mode

This section describes how to configure the access mode for a component. After the configuration, you can access the services provided by the component in the configured mode.

You can only configure the component access mode for components that are deployed in the Kubernetes environment and are in the **Running** state.

Procedure

Step 1 Log in to ServiceStage.

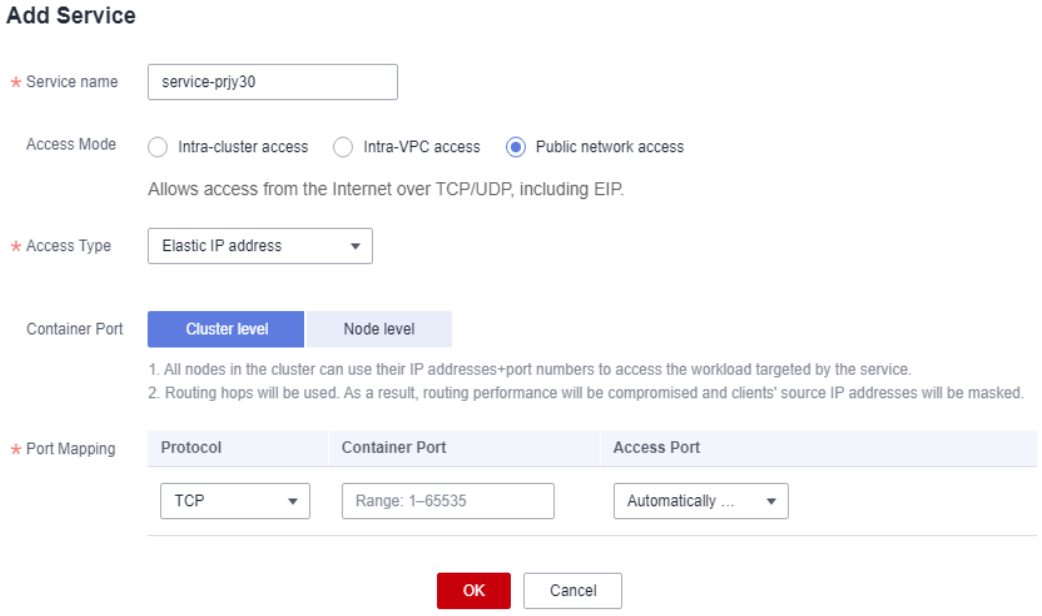
Step 2 Use either of the following methods to go to the **Access Mode** page.

- On the **Application Management** page, click the application to which the component belongs, and click the target component in **Component List**. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu, and choose **Access Mode** in the left navigation pane.
- On the **Component Management** page, click the target component. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu, and choose **Access Mode** in the left navigation pane.

Step 3 Click **Add Service** and set the following parameters. Parameters marked with an asterisk (*) are mandatory.

Parameter	Description
*Service Name	The service name can be the same as the component name.
Access Mode	The options are as follows: • **Intra-cluster access**: allows access from other services in the same cluster over TCP/UDP. • **Intra-VPC access**: allows access from other services in the same VPC over TCP/UDP. • **Public network access**: allows access from the Internet over TCP/UDP, including public EIP.
Intra-VPC Load Balancing	This parameter is available when **Access Mode** is set to **Intra-VPC access**.
*Access Type	• This parameter is available when **Access Mode** is set to **Intra-VPC access** and **Intra-VPC load balancing** is enabled. • This parameter is available when **Access Mode** is set to **Public network access**.
Service Affinity	This parameter is available when **Access Mode** is set to **Intra-VPC access** or **Public network access**.
*Port Mapping	Sets **Protocol**, **Container Port**, and **Access Port** for accessing the service.

Figure 5-13 Setting the component access mode



Step 4 Click **OK**.

----End

5.15 Changing the Component Access Domain Name

For components with enabled public network access and set access domain names, you can change the domain names after the components are deployed.

Prerequisites

- An automatically generated domain name is valid only for seven days. After the validity period expires, the domain name must be changed to a custom domain name.
- You can change the domain name of a component that has been created and deployed, only when the component is in the **Running** state.
- You have obtained the domain name from the domain name provider.
- You have obtained the elastic public IP address of the ELB bound to the component.

Procedure

Step 1 Log in to ServiceStage.

Step 2 Use either of the following methods to go to the **Access Mode** page.

- On the **Application Management** page, click the application to which the component belongs, and click the target component in **Component List**. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu, and choose **Access Mode** in the left navigation pane.
- On the **Component Management** page, click the target component. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu, and choose **Access Mode** in the left navigation pane.

Step 3 Click **Set domain**.

1. Enter the obtained **Domain Name**.
2. Enter **Listening Port**.
3. (Optional) Enable **HTTPS**.
   - Click **Use existing** to select an existing certificate.
   - Click **Create new** to create a server certificate. For details, see **Creating a Certificate**.

----End

5.16 Configuring a Scaling Policy of a Component Instance

After scaling policies are configured, instances can be automatically added or deleted based on resource changes or a specified schedule. This reduces manual resource adjustment to cope with service changes and service peak, helping you save resources and labor costs.

- Graceful scaling-in

  You can configure graceful scale-in policies only for the components deployed in the Kubernetes environment.

  You can set a graceful scale-in time window to save important data before a component instance stops. The value ranges from 0 to 9999, in seconds. The default value is **30**. For example, if an application has two instances and only one instance will be kept after the scale-in operation, you can still perform certain operations on the instance to be stopped in the specified time window.

- Manual scaling

  The number of instances will be increased or decreased immediately after the configuration is complete.

- HPA

  Only CCE cluster 1.13 and later support HPA.

  HPA is a built-in component of Kubernetes, which enables horizontal scaling of pods. It supports the application-level cooldown time window and scaling threshold functions based on the Kubernetes HPA.

Configuring a Graceful Scale-In Policy

Step 1 Log in to ServiceStage.

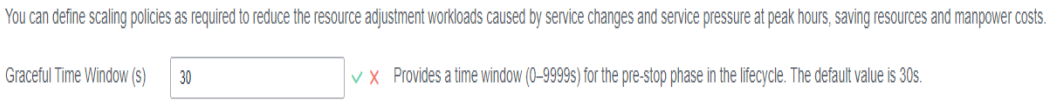
Step 2 Use either of the following methods to go to the **Scaling** page.

- On the **Application Management** page, click the application to which the component belongs, and click the target component in **Component List**. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu, and choose **Scaling** in the left navigation pane.
- On the **Component Management** page, click the target component. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu, and choose **Scaling** in the left navigation pane.

Step 3 On the **Scaling** page, configure a graceful scale-in policy.

Set **Graceful Time Window (s)**. Specifically, click  , enter a value, and click  .

Figure 5-14 Configuring a graceful scale-in policy



----End

Configuring a Manual Scaling Policy

Step 1 Log in to ServiceStage.

Step 2 Use either of the following methods to go to the **Scaling** page.

- On the **Application Management** page, click the application to which the component belongs, and click the target component in **Component List**. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu, and choose **Scaling** in the left navigation pane.

- On the **Component Management** page, click the target component. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu, and choose **Scaling** in the left navigation pane.

Step 3 In the **Manual Scaling** area on the **Scaling** page, configure a manual scaling policy.

- To deploy a component in the Kubernetes environment, perform the following operations:

  a. Click and change the number of instances.

  b. Click for the instance scaling to take effect.

  Figure 5-15 Configuring a manual scaling policy (for Kubernetes components)

- For components deployed on a VM, perform the following operations:

  a. In the **Instances** area, click .

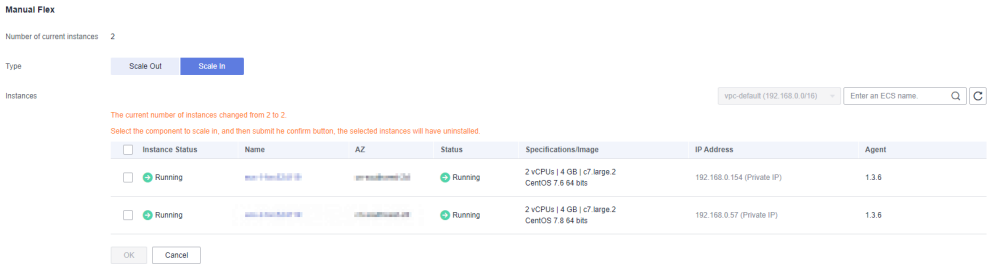
  b. Select **Type** and add or delete component running instances based on the site requirements.

  If **Type** is set to **Scale Out**, click **Add ECS** and create an ECS to run new component instances. For details, see **Purchasing ECSs**.

  If **Type** is set to **Scale In**, the number of running component instances can be reduced to 1.

  c. Click **OK**.

  Figure 5-16 Configuring a manual scaling policy (for VM components)

----End

Configuring an HPA Policy

Step 1 Log in to ServiceStage.

Step 2 Use either of the following methods to go to the **Scaling** page.

- On the **Application Management** page, click the application to which the component belongs, and click the target component in **Component List**.

Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu, and choose **Scaling** in the left navigation pane.

- On the **Component Management** page, click the target component. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu, and choose **Scaling** in the left navigation pane.

Step 3 On the **Scaling** page, click  next to **Auto Scaling by HPA** to enable auto scaling policy configuration. The **Policy** page is displayed.

- If metrics-server has not been installed in the CCE cluster, go to **Step 4**.
- If metrics-server has been installed in the CCE cluster, go to **Step 6**.

Step 4 Click **Configure Now** to install the metrics-server add-on on the CCE console.

Install the metrics-server add-on for the CCE cluster. For details, see **metrics-server**.

Step 5 After the add-on is installed, return to the **Policy** page and click **refresh**.

Step 6 Configure the scaling policy.

1. Policy Name

   Enter a policy name. After an auto scaling policy is configured, its name cannot be changed.

2. Cooldown Period

   Enter a scale-out/scale-in cooldown period.

   The same scaling operation will not be triggered again within the specified period.

3. Pod Range

   Enter the minimum and maximum numbers of instances.

   After the policy is triggered, the workload pods are scaled within this range.

4. Trigger Condition

   You can configure trigger condition on the GUI or by editing the YAML file.

   - GUI

     Set **Desired Value** and **Threshold** (scale-in and scale-out thresholds) of **CPU usage** and **Memory usage**.

     After the policy is triggered, the number of instances to be scaled is calculated by rounding up the value of (Current CPU or memory usage/ Expected value x Number of running instances).

     - Scale-in is triggered when the current CPU or memory usage is less than the scale-in threshold.
     - Scale-out is triggered when the current CPU or memory usage is greater than the scale-out threshold.

   - YAML

```
metrics:
  - type: Resource
    resource:
      name: cpu
      target:
        type: Utilization
        averageUtilization: 50
```



```
- type: Resource
  resource:
    name: memory
    target:
      type: Utilization
      averageUtilization: 50
- type: Pods
  pods:
    metric:
      name: packets-per-second
    target:
      type: AverageValue
      averageValue: 1k
- type: Object
  object:
    metric:
      name: requests-per-second
    describedObject:
      apiVersion: networking.k8s.io/v1beta1
      kind: Ingress
      name: main-route
    target:
      type: Value
      value: 10k
```

As shown in the preceding example, in addition to using the CPU and memory usage as metrics, you can use the YAML format to customize metric parameters and support more metrics such as pods, object, and external.

NOTE

To configure custom metric parameters by using **YAML**, ensure that the prometheus add-on has been installed for the CCE cluster.

Install the prometheus add-on for the CCE cluster. For details, see **prometheus**.

Step 7 Click **OK**.

NOTE

After the HPA policy is configured, you can perform the following operations based on service requirements:

- **Modifying an HPA Policy**
- **Viewing the Running Status of the HPA Policy**
- **Deleting an HPA Policy**

----End

Modifying an HPA Policy

You can edit an existing HPA policy and reconfigure policy parameters.

Step 1 Log in to ServiceStage.

Step 2 Use either of the following methods to go to the **Scaling** page.

- On the **Application Management** page, click the application to which the component belongs, and click the target component in **Component List**. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu, and choose **Scaling** in the left navigation pane.
- On the **Component Management** page, click the target component. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu, and choose **Scaling** in the left navigation pane.

Step 3 On the **Scaling** page, choose **Policy**, click **Edit Policy**, and reconfigure parameters.

1. Cooldown Period

   Change the scale-out/scale-in cooldown period.

2. Pod Range

   Change the minimum and maximum numbers of instances.

3. Trigger Condition

   You can change trigger condition on the GUI or by editing the YAML file.

   – GUI

     Change **Desired Value** and **Threshold** (scale-in and scale-out thresholds) of **CPU usage** and **Memory usage**.

   – YAML

     You can use the YAML format to customize metric parameters and support more metrics such as pods, objects, and external.

     NOTE

     To configure custom metric parameters by using **YAML**, ensure that the prometheus add-on has been installed for the CCE cluster.

     Install the prometheus add-on for the CCE cluster. For details, see **prometheus**.

Step 4 Click **OK**.

----End

Viewing the Running Status of the HPA Policy

ServiceStage allows you to view the running status and events of a configured HPA policy.

Step 1 Log in to ServiceStage.

Step 2 Use either of the following methods to go to the **Scaling** page.

- On the **Application Management** page, click the application to which the component belongs, and click the target component in **Component List**. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu, and choose **Scaling** in the left navigation pane.
- On the **Component Management** page, click the target component. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu, and choose **Scaling** in the left navigation pane.

Step 3 On the **Scaling** page:

- Click the **Status** tab to view the policy running status.
- Click the **Event** tab to view events that occur during policy running.

----End

Deleting an HPA Policy

You can delete an HPA policy that is no longer used.

NOTICE

Deleted policies cannot be recovered. Exercise caution when performing this operation.

Step 1 Log in to ServiceStage.

Step 2 Use either of the following methods to go to the **Scaling** page.

- On the **Application Management** page, click the application to which the component belongs, and click the target component in **Component List**. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu, and choose **Scaling** in the left navigation pane.
- On the **Component Management** page, click the target component. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu, and choose **Scaling** in the left navigation pane.

Step 3 On the **Scaling** page, click  on the right of **Auto Scaling by HPA**.

Step 4 Click **OK**.

----End

5.17 Component O&M

5.17.1 Viewing Component Running Metrics

After a component is created and deployed, you can go to its **Metric Monitoring Graphs** page to view the statistics of component running metrics.

Procedure

Step 1 Log in to ServiceStage.

Step 2 Use either of the following methods to go to the **Metric Monitoring Graphs** page.

- On the **Application Management** page, click the application to which the component belongs, and click the target component in **Component List**. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu, and choose **O&M Configurations** > **Monitoring Overview** in the left navigation pane.
- On the **Component Management** page, click the target component. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu, and choose **O&M Configurations** > **Monitoring Overview** in the left navigation pane.

Step 3 On the **Metric Monitoring Graphs** page, view the statistics of component running metrics in the last hour at an interval of 1 minute.

- Click  on the component running metric page for which you want to suspend statistics collection.

- Click on the component running metric page to continue collecting statistics on the running metric.

----End

5.17.2 Customizing Component Running Metrics

After a Kubernetes component is created and deployed, you can go to its **Metric Monitoring Graphs** page to customize the component running metrics to be viewed.

Procedure

Step 1 Log in to ServiceStage.

Step 2 Use either of the following methods to go to the **Metric Monitoring Graphs** page.

- On the **Application Management** page, click the application to which the component belongs, and click the target component in **Component List**. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu, and choose **O&M Configurations** > **Monitoring Overview** in the left navigation pane.
- On the **Component Management** page, click the target component. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu, and choose **O&M Configurations** > **Monitoring Overview** in the left navigation pane.

Step 3 On the **Metric Monitoring Graphs** page, click **Set Metric Monitoring Graph**.

- Select the system metrics to be viewed and select **Statistical Mode**.
- Deselect the system metrics that you do not need to view.

  You can click **Clear** next to **Selected Metric** to clear all selected system metrics.

Step 4 Click **OK**.

----End

5.17.3 Managing Component Logs

5.17.3.1 Managing Component AOM Logs

ServiceStage component logs are connected to AOM by default. You can view, search for, and export logs to locate and rectify faults that occur during component running.

Procedure

Step 1 Log in to ServiceStage.

Step 2 Use either of the following methods to go to the **Running Logs** page.

- On the **Application Management** page, click the application to which the component belongs, and click the target component in **Component List**. Alternatively, right-click the component and go to the component **Overview**

page from the shortcut menu, and choose **O&M Configurations** > **Logs** in the left navigation pane.

- On the **Component Management** page, click the target component. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu, and choose **O&M Configurations** > **Logs** in the left navigation pane.

Step 3 On the **Running Logs** page, manage component running logs by referring to the following table.

Operation	Description
View Logs	1. Select the instance whose logs you want to view. 2. Select the log file you want to view. 3. Select the time range of the logs you want to view. If you select **Custom**, the time range cannot exceed 30 days. 4. Enter keywords in the search box and click to view details about the specified logs.
Export Logs	1. Click **Last Records**. 2. Select the number of log records to be exported. 3. Open the **log.txt** file exported locally and view the exported log records.
View AOM Logs	Click **View AOM Logs**. You can view the component run logs on the AOM console. For details, see **Viewing Log Files**.

NOTE

If you cannot view logs on the ServiceStage **Logs** page, see **Why Can't I View ServiceStage Logs?**

----End

5.17.3.2 Managing Component LTS Logs

5.17.3.2.1 LTS Log Overview

Log Tank Service (LTS) collects log data from hosts and cloud services. By processing massive amounts of logs efficiently, securely, and in real time, LTS provides useful insights for you to optimize the availability and performance of cloud services and applications. It also helps you efficiently perform real-time decision-making, device O&M, and service trend analysis.

ServiceStage allows you to interconnect with LTS to view, search for, and export LTS logs. You can also view container logs to locate and rectify faults that occur during component running.

5.17.3.2.2 Associating an LTS Log Group

After associating a component with an LTS log group, you can view the component running logs collected by LTS on the ServiceStage console and query the logs.

Prerequisites

1. A log group has been created. For details, see **Creating a Log Group**.
2. A log stream has been created. For details, see **Creating a Log Stream**.
3. The path of the host logs to be collected has been configured in the log stream. For details, see **Collecting Logs from CCE**.

Procedure

Step 1 Log in to ServiceStage.

Step 2 Use either of the following methods to go to the **Running Logs** page.

- On the **Application Management** page, click the application to which the component belongs, and click the target component in **Component List**. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu, and choose **O&M Configurations** > **Logs** in the left navigation pane.
- On the **Component Management** page, click the target component. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu, and choose **O&M Configurations** > **Logs** in the left navigation pane.

Step 3 On the **Running Logs** page, click **Associate LTS Log Group**.

Step 4 Select the target log group and click **OK**.

After the log group is associated, you can view the component running logs collected by LTS on the **Running Logs** tab.

Click **Go To** or  to go to the log stream page on the LTS console. You can manage component run logs. For details, see **Log Management**.

----End

5.17.3.2.3 Searching for Running Logs

After associating a component with an LTS log group, you can set a keyword and time range to search for logs.

Prerequisites

The component has been associated with a log group. For details, see **Associating an LTS Log Group**.

Procedure

Step 1 Log in to ServiceStage.

Step 2 Use either of the following methods to go to the **Running Logs** page.

- On the **Application Management** page, click the application to which the component belongs, and click the target component in **Component List**. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu, and choose **O&M Configurations** > **Logs** in the left navigation pane.
- On the **Component Management** page, click the target component. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu, and choose **O&M Configurations** > **Logs** in the left navigation pane.

Step 3 On the **Running Logs** page, select a log stream from the drop-down list.

Step 4 Select a time range in the upper right corner.

- **From now**: queries log data generated in a time range that ends with the current time, such as the previous 1, 5, or 15 minutes. For example, if the current time is 19:20:31 and **1 hour** is selected as the relative time from now, the charts on the dashboard display the log data that is generated from 18:20:31 to 19:20:31.
- **From last**: queries log data generated in a time range that ends with the current time, such as the previous 1 or 15 minutes. For example, if the current time is 19:20:31 and **1 hour** is selected as the relative time from last, the charts on the dashboard display the log data that is generated from 18:00:00 to 19:00:00.
- **Specified**: queries log data that is generated in a specified time range.

Step 5 On the log details page, you can search for logs using the following methods:

- Enter a keyword in the search box and click **Search**, or select a historical record, index field, or keyword from the displayed drop-down list.
- On the **Raw Logs** tab, hover the cursor over a field in the log content, click it, and choose **Copy**, **Add To Search**, or **Exclude from Search** from the displayed menu.
- In the displayed drop-down list, press the up and down arrows on the keyboard to select a keyword or search syntax, press **Tab** or **Enter**, and click **Search**.

----End

5.17.3.2.4 Quickly Querying Logs

To search for logs using a keyword repeatedly, perform the following operations to configure quick query.

Prerequisites

The component has been associated with a log group. For details, see **Associating an LTS Log Group**.

Procedure

To search for logs using a keyword repeatedly, perform the following operations to configure quick query.

Step 1 Log in to ServiceStage.

Step 2 Use either of the following methods to go to the **Running Logs** page.

- On the **Application Management** page, click the application to which the component belongs, and click the target component in **Component List**. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu, and choose **O&M Configurations** > **Logs** in the left navigation pane.
- On the **Component Management** page, click the target component. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu, and choose **O&M Configurations** > **Logs** in the left navigation pane.

Step 3 On the **Running Logs** page, select a log stream from the drop-down list.

Step 4 Click .

Step 5 Enter a name and keyword, and click **OK**.

----End

5.17.3.2.5 Using Visualization to Analyze Logs

Visualization support SQL query and analysis for structured log fields. After log structuring, wait about 1–2 minutes for SQL query and analysis.

Prerequisites

1. The log stream has been structured. For details, see **Log Structuring**
2. The component has been associated with a log group. For details, see **Associating an LTS Log Group**.

Procedure

Step 1 Log in to ServiceStage.

Step 2 Use either of the following methods to go to the **Running Logs** page.

- On the **Application Management** page, click the application to which the component belongs, and click the target component in **Component List**. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu, and choose **O&M Configurations** > **Logs** in the left navigation pane.
- On the **Component Management** page, click the target component. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu, and choose **O&M Configurations** > **Logs** in the left navigation pane.

Step 3 On the **Running Logs** page, select a log stream from the drop-down list.

Step 4 Select a time range in the upper right corner.

- **From now**: queries log data generated in a time range that ends with the current time, such as the previous 1, 5, or 15 minutes. For example, if the current time is 19:20:31 and **1 hour** is selected as the relative time from now, the charts on the dashboard display the log data that is generated from 18:20:31 to 19:20:31.

- **From last**: queries log data generated in a time range that ends with the current time, such as the previous 1 or 15 minutes. For example, if the current time is 19:20:31 and **1 hour** is selected as the relative time from last, the charts on the dashboard display the log data that is generated from 18:00:00 to 19:00:00.
- **Specified**: queries log data that is generated in a specified time range.

Step 5 Click the **Visualization** tab.

Step 6 Click the SQL query box and enter the SQL query statement.

According to the data returned by the SQL query, select a chart type to display the query result. For details, see **LTS Visualization**.

----End

5.17.3.2.6 Viewing Real-Time Logs

After associating a component with an LTS log group, you can view the component logs reported in real time.

Prerequisites

The component has been associated with a log group. For details, see **Associating an LTS Log Group**.

Procedure

Step 1 Log in to ServiceStage.

Step 2 Use either of the following methods to go to the **Running Logs** page.

- On the **Application Management** page, click the application to which the component belongs, and click the target component in **Component List**. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu, and choose **O&M Configurations** > **Logs** in the left navigation pane.
- On the **Component Management** page, click the target component. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu, and choose **O&M Configurations** > **Logs** in the left navigation pane.

Step 3 On the **Running Logs** page, select a log stream from the drop-down list.

Step 4 Click the **Real-Time Logs** tab and view the real-time logs in the **Log Content** area.

One log record is reported every one minute. You can control log display by clicking **Clear** or **Pause** in the upper right corner.

- **Clear**: clears all displayed logs.
- **Pause**: pauses the real-time log display.

  After you click **Pause**, the button changes to **Continue**. You can click **Continue** to resume the log display.

NOTE

If you are viewing real-time logs, do not switch to another page. Or, logs will not be loaded in real time. The next time you access the tab, the logs that were shown before you left the tab will disappear.

----End

5.17.3.2.7 Unbinding an LTS Log Group

You can unbind an LTS log group if it is no longer used.

Prerequisites

The component has been associated with a log group. For details, see **Associating an LTS Log Group**.

Procedure

Step 1 Log in to ServiceStage.

Step 2 Use either of the following methods to go to the **Running Logs** page.

- On the **Application Management** page, click the application to which the component belongs, and click the target component in **Component List**. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu, and choose **O&M Configurations** > **Logs** in the left navigation pane.
- On the **Component Management** page, click the target component. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu, and choose **O&M Configurations** > **Logs** in the left navigation pane.

Step 3 On the **Running Logs** page, click **Unbind**.

Step 4 Click **OK**.

After the unbinding, you cannot view LTS logs on the **Running Logs** page.

----End

5.17.3.3 Viewing Container Logs

ServiceStage allows you to view container logs of components deployed in the Kubernetes environment to locate and rectify faults that occur during component running.

Procedure

Step 1 Log in to ServiceStage.

Step 2 Use either of the following methods to go to the **Running Logs** page.

- On the **Application Management** page, click the application to which the component belongs, and click the target component in **Component List**. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu, and choose **O&M Configurations** > **Logs** in the left navigation pane.

- On the **Component Management** page, click the target component. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu, and choose **O&M Configurations** > **Logs** in the left navigation pane.

Step 3 On the **Container Logs** page, view the component container logs.

- Select the target instance from the **Instance** drop-down list.
- Specify **Lines** to configure how many rows you can view.
- Click **Download** to download the logs.

----End

5.17.4 Configuring Alarm Thresholds for Resource Monitoring

When a component is deployed based on CCE, if you need to monitor some resources and respond to exceptions in a timely manner, you can create threshold rules for metrics of these key resources, so that you can find and handle exceptions in time.

- If the metric meets the threshold conditions within a specified period, the system sends a threshold alarm.
- If no metric is reported within a specified period, the system sends a data insufficiency event.
- If you cannot query the change information about the threshold rule status on the ServiceStage console, you can enable the notification function to send the change information to related personnel through SMS messages or emails.

Procedure

Step 1 Log in to ServiceStage.

Step 2 Use either of the following methods to go to the **Threshold Alarms** page.

- On the **Application Management** page, click the application to which the component belongs, and click the target component in **Component List**. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu, and choose **O&M Configurations** > **Threshold Alarms** in the left navigation pane.
- On the **Component Management** page, click the target component. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu, and choose **O&M Configurations** > **Threshold Alarms** in the left navigation pane.

Step 3 Click **Set Threshold Rule** and set threshold rule parameters by referring to **Table 5-4**. Parameters marked with an asterisk (*) are mandatory.

Table 5-4 Threshold rule parameters

Parameter	Description
*Threshold Name	Name of the threshold rule to be added. **NOTE** The name must be unique and cannot be modified once specified.
Description	Description about the threshold rule.
Statistic Method	Method used to measure metrics.
Statistical Periods	Interval at which metric data is collected.
Metric	Select the metrics to be monitored.
*Threshold Condition	Trigger of a threshold alarm. A threshold condition consists of two parts: operators (≥, ≤, >, and <) and threshold value. For example, if this parameter is set to **≥ 80**, the system generates a threshold alarm when the metric is greater than or equal to 80.
Consecutive Periods	When the metric meets the threshold condition for a specified number of consecutive periods, a threshold alarm will be generated.
Alarm Severity	Severity of the threshold alarm.
Send Notifications	Whether to send notifications. • If you select **Yes** (recommended), SMN will send notifications to you when a threshold alarm is triggered. • If you select **No**, no notifications will be sent to you.
*Topic Name	If **Send Notification** is set to **Yes**, select a created topic. For details about how to create a topic, see **Creating a Topic**.
*Trigger Condition	Trigger condition for sending a notification when **Send Notification** is set to **Yes**. • **An alarm occurred**: When a threshold alarm is generated, the system sends a notification to a specified user by email or SMS message. • **The alarm is cleared**: When the alarm is cleared, the system sends a notification to a specified user by email or SMS message. • **Insufficient**: When no metric is reported, the system sends a notification to a specified user by email or SMS message.

Step 4 Click **OK**.

----End

Follow-Up Operations

After a threshold rule is created, you can manage threshold alarms by referring to **Table 5-5**.

Table 5-5 Operations related to threshold alarm management

Operation	Description
Modify a Threshold Alarm	When you find that the current threshold rule is not properly set, you can perform the following operations to modify the threshold rule to better meet your service requirements. 1. Click **Modify** in the **Operation** column of the threshold alarm list. 2. On the **Modify Threshold Rule** page, modify the parameters of the threshold rule as prompted. 3. Click **Modify**.
Delete a Threshold Alarm	When you find that the current threshold rule is no longer needed, you can perform the following operations to delete the threshold rule to release more threshold rule resources. 1. Delete one or multiple threshold rules. • To delete a single threshold, click **Delete** in the **Operation** column of the threshold rule list. • To delete threshold rules in bathes, select one or more threshold rules and click **Delete** on the upper part of the page. 2. In the displayed dialog box, click **OK**.
Search for Threshold Alarms	1. Select a time segment from the drop-down list. 2. Enter the keyword of the alarm name or description in the search box on the upper right corner of the page. 3. Click or press **Enter**. Alternatively, click **Advanced Search**, set the search criteria, and click **Search**.
View Threshold-Crossing Alarms	If the metric meets the threshold conditions within a specified period, the system sends a threshold alarm. View the alarm in the threshold alarm list.
View Alarm History	Click **History** in the **Operation** column of the threshold rule list to view historical alarms.

Operation	Description
Check the data insufficiency event.	If no metric is reported within a specified period, the system sends a data insufficiency event. You can view the event on the **Event** page. For details, see **Viewing Component Running Events**.

5.17.5 Viewing Component Running Events

If a component is deployed in the Kubernetes environment, you can view events that occur during component running to locate and rectify faults that occur during component running.

Procedure

Step 1 Log in to ServiceStage.

Step 2 Use either of the following methods to go to the **Events** page.

- On the **Application Management** page, click the application to which the component belongs, and click the target component in **Component List**. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu, and choose **O&M Configurations** > **Events** in the left navigation pane.
- On the **Component Management** page, click the target component. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu, and choose **O&M Configurations** > **Events** in the left navigation pane.

Step 3 On the **Events** page, view the component running events.

- You can select the query time to view the component running events within a specified time range.
- You can enter an event keyword to search for and view specific component running events.

----End

5.18 Viewing the Component Running Environment

After a component is successfully deployed, you can view the resources (such as CCE clusters and microservice engines) on which the component depends and the resource status and usage on the component **Infrastructure** page.

Procedure

Step 1 Log in to ServiceStage.

Step 2 Use either of the following methods to go to the **Infrastructure** page.

- On the **Application Management** page, click the application to which the component belongs, and click the target component in **Component List**.

Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu, and choose **Infrastructure** in the left navigation pane.

- On the **Component Management** page, click the target component. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu, and choose **Infrastructure** in the left navigation pane.

Step 3 On the component **Infrastructure** page, select component running resources and view the resource status and usage.

----End

5.19 Starting and Stopping a Component Instance

After a component is successfully deployed, you can restart or stop the component as required.

Procedure

Step 1 Log in to ServiceStage.

Step 2 Use either of the following methods to go to the component **Overview** page.

- On the **Application Management** page, click the application to which the component belongs, and click the target component in **Component List**. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu.
- On the **Component Management** page, click the target component. Alternatively, right-click the component and go to the component **Overview** page from the shortcut menu.

Step 3 Start or stop a component.

- Click **Stop** to stop an application component in the **Running** or **Not ready** state.
- Click **Start** to start an application component in the **Stopped** state.
- Click **Restart** to restart an application component in the **Running** or **Not ready** state.

----End

5.20 Deleting a Component

This topic describes how to delete a component that is no longer used.

NOTICE

Deleted application components cannot be restored. Exercise caution when performing this operation.

Procedure

Step 1 Log in to ServiceStage.

Step 2 On the **Application Management** page, click the application to which the component belongs.

Step 3 In **Component List**:

- Single deletion: Locate the component to be deleted and choose **More** > **Delete** in the **Operation** column.
- Batch deletion: Select the components to be deleted and click **Bulk Delete**.

Step 4 In the displayed dialog box, click **OK**.

----End

5.21 Synchronizing Component Status

ServiceStage periodically syncs the number and status of component instances from compute resources (CCE and ECS) that run component instances. To manually update and sync component status, perform the following operations:

Procedure

Step 1 Log in to ServiceStage.

Step 2 Use either of the following methods to select the target component:

- On the **Application Management** page, click the application to which the target component belongs, and click the component in **Component List**.
- On the **Component Management** page, click the target component.

Step 3 In the **Operation** column, choose **More** > **Sync** to synchronize the component information.

----End

5.22 Component Advanced Setting

5.22.1 Configuring Environment Variables of a Component

Environment variables are set in the container running environment and can be modified after component deployment, ensuring the flexibility of applications.

Environment variables set for an application component are local environment variables and take effect only for this application component.

If you add an application environment variable to the application where the component is located and the name of the application environment variable is the same as that of the component environment variable in the application, the application environment variable is shielded by the component environment variable and does not take effect for the component. For details about how to add

application environment variables, see **Managing Application Environment Variables**.

This topic describes how to configure component environment variables in different deployment modes during component deployment. For details about the component deployment mode, see **Deploying a Component**.

Deploying a Component Using CCE

During component deployment, add environment variables on the **Advanced Settings** page by referring to the following steps.

Step 1 Choose **Advanced Settings** > **Component Configuration**.

Step 2 Add environment variables by referring to **Table 5-6**.

Currently, environment variables can be added using any of the following methods:

Table 5-6 Environment variable types

Environment Variable Type	Procedure
Add manually	1. Click **Add Environment Variable** and select **Add manually**. 2. Set **Name** and **Variable/Variable Reference** to add an environment variable.
Import from a secret	1. Create a secret. For details, see **Creating a Secret**. 2. Click **Add Environment Variable** and select **Add from secret**. 3. Enter **Name**. 4. Select a secret from the **Variable/Variable Reference** drop-down list.
Import from configuration items	1. Create a configuration item. For details, see **Creating a Configuration Item**. 2. Click **Add Environment Variable** and select **Add from ConfigMap**. 3. Enter **Name**. 4. Select a configuration item from the **Variable/Variable Reference** drop-down list.
Import from a file	Click **Import** and select a local configuration file. The imported file must be a key-value pair mapping file in JSON or YAML format. For example: {"key1":"value1","key2":"value2"}

----End

Deploying a Component Using VM

During component deployment, add environment variables by referring to the following steps.

Step 1 Click **Add Environment Variable**.

Step 2 Enter **Key** and **Value**.

----End

5.22.2 Configuring the Lifecycle of a Component

For container-deployed components, ServiceStage provides callback functions for the lifecycle management of applications. For example, if you want an application component to perform a certain operation before stopping, you can register a hook function.

ServiceStage provides the following lifecycle callback functions:

- Startup command: used to start a container.
- Post-start processing: triggered after an application is started.
- Pre-stop processing: triggered before an application is stopped.

Procedure

Step 1 During component deployment, choose **Advanced Settings** > **Deployment Configuration** in the **Advanced Settings** area.

Step 2 Click **Startup Command** to set **Command** and **Parameter** for the container.

A Docker image has metadata that stores image information. If no **Lifecycle** command or parameter is set, the container runs the default command and parameter provided during image creation. The Docker defines the default command and parameter as **CMD** and **Entrypoint**. For details about the two fields, see ***Entrypoint Description*** and ***CMD Description***.

If the running command and parameter of the application are set during application component deployment, the default **Entrypoint** and **CMD** will be overwritten during image building. **Table 5-7** describes the rules.

Table 5-7 Startup command parameters

Image Entrypoint	Image CMD	Application Running Command	Application Running Parameter	Final Execution
[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]

Image Entrypoint	Image CMD	Application Running Command	Application Running Parameter	Final Execution
[touch]	[/root/test]	[mkdir]	[/opt/test]	[mkdir /opt/test]

Step 3 Click **Lifecycle** and set **Post-Start** and **Pre-Stop** parameters. Table 5-8 describes the parameters. Select one of the parameters.

Table 5-8 Container lifecycle parameters

Parameter	Description

CLI Mode	Command to be executed in the component instance. The command format is **Command** *Args[1] Args[2]...*. **Command** is a system command or a user-defined executable program. If no path is specified, an executable program in the default path will be selected. If multiple commands need to be executed, write the commands into a script for execution.
	For example, the following commands need to be executed:
	`exec:`
	`  command:`
	`  - /install.sh`
	`  - install_agent`
	Write **/install.sh install_agent** in the script.
	This command indicates that the agent will be installed after the component is deployed.

HTTP Request Mode	HTTP call request. The related parameters are described as follows: • **Path**: (optional) URL of a request. • **Port**: (mandatory) request port. • **Host Address**: (optional) IP address of the request. The default value is the IP address of the node where the application is located.

----End

5.22.3 Configuring Data Storage

Container storage is a component that provides storage for applications. Multiple types of storage are supported. An application component can use any amount of storage.

Components deployed using CCE support data storage settings.

Scenario

Table 5-9 Storage scenarios

Storage Type	Scenario
EVS Disks	EVS supports three specifications: common I/O, high I/O, and ultra-high I/O. • Common I/O: The backend storage is provided by the Serial Advanced Technology Attachment (SATA) storage media. Common I/O is applicable to scenarios where large capacity is needed but high read/write rate is not required, and the volume of transactions is low. Examples include development testing and enterprise office applications. • High I/O: The backend storage is provided by the Serial Attached SCSI (SAS) storage media. High I/O is applicable to scenarios where relatively high performance, high read/write rate, and real-time data storage are required. Examples include creating file systems and sharing distributed files. • Ultra-high I/O: The backend storage is provided by the Solid-State Drive (SSD) storage media. Ultra-high I/O is applicable to scenarios where high performance, high read/write rate, and data-intensive applications are required. Examples include NoSQL, relational database, and data warehouse (such as Oracle RAC and SAP HANA).
SFS File Systems	SFS applies to a wide range of scenarios, including media processing, content management, big data, and workload analysis.
OBS Buckets	• Standard OBS buckets: This type of OBS buckets applies to scenarios where a large number of hotspot files or small-sized files need to be accessed frequently (multiple times per month on average) and data can be quickly obtained. For example, cloud applications, data analysis, content analysis, and hotspot objects. • Infrequent access OBS buckets: This type of OBS buckets applies to scenarios where data is not frequently accessed (less than 12 times per year on average) but fast access response is required. For example, static website hosting, backup/active archiving, storage resource pools or backup storage for cloud services.
SFS Turbo	SFS Turbo file systems are fast, on-demand, and scalable, which are suitable for DevOps, containerized microservices, and enterprise office applications.

Storage Type	Scenario
HostPath	The file directory of the host where the application component is located is mounted to the specified mounting point of the application. If the application component needs to access **/etc/hosts**, use **HostPath** to map **/etc/hosts**. **NOTICE** Do not mount the file directory to a system directory such as **/** or **/var/run**. Otherwise, an exception occurs. An empty directory is recommended. If the directory is not empty, ensure that the directory does not contain any files that affect the application component instance startup. Otherwise, the file will be replaced, causing application component instance startup exceptions.
EmptyDir	Used for temporary storage. The lifecycle of temporary storage is the same as that of an application component instance. When an application instance disappears, **EmptyDir** will be deleted and the data is permanently lost.
ConfigMap	Keys in a configuration item are mapped to an application so that configuration files can be mounted to the specified application component directory.
Secret	Sensitive information such as application authentication and application keys is stored in a secret, and the secret is mounted to a specified path of the application component.

EVS Disks

Step 1 During component deployment, choose **Advanced Settings** > **Deployment Configuration** in the **Advanced Settings** area.

Step 2 Choose **Data Storage** > **Cloud Storage** > **Add Cloud Storage** and set parameters by referring to **Table 5-10**.

Table 5-10 EVS disks

Parameter	Description
Storage Type	Select **EVS disk**. The method of using an EVS disk is the same as that of using a traditional disk. However, EVS disks have higher data reliability and I/O throughput and are easier to use. They apply to file systems, databases, or other system software or workloads that require block storage devices.

Parameter	Description
Storage Allocation Mode	• **Manual** Select an EVS volume. You need to create an EVS volume in advance. For details, see **EVS Volumes**. • **Automatic** A storage is created automatically. You need to enter the storage capacity. 1. If **Storage Class** is set to **EVS Disk**, select an AZ for creating the EVS disk first. 2. Select a storage sub-type. High I/O: EVS disks that have high I/O and use SAS. Common I/O: EVS disks that use SATA. Ultra-high I/O: EVS disks that have ultra-high I/O and use SSD. 3. Enter the storage capacity, in the unit of GB. Ensure that the storage capacity quota is not exceeded; otherwise, creation will fail.
Add Docker Mounting	1. Set **Sub-path** and **Container Path** to the path to which the data volume is mounted. For example, if **Container Path** is set to **/tmp** and **Sub-path** is set to **/app**, the data volume is mounted to **/tmp/app** of the application. **NOTICE** – Do not mount a data volume to a system directory such as **/** or **/var/run**. Otherwise, an exception occurs. An empty directory is recommended. If the directory is not empty, ensure that the directory does not contain any file that affects application startup. Otherwise, the files will be replaced, causing application startup exceptions. As a result, the application fails to be created. – When the data volume is mounted to a high-risk directory, you are advised to use a low-permission account to start the application; otherwise, high-risk files on the host may be damaged. 2. Set **Permission**. – Read-only: allows you only to read data volumes in the application path. – Read/Write: allows you to modify data volumes in the application path. To prevent data loss, newly written data will not be migrated during application migration.

Step 3 Click **OK**.

----End

SFS File Systems

Step 1 During component deployment, choose **Advanced Settings** > **Deployment Configuration** in the **Advanced Settings** area.

Step 2 Choose **Data Storage** > **Cloud Storage** > **Add Cloud Storage** and set parameters by referring to **Table 5-11**.

Table 5-11 SFS file systems

Parameter	Description
Storage Type	Select **SFS**. SFS applies to a wide range of scenarios, including media processing, content management, big data, and application analysis.
Storage Allocation Mode	• **Manual** Select an SFS volume. You need to create an SFS volume in advance. For details, see **SFS Volumes**. • **Automatic** A storage is created automatically. You need to enter the storage capacity. 1. Select a storage sub-type. Set the sub-type to NFS. 2. Enter the storage capacity, in the unit of GB. Ensure that the storage capacity quota is not exceeded; otherwise, creation will fail.
Add Docker Mounting	1. Set **Sub-path** and **Container Path** to the path to which the data volume is mounted. For example, if **Container Path** is set to **/tmp** and **Sub-path** is set to **/app**, the data volume is mounted to **/tmp/app** of the application. **NOTICE** – Do not mount a data volume to a system directory such as **/** or **/var/run**. Otherwise, an exception occurs. An empty directory is recommended. If the directory is not empty, ensure that the directory does not contain any file that affects application startup. Otherwise, the files will be replaced, causing application startup exceptions. As a result, the application fails to be created. – When the data volume is mounted to a high-risk directory, you are advised to use a low-permission account to start the application; otherwise, high-risk files on the host may be damaged. 2. Set **Permission**. – Read-only: allows you only to read data volumes in the application path. – Read/Write: allows you to modify data volumes in the application path. To prevent data loss, newly written data will not be migrated during application migration.

Step 3 Click **OK**.

----End

OBS Buckets

Step 1 During component deployment, choose **Advanced Settings** > **Deployment Configuration** in the **Advanced Settings** area.

Step 2 Choose **Data Storage** > **Cloud Storage** > **Add Cloud Storage** and set parameters by referring to **Table 5-12**.

Table 5-12 OBS buckets

Parameter	Description
Storage Type	Select **OBS**. Standard and Infrequent Access OBS classes are supported. OBS buckets apply to scenarios such as big data analytics, cloud native application data, static website hosting, and backup/active archiving.
Storage Allocation Mode	• **Manual** Select an OBS volume. You need to create an OBS volume in advance. For details, see **OBS Volumes**. • **Automatic** 1. Set **Secret**. **Namespace** is the namespace of the container where the component instance is deployed when **creating and deploying a component**. It cannot be changed. Click **Use Existing Secret** to select the secret in the namespace of the container where the component instance is located. You can also create a secret: Enter a new secret name, click **Add Key File**, and upload the obtained local secret file. For details about how to obtain the secret file, see **Access Keys**. 2. Select a storage sub-type. You can select **Standard** or **Infrequent Access**.

Parameter	Description
Add Docker Mounting	1. Set **Sub-path** and **Container Path** to the path to which the data volume is mounted. For example, if **Container Path** is set to **/tmp** and **Sub-path** is set to **/app**, the data volume is mounted to **/tmp/app** of the application. **NOTICE** – Do not mount a data volume to a system directory such as **/** or **/var/run**. Otherwise, an exception occurs. An empty directory is recommended. If the directory is not empty, ensure that the directory does not contain any file that affects application startup. Otherwise, the files will be replaced, causing application startup exceptions. As a result, the application fails to be created. – When the data volume is mounted to a high-risk directory, you are advised to use a low-permission account to start the application; otherwise, high-risk files on the host may be damaged. 2. Set **Permission**. – Read-only: allows you only to read data volumes in the application path. – Read/Write: allows you to modify data volumes in the application path. To prevent data loss, newly written data will not be migrated during application migration.

Step 3 Click **OK**.

----End

SFS Turbo

Step 1 During component deployment, choose **Advanced Settings** > **Deployment Configuration** in the **Advanced Settings** area.

Step 2 Choose **Data Storage** > **Cloud Storage** > **Add Cloud Storage** and set parameters by referring to **Table 5-13**.

Table 5-13 SFS Turbo

Parameter	Description
Storage Type	Select **SFS Turbo**. Standard and Infrequent Access OBS buckets are supported. OBS buckets apply to scenarios such as big data analytics, cloud native application data, static website hosting, and backup/active archiving.

Parameter	Description
Storage Allocation Mode	• **Manual** Select an SFS Turbo volume. You need to create an SFS Turbo volume in advance. For details, see **SFS Turbo**. • **Automatic** Select an SFS Turbo file system. You need to create an SFS Turbo file system in advance. For details, see **Creating an SFS Turbo File System**.
Add Docker Mounting	1. Set **Sub-path** and **Container Path** to the path to which the data volume is mounted. For example, if **Container Path** is set to **/tmp** and **Sub-path** is set to **/app**, the data volume is mounted to **/tmp/app** of the application. **NOTICE** – Do not mount a data volume to a system directory such as **/** or **/var/run**. Otherwise, an exception occurs. An empty directory is recommended. If the directory is not empty, ensure that the directory does not contain any file that affects application startup. Otherwise, the files will be replaced, causing application startup exceptions. As a result, the application fails to be created. – When the data volume is mounted to a high-risk directory, you are advised to use a low-permission account to start the application; otherwise, high-risk files on the host may be damaged. 2. Set **Permission**. – Read-only: allows you only to read data volumes in the application path. – Read/Write: allows you to modify data volumes in the application path. To prevent data loss, newly written data will not be migrated during application migration.

Step 3 Click **OK**.

----End

HostPath

The file or directory of the host is mounted to the component.

Step 1 During component deployment, choose **Advanced Settings** > **Deployment Configuration** in the **Advanced Settings** area.

Step 2 Choose **Data Storage** > **Local Disk** > **Add Local Disk** and set parameters by referring to **Table 5-14**.

Table 5-14 HostPath

Parameter	Description
Local Disk Type	Select **HostPath**.
Host Path	Enter the host path, for example, **/etc/hosts**.
Docker Mounting	1. Set **Sub-path** and **Container Path** to the path to which the data volume is mounted. For example, if **Container Path** is set to **/tmp** and **Sub-path** is set to **/app**, the data volume is mounted to **/tmp/app** of the application. **NOTICE** – Do not mount a data volume to a system directory such as **/** or **/var/run**. Otherwise, an exception occurs. An empty directory is recommended. If the directory is not empty, ensure that the directory does not contain any file that affects application startup. Otherwise, the files will be replaced, causing application startup exceptions. As a result, the application fails to be created. – When the data volume is mounted to a high-risk directory, you are advised to use a low-permission account to start the application; otherwise, high-risk files on the host may be damaged. 2. Set **Permission**. – Read-only: allows you only to read data volumes in the application path. – Read/Write: allows you to modify data volumes in the application path. To prevent data loss, newly written data will not be migrated during application migration.

Step 3 Click **OK**.

----End

EmptyDir

EmptyDir applies to temporary data storage, disaster recovery, and shared running. It will be deleted upon deletion or transfer of application component instances.

Step 1 During component deployment, choose **Advanced Settings** > **Deployment Configuration** in the **Advanced Settings** area.

Step 2 Choose **Data Storage** > **Local Disk** > **Add Local Disk** and set parameters by referring to **Table 5-15**.

Table 5-15 EmptyDir

Parameter	Description
Local Disk Type	Select **EmptyDir**.

Parameter	Description
Disk Media	• If you select **Memory**, the running speed is improved, but the storage capacity is limited by the memory size. This mode applies to a small amount of data with high requirements on reading and writing efficiency. • If **Memory** is not selected, data is stored in disks, which is applicable to a large amount of data with low requirements on reading and writing efficiency.
Docker Mounting	1. Set **Sub-path** and **Container Path** to the path to which the data volume is mounted. For example, if **Container Path** is set to **/tmp** and **Sub-path** is set to **/app**, the data volume is mounted to **/tmp/app** of the application. **NOTICE** – Do not mount a data volume to a system directory such as **/** or **/var/run**. Otherwise, an exception occurs. An empty directory is recommended. If the directory is not empty, ensure that the directory does not contain any file that affects application startup. Otherwise, the files will be replaced, causing application startup exceptions. As a result, the application fails to be created. – When the data volume is mounted to a high-risk directory, you are advised to use a low-permission account to start the application; otherwise, high-risk files on the host may be damaged. 2. Set **Permission**. – Read-only: allows you only to read data volumes in the application path. – Read/Write: allows you to modify data volumes in the application path. To prevent data loss, newly written data will not be migrated during application migration.

Step 3 Click **OK**.

----End

ConfigMap

ServiceStage separates the application codes from configuration files. **ConfigMap** is used to process application component configuration parameters.

Step 1 During component deployment, choose **Advanced Settings** > **Deployment Configuration** in the **Advanced Settings** area.

Step 2 Choose **Data Storage** > **Local Disk** > **Add Local Disk** and set parameters by referring to **Table 5-16**.

Table 5-16 ConfigMap

Parameter	Description
Local Disk Type	Select **ConfigMap**.
Configuration Item	Select the desired configuration item name. Create a configuration item. For details, see **Creating a Configuration Item**.
Docker Mounting	Set **Sub-path** and **Container Path** to the path to which the data volume is mounted. For example, if **Container Path** is set to **/tmp** and **Sub-path** is set to **/app**, the data volume is mounted to **/tmp/app** of the application. When you select **ConfigMap**, only **Read-only** is supported. You can only read the data volume in the container path. **NOTICE** • Do not mount a data volume to a system directory such as **/** or **/var/run**. Otherwise, an exception occurs. An empty directory is recommended. If the directory is not empty, ensure that the directory does not contain any file that affects application startup. Otherwise, the files will be replaced, causing application startup exceptions. As a result, the application fails to be created. • When the data volume is mounted to a high-risk directory, you are advised to use a low-permission account to start the application; otherwise, high-risk files on the host may be damaged.

Step 3 Click **OK**.

----End

Secret

The data in the secret is mounted to the specified application component. The content of the secret is user-defined.

Step 1 During component deployment, choose **Advanced Settings** > **Deployment Configuration** in the **Advanced Settings** area.

Step 2 Choose **Data Storage** > **Local Disk** > **Add Local Disk** and set parameters by referring to **Table 5-17**.

Table 5-17 Secret

Parameter	Description
Local Disk Type	Select **Secret**.
Secret Item	Select the desired secret name. For details about how to create a secret, see **Creating a Secret**.

Parameter	Description
Docker Mounting	Set **Sub-path** and **Container Path** to the path to which the data volume is mounted. For example, if **Container Path** is set to **/tmp** and **Sub-path** is set to **/app**, the data volume is mounted to **/tmp/app** of the application. When you select **Secret**, only **Read-only** is supported. You can only read the data volume in the container path. **NOTICE** • Do not mount a data volume to a system directory such as **/** or **/var/run**. Otherwise, an exception occurs. An empty directory is recommended. If the directory is not empty, ensure that the directory does not contain any file that affects application startup. Otherwise, the files will be replaced, causing application startup exceptions. As a result, the application fails to be created. • When the data volume is mounted to a high-risk directory, you are advised to use a low-permission account to start the application; otherwise, high-risk files on the host may be damaged.

Step 3 Click **OK**.

----End

5.22.4 Configuring Distributed Cache Service

Traditional single-instance applications use local session management. Session contexts generated by user requests are stored in the process memory. After the load balancing module is added, multi-instance sessions need to be shared using distributed storage.

ServiceStage provides the out-of-the-box distributed session function. It uses the **Distributed Cache Service** as the session persistence layer. Without code modification, ServiceStage supports distributed session management for Tomcat applications, Node.js applications that use express-session, and PHP applications that use session handle.

During component deployment, you can bind the distributed cache when configuring **Advanced Settings**. After binding, you can read environment variables upon application running to obtain information about the distributed cache. For details, see **Common Environment Variables**.

Prerequisites

A distributed cache has been created. For details, see **Buying a DCS Redis Instance**.

Procedure

Step 1 Choose **Advanced Settings** > **Distributed Cache**.

Step 2 Click **Bind Distributed Cache**.

Step 3 Select a distributed cache instance that has been bound in the environment.

If no distributed cache instance is bound to the environment, click **Add One**. On the **Edit Environment** page that is displayed, click **Add Optional Resource** to add created DCS resources to the environment.

Step 4 If the DCS instance must be accessed using a password, enter the access password.

Step 5 Click **OK**.

----End

5.22.5 Configuring Relational Databases

To store application data permanently, you need to use Relational Database Service (RDS). Based on the cloud computing platform, ServiceStage provides RDS for MySQL which is reliable, scalable, easy to manage, and ready for use. **RDS for MySQL** enables you to easily set and scale relational databases on the cloud. Using the RDS service, you can perform nearly all necessary tasks without programming. This service simplifies operation procedures and reduces routine O&M workloads, so that you can focus on application and service development.

During component deployment, you can bind relational databases in **Database**. The procedure is as follows: After binding, you can read environment variables upon application running to obtain MySQL information. For details, see **Common Environment Variables**.

Prerequisites

An RDS MySQL DB instance has been created. For details, see **Step 1: Buy a DB Instance**.

Procedure

Step 1 Choose **Advanced Settings** > **Cloud Database**.

Step 2 Click **Bind Cloud Database**.

Step 3 Select a cloud database instance that has been bound in the environment and click **OK**.

If no cloud database instance is bound to the environment, click **Add One**. On the **Edit Environment** page that is displayed, click **Add Optional Resource** to add created RDS resources to the environment.

Step 4 Set the parameters by referring to the following table. Parameters marked with an asterisk (*) are mandatory.

Parameter	Description
*Connection Type	Select a connection type. • **JNDI**: standard Java connection mode. You need to enter the JNDI name. • **Spring Cloud Connector**: Spring connection mode.

Parameter	Description
*Database Name	Enter a database name.
*Database Account	Enter a database account.
*Database Password	Enter the database password.

Step 5 Click **OK**.

----End

5.22.6 Configuring a Scheduling Policy of a Component Instance

Based on features of components deployed using CCE, ServiceStage divides application components into the minimum deployment instances. The application scheduler monitors application instance information in real time. When detecting that a new pod needs to be scheduled, the application scheduler calculates all remaining resources (compute, network resources, and middleware) in the cluster to obtain the most appropriate scheduling target node.

ServiceStage supports multiple scheduling algorithms, including affinity scheduling between applications and AZs, between applications and nodes, and between applications.

You can freely combine these policies to meet your requirements.

Affinity

If an application is not containerized, multiple components of the application may run on the same virtual machine, and processes communicate with each other.

However, during containerization splitting, containers are usually split by process. For example, service processes are stored in a container, monitoring log processing or local data is stored in another container, and there is an independent life cycle. If closely related container processes run on distant nodes, routing between them will be costly and slow.

Affinity: Containers are scheduled onto the nearest node. This makes routing paths between containers as short as possible, which in turn reduces network overhead.

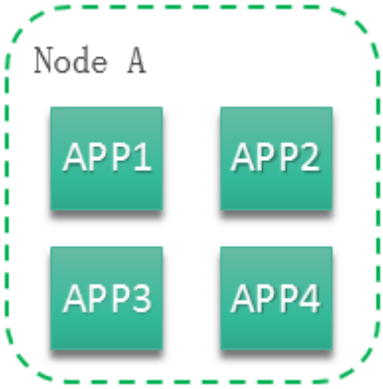
Anti-affinity: Instances of the same application are spread across different nodes to achieve higher availability. Once a node is down, instances on other nodes are not affected.

- Application-AZ Affinity and Anti-Affinity
  - **Affinity with AZs**: Application components can be deployed in specific AZs.
  - **Non-affinity with AZs**: Application components cannot be deployed in specific AZs.

- Application-Node Affinity and Anti-Affinity
  - **Affinity with Nodes**: Application components can be deployed on specific nodes.
  - **Non-affinity with Nodes**: Application components cannot be deployed on specific nodes.
- Application Affinity

  It determines whether application components are deployed on the same node or different nodes.
  - **Affinity with Applications**: Application components are deployed on the same node. You can deploy application components based on service requirements. The nearest route between application components is used to reduce network consumption. For example, **Figure 5-17** shows affinity deployment, in which all applications are deployed on the same node.

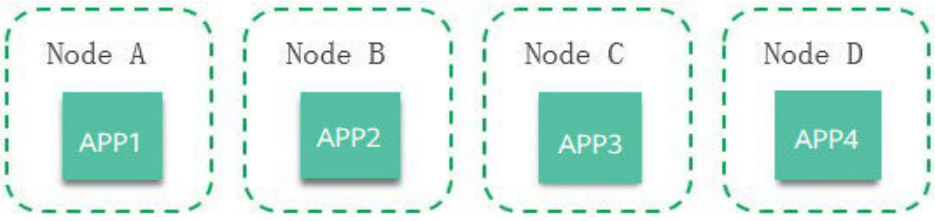
    Figure 5-17 Application affinity

  - **Anti-affinity with Applications**: Different applications or multiple instances of the same application component are deployed on different nodes. Anti-affinity deployment for multiple instances of the same application reduces the impact of system breakdowns. Anti-affinity deployment for applications can prevent interference between the applications.

    As shown in **Figure 5-18**, four applications are deployed on four different nodes. The four applications are deployed in anti-affinity mode.

    Figure 5-18 Application anti-affinity

Precautions

When setting application component-node affinity and application component-application component affinity, ensure that the affinity relationships are not

mutually exclusive; otherwise, application deployment will fail. For example, application deployment will fail when the following conditions are met:

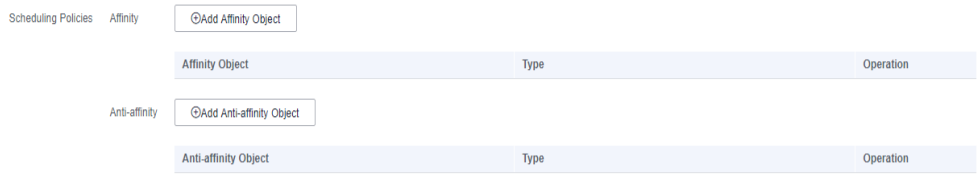
- Anti-affinity is configured for two application components APP 1 and APP 2. For example, APP 1 is deployed on node A and APP 2 is deployed on node B.
- When APP 3 is deployed on node C and goes online, affinity is configured between APP 3 and APP 2. As a result, affinity relationships are mutually exclusive, and APP 3 fails to be deployed.

Procedure

Step 1 Choose **Advanced Settings** > **Deployment Configuration**.

Step 2 On the **Scheduling Policy** tab, configure the component instance scheduling policy by referring to the following table.

Figure 5-19 Configuring a scheduling policy on the component configuration page



Purpose	Procedure
Setting application component-AZ affinity	1. Click **Add Affinity Object**. 2. Set the object type to **AZ**, and select the desired AZ. 3. Click **OK**.
Setting application component-AZ anti-affinity	1. Click **Add Anti-affinity Object**. 2. Set the object type to **AZ**, and select the desired AZ. 3. Click **OK**.
Setting application component-node affinity	1. Click **Add Affinity Object**. 2. Set the object type to **Node**, and select the desired node. 3. Click **OK**.
Setting application component-node non-affinity	1. Click **Add Anti-affinity Object**. 2. Set the object type to **Node**, and select the desired node. 3. Click **OK**.

Purpose	Procedure
Setting application component-application component affinity	1. Click **Add Affinity Object**. 2. Set the object type to **Component**, and select the desired application components. 3. Click **OK**. The selected application components are deployed on the same node.
Setting application component-application component anti-affinity	1. Click **Add Anti-affinity Object**. 2. Set the object type to **Component**, and select the desired application components. 3. Click **OK**. The selected application components are deployed on different nodes.

----End

5.22.7 Configuring a Log Policy of an Application

ServiceStage allows you to configure application log policies for application components deployed in containers. You can view related logs on the AOM console.

You can configure log policies during component deployment. If no configuration is performed, the system collects standard application output logs by default.

Procedure

Step 1 Choose **Advanced Settings** > **O&M Monitoring**.

Step 2 On the **Log Collection** tab, click **Add Log Policy** and set the parameters by referring to the following table. Parameters marked with an asterisk (*) are mandatory.

Parameter	Description
Storage Type	Select a storage type. • **HostPath**: Mount a host path to a specified container path. • **Mounting Path**: Logs are exported only to the container path. You do not need to mount the host path.
***Host Path**	This parameter is mandatory when **Storage Type** is set to **HostPath**. Enter the log storage path on the host.

Parameter	Description
***Docker Mounting**	1. Set **Mounting Path**: Enter the application path to which the data volume is mounted. **NOTICE** – Do not mount a data volume to a system directory such as **/** or **/var/run**. Otherwise, an exception occurs. An empty directory is recommended. If the directory is not empty, ensure that the directory does not contain any file that affects application startup. Otherwise, the files will be replaced, causing application startup exceptions. As a result, the application fails to be created. – When the data volume is mounted to a high-risk directory, you are advised to use a low-permission account to start the application; otherwise, high-risk files on the host may be damaged. 2. Set **Extended Host Path**. – **None**: No extended path is configured. – **PodUID**: Pod ID. – **PodName**: Pod name. – **PodUID/ContainerName**: Pod ID or container name. – **PodName/ContainerName**: Pod name or container name. 3. Set **Aging Period**. – **Hourly**: Log files are scanned every hour. If the size of a log file exceeds 20 MB, the system compresses the log file to a historical file, dumps the historical file to the directory where the log file is stored, and clears the original log file. – **Daily**: Log files are scanned once a day. If the size of a log file exceeds 20 MB, the system compresses the log file to a historical file, dumps the historical file to the directory where the log file is stored, and clears the original log file. – **Weekly**: Log files are scanned once a week. If the size of a log file exceeds 20 MB, the system compresses the log file to a historical file, dumps the historical file to the directory where the log file is stored, and clears the original log file.

NOTE

After the component configuration and deployment are complete, you can view run logs on the AOM console. For details, see **Viewing Log Files**.

----End

5.22.8 Configuring Custom Monitoring of a Component

ServiceStage allows you to obtain custom metrics when components are deployed using CCE. You can use this method to report custom component running metrics.

Precautions

- Currently, only **Gauge metrics** of Prometheus can be obtained.
- Before setting custom metric monitoring for an application component, you must understand **Prometheus** and provide the GET API for obtaining custom metric data in your application component so that ServiceStage can obtain custom metric data using this API.

Procedure

Step 1 Choose **Advanced Settings** > **O&M Monitoring**.

Step 2 On the **O&M Policy** tab, configure the custom monitoring by referring to the following table.

Parameter	Description	Mandatory
Report Path	URL provided by the exporter for ServiceStage to obtain custom metric data. Example: **/metrics**	Yes
Report Port	Port provided by the exporter for ServiceStage to obtain custom metric data. Example: **8080**	Yes
Monitoring Metrics	Name of the custom metric provided by the exporter. Example: **["cpu_usage","mem_usage"]** • If this parameter is not set, ServiceStage collects data of all custom metrics. • If you set this parameter, for example, to **["cpu_usage","mem_usage"]**, ServiceStage collects the data of the specified cpu_usage and mem_usage metrics.	No

NOTE

After the configuration and deployment are complete, you can view the monitoring metric data on the AOM console. For details, see **Metric Monitoring**.

----End

5.22.9 Configuring Application Performance Management

Performance management helps you quickly locate problems and identify performance bottlenecks to improve your experience. Selecting Java probe will start performance management and install Java probes on the nodes deployed with performance management, which consumes a small amount of resources. Java probes use the bytecode enhancement technology to trace Java application calls and generate topology and call chain data.

ServiceStage allows you to configure performance management during component deployment (using CCE).

Precautions

- This function can be enabled only when APM of the corresponding version is deployed and enabled in the environment.
- JDK 8, 11, and 17 are supported.
- Supported Tomcat versions: 8.x. For details, see **Usage Restrictions**.
- Performance management can be configured for Java applications (technology stack type being Java, Tomcat, or Java-based Docker) deployed using CCE.

Procedure

Step 1 Choose **Advanced Settings** > **O&M Monitoring**.

Step 2 On the **Performance Management** tab, configure performance management as follows:

Select **Java probe** and set probe parameters.

- If the probe type is **APM 1.0**, select a probe version from the drop-down list.
- If the probe type is **APM 2.0**, set the following parameters:
  - **Application**: Select the application that has the same name as the application where the application component is located from the drop-down list. If the application does not exist, click **Create Application** to create it. After the application is created, you can log in to the APM console and view the new application in the application list.
  - **Probe Version**: Select a probe version from the drop-down list.

    NOTE

    To use the probe of the latest version, select **latest**.
  - **Upgrade Policy**: The following upgrade policies are supported. By default, **Automatic upgrade upon restart** is used.
    - **Automatic upgrade upon restart**: The system downloads the probe image each time the pod is restarted.
    - **Manual upgrade**: This policy means that if a local image is available, the local image will be used. The system downloads the probe image only when a local image is unavailable.
  - **Access Key**: The access key is automatically obtained. If the access key cannot be automatically obtained, manually enter it.

----End

5.22.10 Configuring Health Check

Health check periodically checks health status during component running according to your needs.

ServiceStage provides the following health check methods:

- **Component Liveness Probe**: checks whether an application component exists. It is similar to the **ps** command that checks whether a process exists. If the liveness check of an application component fails, the cluster restarts the

application component. If the liveness check is successful, no operation is executed.

- **Component Service Probe**: checks whether an application component is ready to process user requests. It may take a long time for some applications to start before they can provide services. This is because that they need to load disk data or rely on startup of an external module. In this case, the application process exists, but the application cannot provide services. This check method is useful in this scenario. If the application component readiness check fails, the cluster masks all requests sent to the application component. If the application component readiness check is successful, the application component can be accessed.

Health Check Modes

- HTTP request-based check

  This health check mode is applicable to application components that provide HTTP/HTTPS services. The cluster periodically sends an HTTP/HTTPS GET request to such application components. If the return code of the HTTP/HTTPS response is within 200–399, the check is successful. Otherwise, the check fails. In this health check mode, you must specify an application listening port and an HTTP/HTTPS request path.

  For example, if the application component provides the HTTP service, the port number is 80, the HTTP check path is **/health-check**, and the host address is **containerIP**, the cluster periodically initiates the following request to the application:

  ```
  GET http://containerIP:80/health-check
  ```

  NOTE

  If the host address is not set, the instance IP address is used by default.

- TCP port-based check

  For applications that provide a TCP communication service, the cluster periodically establishes a TCP connection to the application. If the connection is successful, the probe is successful. Otherwise, the probe fails. In this health check mode, you must specify an application listening port. For example, if you have a Nginx application component with service port 80, after you configure a TCP port-based check for the application component and specify port 80 for the check, the cluster periodically establishes a TCP connection with port 80 of the application component. If the connection is successful, the check is successful. Otherwise, the check fails.

- CLI-based check

  In this mode, you must specify an executable command in an application component. The cluster will periodically execute the command in the application component. If the command output is **0**, the health check is successful. Otherwise, the health check fails.

  The CLI mode can be used to replace the following modes:

  – TCP port-based check: Write a program script to connect to an application component port. If the connection is successful, the script returns **0**. Otherwise, the script returns **–1**.

  – HTTP request-based check: Write a program script to run the **wget** command for an application component.

wget http://127.0.0.1:80/health-check

Check the return code of the response. If the return code is within 200–399, the script returns **0**. Otherwise, the script returns **–1**.

NOTICE

- Put the program to be executed in the application component image so that the program can be executed.
- If the command to be executed is a shell script, add a script interpreter instead of specifying the script as the command. For example, if the script is **/data/scripts/health_check.sh**, you must specify **sh/data/scripts/health_check.sh** for command execution. The reason is that the cluster is not in the terminal environment when executing programs in an application component.

Common Parameter Description

Table 5-18 Common parameter description

Parameter	Description
Latency (s)	Check delay time. Unit: second. Set this parameter according to the normal startup time of services. For example, if this parameter is set to 30, the health check will be started 30 seconds after the application starts. The time is reserved for containerized services to start.
Timeout Period (s)	Timeout duration. Unit: second. If the time exceeds this value, the health check fails. For example, setting this parameter to 10 indicates that the health check timeout period is 10s. If the parameter is left blank or set to **0**, the default timeout time is 1s.

Procedure

Step 1 Choose **Advanced Settings** > **O&M Monitoring**.

Step 2 Click **Health Check**, and set health check parameters based on service requirements.

For details about common parameters, see **Table 5-18**.

----End

6 Deployment Source Management

6.1 Software Center

6.1.1 Managing Software Packages

To upload a software package to a new SWR software repository, you can create an SWR software repository after selecting an organization during software package creation.

NOTICE

- The SWR software repository does not scan or verify the security of the uploaded software packages. To avoid privacy leakage, do not include privacy information such as unencrypted passwords in uploaded software packages. When downloading public software packages, ensure that they are from trusted repositories and prevent malicious software from being downloaded.
- If a disk is full, software packages cannot be uploaded to the repository and error information is displayed, but services are not affected. To prevent services such as logs from occupying the entire disk, you are advised to attach an independent disk to the repository.

Creating a Software Package

Step 1 Log in to ServiceStage. Choose **Deployment Source Management** > **Software Center**, and click **Create Package**.

Step 2 Configure the software package by referring to the following table. Parameters marked with an asterisk (*) are mandatory.

Table 6-1 Parameter description

Parameter	Description
*Software Center	Select an organization and a software repository. To create a software repository: 1. Click **Create Repository** and enter a new software repository name. 2. Click ✓.
***Repository Type**	Type of the software repository. The default value is **Private**. • **Private**: only for the current tenant and users under the current tenant. • **Public**: for all tenants and users.
***Name**	Software package name, which must be unique in an organization.
***Version**	Software package version. Multiple software package versions can be uploaded.
Package Description	Description of the software package.
Version Description	Description of the software package version.
Upload Software package	• **Upload now**: Upload the software package by referring to **Step 3** in **Uploading the Software Package**. • **Upload later**: After the software package is created, upload it by referring to **Uploading the Software Package**.

Step 3 Click **OK**.

----End

Uploading the Software Package

NOTICE

A maximum of 10 files can be uploaded at a time. The size of a single file (including the decompressed files) cannot exceed 2 GB.

Step 1 Log in to ServiceStage and choose **Deployment Source Management** > **Software Center**.

Step 2 Select an organization from the drop-down list on the right of **Org Management**.

Step 3 Click **Upload** next to the target software package.

1. Click **Select File**, select the target software package, and click **Open**. Alternatively, drag the target software package to the page.

2. Set the parameters in the following table. All the parameters are optional.

Table 6-2 Parameter description

Parameter	Description
Cover	If you select this option, the software package with the same name in the same path will be overwritten.
Package Path	Enter a path to store the software package. The path is the virtual path of the software repository. By default, the root directory is used. By setting the path, you can easily view and manage the software package.

Repeat the preceding steps to upload other software packages.

3. After the software package is selected:
   - Select a software file from the list of software files to be uploaded and click **Upload** in the **Operation** column to upload the specified software file.
   - In the upper part of the list of software to be uploaded, click **Upload** to upload software files in batches.

----End

Editing a Software Package

Step 1 Log in to ServiceStage and choose **Deployment Source Management** > **Software Center**.

Step 2 Select an organization from the drop-down list on the right of **Org Management**.

Step 3 Click the target software package to enter the details page.

Step 4 Click **Edit** in the upper-right corner and set the following parameters:

- **Sharing Type**: Set the type of the software repository. **Private**: only for the current tenant and users under the current tenant. **Public**: for all tenants and users.
- **Package Description**: Enter the description of the software package.

Step 5 Click **OK**.

----End

Querying the Address of a Software Package

Step 1 Log in to ServiceStage and choose **Deployment Source Management** > **Software Center**.

Step 2 Select an organization from the drop-down list on the right of **Org Management**.

Step 3 Click the target software package to enter the details page.

Step 4 In the version list, click before the target version to view the software package address.

Click to copy **Intranet address** or **External address**.

NOTE

In the row where a version file is located:

- Click **Download** to download the file.
- Click **Delete** to delete the file.

----End

Deleting a Software Package

Step 1 Log in to ServiceStage and choose **Deployment Source Management** > **Software Center**.

Step 2 Select an organization from the drop-down list on the right of **Org Management**.

Step 3 Click **Delete** on the right side of the target software package, and delete it as prompted.

NOTE

Before deleting a software package, ensure that all versions in the software package are deleted. For details, see **Deleting a Software Package Version**.

----End

Deleting a Software Package Version

Step 1 Log in to ServiceStage and choose **Deployment Source Management** > **Software Center**.

Step 2 Select an organization from the drop-down list on the right of **Org Management**.

Step 3 Click the target software package to enter the details page. In the version list:

- Deleting a single software package version

  Choose **More** > **Delete** in the **Operation** column of the target version, and delete it as prompted.
- Deleting software package versions in batches

  Select the target versions, click **Delete** above the version list, and delete the software package versions as prompted.

----End

6.1.2 Packaging Specifications of Software Packages

JAR and WAR packages can be directly uploaded.

For other types of software packages, such as ZIP packages:

The software package name must be in the following format: software name +suffix. The suffix must be .tar.gz, .tar, or .zip.

> **NOTE**
>
> The extension must be consistent with the package compression mode. Otherwise, the software package cannot be decompressed.

Directory Structure

For decompressed software packages, ensure that lifecycle command scripts can be normally executed.

The following software package directory structure is recommended:

```
|- bin
   |- xxx.tar.gz
   |- xxx.bin
|- scripts
   |- install.sh
   |- start.sh
...
```

> **NOTE**
>
> Currently, you are advised not to store decompressed software packages in the top-level directory. Otherwise, when you need to modify lifecycle execution commands, you have to use the top-level directory name to find the corresponding scripts.

Table 6-3 Description of the software package directory

Directory	Description
bin	Stores execution information about software packages, such as executable bin files and dependent compressed packages.
scripts	Stores lifecycle scripts. When creating an application, you can specify execution commands based on the location of lifecycle scripts. For example, specify **bash scripts/install.sh** in the install phase to run the installation script. Lifecycles supported by software package applications are as follows: • **Install**: Command for installing software. • **PostStart**: Operation performed after software is started. • **Start**: Command for starting software. • **Restart**: Command for restarting software, which is used to recover the applications failing in health check. • **PreStop**: Operation which is performed before software is stopped. • **Stop**: Command for stopping software. • **Update**: Command for upgrading software. • **Uninstall**: Command for uninstalling software.

6.2 Image Repository

6.2.1 Uploading an Image

After an organization is created, you can upload an image to it through the page or client.

- **Uploading an Image Through the Page**: Upload an image to SWR through the page.
- **Uploading an Image Through the Client**: Upload an image to an image repository of SWR by running commands on the client.

Container repositories are used to easily store, deploy, and manage Docker images.

Prerequisites

- An organization has been created. For details, see **Creating an Organization**.
- The image has been saved as a .tar or .tar.gz file. For details, see **Creating an Image Package**.
- The image package is created using Docker 1.11.2 or later.
- If the image is uploaded through a client, the version of the container engine client to which the image is uploaded must be 1.11.2 or later.

Uploading an Image Through the Page

NOTE

- A maximum of 10 files can be uploaded at a time. The size of a single file (including the decompressed files) cannot exceed 2 GB.
- The file name should be a string starting with a letter or digit, containing 255 characters at most, and including letters, digits, underscores (_), and hyphens (-).

Step 1 Log in to ServiceStage.

Step 2 Choose **Deployment Source Management** > **Image Repository**.

Step 3 On the **Image Repository** page, click **Upload Through SWR**.

Step 4 In the displayed dialog box, specify **Organization** to which the image is to be uploaded, click **Select File**, and select the target image file.

NOTE

If you select multiple images to upload, the system uploads them one by one. Concurrent upload is not supported.

Step 5 In the displayed dialog box, click **Start Upload**.

If **Upload completed** is displayed, the image is successfully uploaded.

NOTE

If the image fails to be uploaded, the possible causes are as follows:

- The network is abnormal. In this case, check network connectivity.
- The HTTPS certificate has errors. Press **F12** to copy the URL that fails to be requested to the address bar of the browser, open the URL again, agree to continue the access, and return to the upload page to upload the certificate again.

----End

Uploading an Image Through the Client

NOTE

If you use the client to upload an image, each image layer cannot exceed 10 GB.

Step 1 Log in to ServiceStage.

Step 2 Choose **Deployment Source Management** > **Image Repository**.

Step 3 On the **Image Repository** page, click **Upload Through Client**.

Step 4 Upload the image as prompted.

----End

6.2.2 Managing Images

Obtaining an Image Pull Address

Step 1 Log in to ServiceStage.

Step 2 Choose **Deployment Source Management** > **Image Repository** > **My Images**.

Step 3 Select an organization from the drop-down list on the right of **Org Management**.

Step 4 In the image repository list, click an image repository name to go to the details page.

Step 5 Click the **Image Tags** tab and obtain the command for pulling an image.

Click on the right of the command of the image version to be downloaded to copy the command.

----End

Setting Image Repository Attributes

Step 1 Log in to ServiceStage.

Step 2 Choose **Deployment Source Management** > **Image Repository** > **My Images**.

Step 3 Select an organization from the drop-down list on the right of **Org Management**.

Step 4 In the image repository list, click an image repository name to go to the details page.

Step 5 Click **Edit** in the upper-right corner. In the displayed dialog box, perform the following operations:

- Set **Sharing Type** to **Public** or **Private**.

  NOTE

  Public images can be downloaded and used by all users.

  - If your node and the image repository are in the same region, you can access the image repository over private networks.
  - If your node and the image repository are in different regions, the node must have access to public networks to pull images from the image repository.

- Set **Category** to set the repository category.
- Set **Description** to update the description of the image repository.

Step 6 Click **OK**.

----End

Sharing a Private Image

You can share your private images with other users and grant the users access permissions.

Only administrator and Identity and Access Management (IAM) users authorized to manage the private image can share the image. The users with whom you share the image only have the read permission. That is, they can only pull the image.

Step 1 Log in to ServiceStage and choose **Deployment Source Management** > **Image Repository** > **My Images**.

Step 2 Select an organization from the drop-down list on the right of **Org Management**.

Step 3 In the image repository list, click an image repository name to go to the details page.

Step 4 Click the **Sharing** tab, click **Share Image**, and set the following parameters:

1. **Share With**: Enter an account name.
2. **Valid Until**: Set the expiration date. If you want the image to be permanently accessible to the account, select **Permanently valid**.
3. **Description**: Enter the description.
4. **Permission**: Select the permission. Currently, only the **Download** permission is supported.

Step 5 Click **OK**.

- You can view all shared images in the shared image list.
- Select an account name and click **Edit** in the **Operation** column to edit the parameters of the shared image.
- Select an account name and click **Delete** in the **Operation** column to cancel sharing.

----End

Setting Automatic Image Synchronization

If image synchronization is enabled, the latest images are automatically synchronized to image repositories in other regions. Only accounts and users with administrator permissions can configure automatic image synchronization.

> **NOTE**
>
> After you configure automatic image synchronization, image updates will also be synchronized to target repositories. However, images that were pushed to repositories before automatic image synchronization was enabled will not be automatically synchronized.
>
> For details on how to synchronize images pushed before you set the automatic synchronization, see **Can Existing Images be Automatically Synchronized**.

Step 1 Log in to ServiceStage and choose **Deployment Source Management** > **Image Repository** > **My Images**.

Step 2 Select an organization from the drop-down list on the right of **Org Management**.

Step 3 In the image repository list, click an image repository name to go to the details page.

Step 4 Click **Set Image Synchronization** in the upper-right corner.

Step 5 In the displayed dialog box, click **Add**, set the following parameters, and click **OK** in the **Operation** column.

- **Target Region**: Select the target region for synchronization.
- **Target Organization**: Select the target organization for synchronization.
- **Overwrite Existing Image**: Select this option if you want to overwrite any nonidentical images that have the same name in the target organization. Deselect this option if you do not want any nonidentical images having the same name in the target organization to be overwritten and you want to receive a notification of the existence of such images.

Step 6 Click **OK**.

On the **Synchronization Records** tab of image details page, you can view the details of each synchronization task, including the start time, image tag, task status, type, duration, target region and organization, and task operator.

----End

Adding Image Permissions

To allow IAM users of your account to read, write, and manage a specific image, add the required permissions to the IAM users on the details page of this image.

Step 1 Log in to ServiceStage and choose **Deployment Source Management** > **Image Repository** > **My Images**.

Step 2 Select an organization from the drop-down list on the right of **Org Management**.

Step 3 In the image repository list, click an image repository name to go to the details page.

Step 4 Click the **Permission Management** tab, click **Add Permission**, select an IAM user, add the **Read**, **Write**, or **Manage** permission, and click **OK**.

Then, this IAM user has the corresponding permission.

----End

Deleting an Image

NOTICE

Deleted images cannot be recovered.

Step 1 Log in to ServiceStage and choose **Deployment Source Management > Image Repository > My Images**.

Step 2 Select an organization from the drop-down list on the right of **Org Management**.

Step 3 In the image repository list, click an image repository name to go to the details page.

- Deleting an image repository

  Click **Delete** in the upper-right corner of the page and delete the image repository as prompted.

- Deleting an image tag

  In the **Operation** column of the target image tag, click **Delete** to delete the image tag as prompted.

- Deleting image tags in batches

  Select the target image tags, click **Delete** above the tag list, and delete the image tags as prompted.

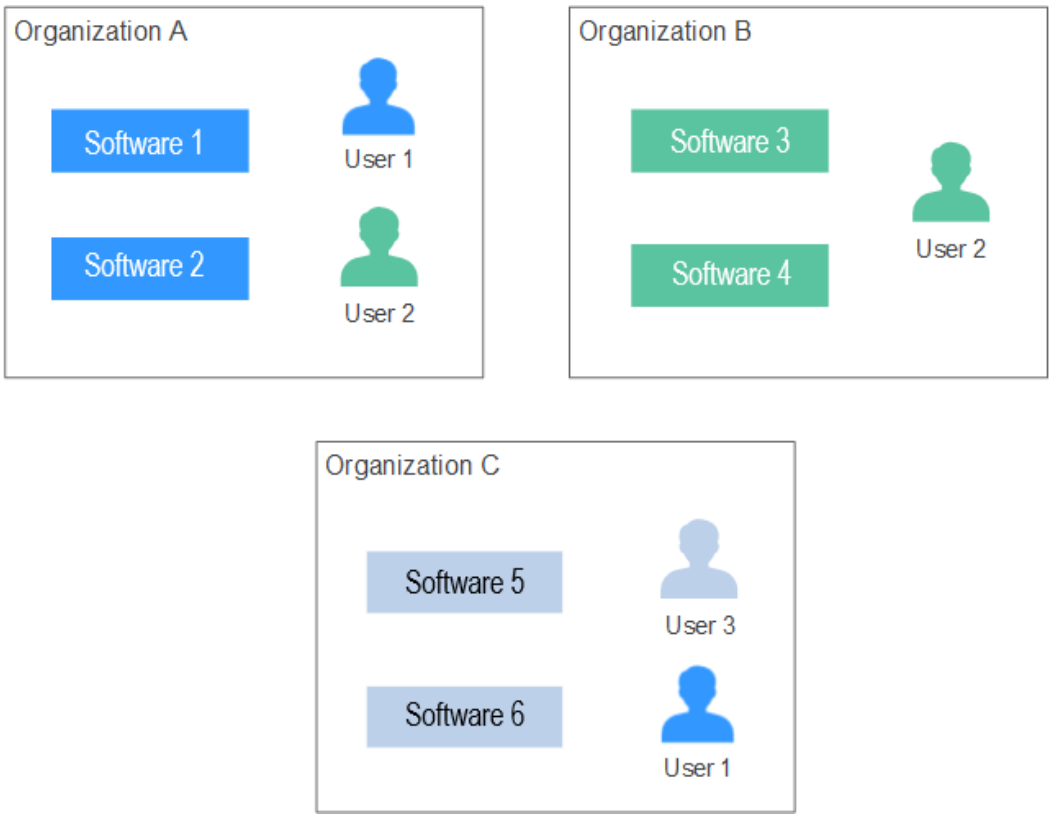
----End

6.3 Organization Management

Overview

Organizations are used to isolate software and image repositories. With each organization being limited to one company or department, software can be managed in a centralized manner. A software name only needs to be unique within an organization. The same IAM user can join different organizations. Different permissions, namely read, write, and manage, can be assigned to different IAM users in the same account.

Figure 6-1 Organization



Creating an Organization

Step 1 Log in to ServiceStage and choose **Deployment Source Management** > **Organization**.

Step 2 Click **Create Organization**, enter **Organization Name**, and click **OK**.

----End

Adding Permissions

Grant permissions to users in an organization so that they can read, edit, and manage all images in the organization.

Only users with the **Management** permission can grant permissions.

User permissions include:

- **Read-only**: Users can only download software but cannot upload software.
- **Read/write**: Users can download software, upload software, and edit software attributes.
- **Management**: Users can download and upload software, delete software or versions, edit software attributes, grant permission, and share images.

Step 1 Log in to ServiceStage and choose **Deployment Source Management** > **Organization**.

Step 2 Click **Add Permission** on the right of an organization.

Step 3 In the displayed dialog box, specify **Permission** and click **OK**.

----End

Deleting an Organization

Step 1 Log in to ServiceStage and choose **Deployment Source Management** > **Organization**.

Step 2 Click **Delete** on the right of an organization.

Before deleting an organization, delete the image and software repositories of the organization.

For details about how to delete an image repository, see **Deleting an Image**.

For details about how to delete a software repository, see **Deleting a Software Package**.

Step 3 Click **OK**.

----End

7 Continuous Delivery

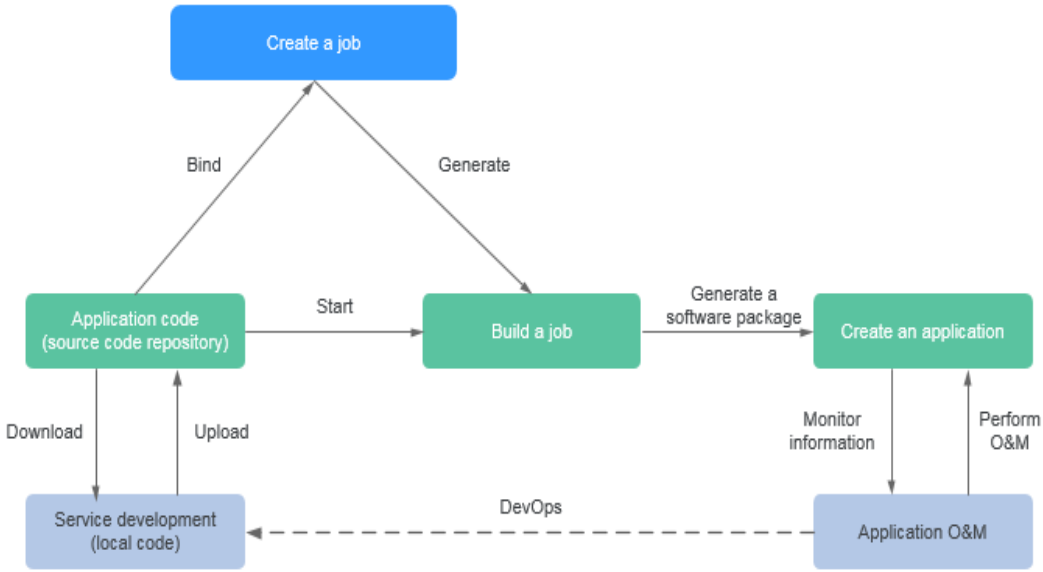
7.1 Overview

Continuous delivery provides functions such as project build and release.

Creating a Build Job

Based on the existing service code, you can create a build job, start the build job, package the service code, and archive the package to the deployment source. Then, you can use the package when deploying an application component.

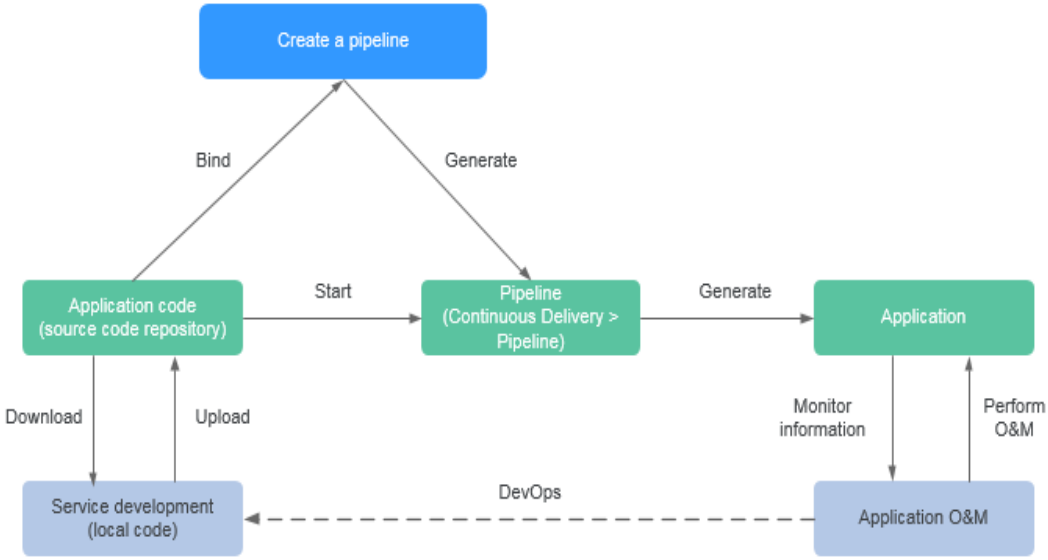
Figure 7-1 Creating a build job



Creating a Pipeline

Based on the existing service code, you can create a pipeline and then start the pipeline to complete service code building and deployment. Application O&M can also be completed on ServiceStage.

Figure 7-2 Creating a pipeline



7.2 Viewing Build Jobs

For components deployed in the Kubernetes environment, you can view the build records and logs of a specified build job in the build job list to locate and rectify faults that occur during component deployment.

Procedure

Step 1 Log in to ServiceStage.

Step 2 Choose **Continuous Delivery** > **Build**.

Step 3 On the **Build** page, use either of the following methods to search for a build job:

- Select the CCE cluster where the component is deployed and the build job status, and select the specified build job in the build list.
- Search for the build job in the search box.

Step 4 Click the build task name. The build task details page is displayed.

- View the **Basic Information** and **Build Record** of the build task.
- Click **View Log** next to a build record to view **Build Details**, **CodeCheck**, and **Log**.

  NOTE

  Only the Maven build project supports code check. Currently, the following code check plug-ins are supported: Checkstyle, FindBugs, and PMD.

----End

7.3 Creating a Source Code Job

The software package or image package can be generated with a few clicks in a build job. In this way, the entire process of source code pull, compilation, packaging, and archiving is automatically implemented.

- Images built in the x86-system jobs are ones of the x86 system.
- Images built in the Arm-system jobs are ones of the Arm system.

Prerequisites

1. A cluster has been created. For details, see **Buying a CCE Cluster**.

   NOTICE

   - The build job starts a build container on the cluster node to perform build-related operations. To ensure build security, you are advised to perform security hardening on CCE cluster nodes. For details, see **Forbidding Containers to Obtain Host Machine Metadata**.
   - The build job depends on the JDK, Golang, Maven, Gradle, Ant, or Node.js compilation tool preconfigured in the build container.
   - Different IAM users under the same account can perform operations on the same cluster. To cancel the build permission from a specific IAM user, set the **servicestage:assembling:create**, **servicestage:assembling:modify**, and **servicestage:assembling:delete** permissions to **Deny** by referring to **Creating a Custom Policy**.

2. An EIP has been bound to the build node. For details, see **Assigning an EIP and Binding It to an ECS**.

Procedure

Step 1 Log in to ServiceStage, choose **Continuous Delivery** > **Build**, and click **Create Source Code Job**.

Step 2 Configure basic information.

1. Enter **Name**.
2. Enter **Enterprise Project**.

   Enterprise projects let you manage cloud resources and users by project.

   It is available after you **enable the enterprise project function**.
3. (Optional) Enter **Description**.
4. Set **Code Source**.
   - Create authorization by referring to **Authorizing a Repository** and set the code source.
   - Click **Samples** and select a required sample.
5. Select a cluster from the **Cluster** drop-down list. The cluster must belong to the enterprise project set in **Step 2.2**.

6. (Optional) Specify **Node Label** to deliver the build job to a fixed node based on the node label. For details about how to add a label, see **Adding a Node Label**.
7. Click **Next**.

Step 3 Select a build template.

- If you select **Maven**, **Ant**, **Gradle**, **Go**, or **Docker**, you can compile and archive binary packages or Docker images at the same time. Go to **Step 4**.
- If you select **Custom**, you can customize the build mode. Go to **Step 6**.

Step 4 Select an archive mode.

- **Not archived**: No Docker build job is added or archived.
- **Archive binary package**: No Docker build job is added and binary packages are archived.
- **Archive image compilation**: Docker build job is added and Docker images are archived.

Step 5 Set mandatory parameters.

To delete a parameter setting, click  on the parameter setting page.

- Build parameters

  Compilation parameters are set with different values. For details about parameter description, click a text box or  next to it.

- Image parameters

  On the  page, enter **Job Name**, **Dockerfile Path**, **Image Name**, and **Image Tag**.

- Image archiving parameters

  On the  page, enter **Job Name**, **Archive Image**, **Repository Organization**, and **Type** of the corresponding image to archive the image.

- Binary parameters

  On the  page, set the following parameters.

Parameter	Description
Task Name	Task name.
Sharing Type	Repositories are classified into public repositories and private repositories. – Public repositories are isolated from each other. Tenants in the same system can resources. – Private repositories are isolated by tenants. Users under the current tenant share resources. Other tenants cannot access resources of the current tenant.
Repository Organization	Namespace of a repository.

Parameter	Description
Software Repository	Name of a software repository.
Name	Name of the archived software package after the build completes.
Software Package Version	Version of the archived software package.
Build Package Path	Address of the binary software package generated after the compilation and build are complete. For example, **./target/xxx.jar** in the Java project.

Step 6 (Optional) Click **Advanced Configuration** to set the environment.

To add multiple tasks, you can customize them in **Advanced Configuration**.

1. Click **Add Plug-in** in the corresponding stage on the left. The **Select Job Type** page is displayed.
2. Click **Select** of the target task type to add a task type. Then, configure task parameters in the right pane of the **Environment Configurations** page.

NOTICE

When the Build Common Cmd plug-in is added to the compilation process, pay attention to the following:

- Exercise caution when inputting sensitive information in the **echo**, **cat**, or **debug** command, or encrypt sensitive information to avoid information leakage.
- Enter the compilation command. A maximum of 512 characters are allowed. Otherwise, an error message is displayed, indicating that the task input parameter is incorrect. In this case, you can add multiple Build Common Cmd plug-ins to split the command.
- When **Language** is set to **Python** and **Python Framework Type** is set to a Python project that complies with the **WSGI** standard, you need to set **Main Python Module** and **Function of the Main Python Module**. The following is an example of the main Python module and main function:

  Main Python Module: If the entry point file of the Python project is **server.py**, the main module name is **server**.

  Function of the Main Python Module: If the application function name of the Python project entry point file **server.py** is **app=get_wsgi_application()**, the function name of the main module is **app**.

Step 7 Click **Build** to save the settings and build the job.

Click **Save** to save the settings (not to start the build).

----End

Follow-Up Operations

After an application component is successfully built, you can manage it on ServiceStage. For details, see **Deploying a Component**.

7.4 Creating a Package Job

The image package can be generated with a few clicks in a build job. In this way, the entire process of package obtainment, and image compilation and archiving is automatically implemented.

Prerequisites

1. A cluster has been created. For details, see **Buying a CCE Cluster**.

   NOTICE

   - The build job starts a build container on the cluster node to perform build-related operations. To ensure build security, you are advised to perform security hardening on CCE cluster nodes. For details, see **Forbidding Containers to Obtain Host Machine Metadata**.
   - The build job depends on the JDK, Golang, Maven, Gradle, Ant, or Node.js compilation tool preconfigured in the build container.
   - Different IAM users under the same account can perform operations on the same cluster. To cancel the build permission from a specific IAM user, set the **servicestage:assembling:create**, **servicestage:assembling:modify**, and **servicestage:assembling:delete** permissions to **Deny** by referring to **Creating a Custom Policy**.

2. An EIP has been bound to the build node. For details, see **Assigning an EIP and Binding It to an ECS**.

Procedure

Step 1 Log in to ServiceStage, choose **Continuous Delivery** > **Build**, and click **Create Package Job**.

Step 2 Enter **Job Name**.

Step 3 Enter **Enterprise Project**.

Enterprise projects let you manage cloud resources and users by project.

It is available after you **enable the enterprise project function**.

Step 4 (Optional) Enter **Description**.

Step 5 Set **Package Source**.

The following upload modes are supported:

- Select the corresponding software package from the SWR software repository. Upload the software package to the software repository in advance. For details, see **Uploading the Software Package**.
- Select the corresponding software package from OBS. Upload the software package to the OBS bucket in advance. For details, see **Uploading an Object**.

Click **Select Software Package** and select the corresponding software package.

Step 6 Set **Build Type**.

- System default
  a. Select a basic image, which must be the same as the software package compilation language selected in **Step 5**.
  b. Set **Basic Image Tag**.
- Custom Dockerfile

  Enter custom commands in the compilation box.

  NOTICE

  Exercise caution when inputting sensitive information in the **echo**, **cat**, or **debug** command, or encrypt sensitive information to avoid information leakage.
- Image

  Select a basic image, which must be the same as the software package compilation language selected in **Step 5**.

Step 7 Set **Image Class**.

- Public: This is a widely used standard image that contains an OS and pre-installed public applications and is visible to all users. You can configure the applications or software in the public image as needed.
- Private: A private image contains an OS or service data, pre-installed public applications, and private applications. It is available only to the user who created it.

Step 8 Specify **Archived Image Address**.

Step 9 Select a cluster. The cluster must belong to the enterprise project set in **Step 3**.

If you use the selected cluster to perform a build job, you can deliver the build job to a fixed node through node labels. For details about how to add a label, see **Adding a Node Label**.

Step 10 Click **Build Now** to start the build.

Click **Save** to save the settings (not to start the build).

----End

Follow-Up Operations

After an application component is successfully built, you can manage it on ServiceStage. For details, see **Deploying a Component**.

7.5 Maintaining Build Jobs

For components deployed in the Kubernetes environment, you can maintain build jobs in the build job list.

Maintenance Operations

Table 7-1 Maintenance operations

Operation	Description
Editing a Build Job	See **Editing a Package Job** or **Editing a Source Code Job**. User-created jobs support this operation.
Starting a Build Job	See **Starting a Build Job**.
Viewing Details/Build History	See **Viewing Build Jobs**.
Branch/Tag	See **Branch/Tag**. Source code jobs support this operation.
Deleting a Build Job	See **Deleting a Build Job**. User-created jobs support this operation.

Editing a Package Job

Step 1 Log in to ServiceStage.

Step 2 Choose **Continuous Delivery** > **Build**.

Step 3 On the **Build** page, use either of the following methods to search for a build job:

- Select **User created** and select the CCE cluster and build job status. Then, select the target build job in the build list.
- Search for the build job created by the specified user in the search box.

Step 4 Click **More** > **Edit**. The build job configuration page is displayed.

Step 5 Enter a job name.

Step 6 (Optional) Enter the description.

Step 7 Set **Package Source**.

The following upload modes are supported:

- Select the corresponding software package from the SWR software repository. Upload the software package to the software repository. For details, see **Uploading the Software Package**.

- Select the corresponding software package from OBS. Upload the software package to the OBS bucket in advance. For details, see **Uploading an Object**.

Step 8 Set **Build Type**.

- System default
  a. Select the language of the basic image, which must be the same as that of the software package.
  b. Set **Basic Image Tag**.
     The build node can download basic images only when it can access the public network.
- Custom Dockerfile
  Enter custom commands in the compilation box.
- Image
  Set **Basic Image**.

Step 9 Set **Image Class**.

- **Public**: This is a widely used standard image that contains an OS and pre-installed public applications and is visible to all users. You can configure the applications or software in the public image as needed.
- **Private**: A private image contains an OS or service data, pre-installed public applications, and private applications. It is available only to the user who created it.

Step 10 Specify **Archived Image Address**.

Step 11 Select **Cluster**.

Step 12 (Optional) Specify **Node Label** to deliver the build job to a fixed node based on the node label.

For details about how to add a label, see **Adding a Node Label**.

Step 13 Click **Build Now** to start the build.

Click **Save** to save the settings (not to start the build).

----End

Editing a Source Code Job

Step 1 Log in to ServiceStage.

Step 2 Choose **Continuous Delivery** > **Build**.

Step 3 On the **Build** page, use either of the following methods to search for a build job:

- Select **User created** and select the CCE cluster and build job status. Then, select the target build job in the build list.
- Search for the build job created by the specified user in the search box.

Step 4 Click **More** > **Edit**. The build job configuration page is displayed.

Step 5 Enter a name.

Step 6 (Optional) Enter the description.

Step 7 Click **Modify** and set **Code Source**.

You need to create a repository authorization first. For details, see **Authorizing a Repository**.

Step 8 Select **Cluster**.

1. (Optional) Specify **Node Label** to deliver the build job to a fixed node based on the node label. For details about how to add a label, see **Adding a Node Label**.
2. Click **Next**.

Step 9 Set the environment.

1. Edit a build template.

   Select **Maven**, **Ant**, **Gradle**, **Go**, **Docker**, or **Build Common Cmd**. You can compile and archive binary packages or Docker images at the same time.

   NOTICE

   When using the Build Common Cmd template for build, enter a compilation command that contains up to 512 characters. If there are more than 512 characters, an error message is displayed, indicating that the task input parameter is incorrect. In this case, you can add multiple Build Common Cmd plug-ins to split the command.

2. Select an archive mode.
   - **Publish Build Artifact**: Binary package archive plug-in, archived to the SWR software repository.
   - **Publish Build Image**: Image archive plug-in, archived to the SWR image repository.

Step 10 Click **Build** to save the settings and start the build.

Click **Save** to save the settings (not to start the build).

----End

Starting a Build Job

Step 1 Log in to ServiceStage.

Step 2 Choose **Continuous Delivery** > **Build**.

Step 3 On the **Build** page, use either of the following methods to search for a build job:

- Select the CCE cluster where the component is deployed and the build job status, and select the specified build job in the build list.
- Search for the build job in the search box.

Step 4 Click **Build Now** to start the build.

----End

Branch/Tag

Step 1 Log in to ServiceStage.

Step 2 Choose **Continuous Delivery** > **Build**.

Step 3 On the **Build** page, use either of the following methods to search for a build job:

- Select the CCE cluster where the component is deployed and the build job status, and select the specified build job in the build list.
- Search for the build job in the search box.

Step 4 Click **Branch/Tag** and set build parameters.

1. Select **Branch/Tag**.
2. Select the corresponding branch or tag from the drop-down list.
3. Specify **Commit ID** for the branch or tag.

Step 5 Click **OK**.

----End

Deleting a Build Job

Step 1 Log in to ServiceStage.

Step 2 Choose **Continuous Delivery** > **Build**.

Step 3 On the **Build** page, use either of the following methods to search for a build job:

- Select **User created** and select the CCE cluster and build job status. Then, select the target build job in the build list.
- Search for the build job created by the specified user in the search box.

Step 4 Click **More** > **Delete**.

Step 5 Click **OK**.

----End

7.6 Managing Pipelines

One-click deployment can be achieved through pipeline. In this way, the entire process of source code pull, compilation, packaging, archiving, and deployment is automatically implemented. This unifies the integration environment and standardizes the delivery process.

In the new pipeline, the "phase/task" model is optimized to the "build/environment" model. Each pipeline includes a group of build jobs and one or more groups of environment (such as development environment, production-like environment, and production environment) tasks, each group of environment tasks contains one or more subtasks (such as deployment and test tasks) and provides templates.

ServiceStage allows a single user to create a maximum of 100+N pipelines in a project. N indicates the total number of components created by the user.

Creating a Pipeline

Step 1 Log in to ServiceStage, choose **Continuous Delivery** > **Pipeline**, and click **Create Pipeline**.

Step 2 Enter the basic pipeline information.

1. Enter **Pipeline**.
2. Enter **Enterprise Project**.

   Enterprise projects let you manage cloud resources and users by project.

   It is available after you **enable the enterprise project function**.
3. (Optional) Enter **Description**.

Step 3 Select a pipeline template.

ServiceStage provides built-in pipeline templates in typical scenarios. After you select a pipeline template, the Build/Environment model is automatically generated. You can directly use the model.

Table 7-2 Template description

Template	Description	Description
Empty template	You need to add the build/environment model.	Set this parameter as required. For details, see **Step 3.1** to **Step 3.3**.
Simple template	The "build" model is automatically added to compile and build the source code of the code library.	For details, see **Step 3.1**.
Common template	The "build/environment" model is automatically added to compile and build the source code in the code library, and the generated software package or image is continuously released to the production environment.	For details, see **Step 3.1** to **Step 3.3**.

1. Add a build job.

   Click **Select Build Job**, select a created build job, and click **OK**.

   If no build job is available, choose **Select Build Job** > **New build task** to create a source code build job or package build job. For details, see **Creating a Source Code Job** or **Creating a Package Job**.

   Repeat this step to add more build jobs. The build jobs must belong to the enterprise project selected when **adding a build job**.

2. Add a deploy job.

   Click **Add Environment** and enter an environment name. Select a deployed application component.

   If no application component is available, create and deploy an application component. For details, see **Creating and Deploying a Component**.

   Select the build job added in **Step 3.1** from the **Select Build Job** drop-down list box.

   Select build output.

   Repeat this step to add more environments.

3. Set pipeline approval.

   Click  in the environment area to set the approval mode and approver.

   - **Approval Mode**: **By all** and **By one person** are now supported.
   - **Approved By**: You can select multiple accounts as approvers. The system automatically loads all subaccounts of the account.

Step 4 Click **Create and Start** to start the pipeline.

Click **Create** to save the settings and do not execute the pipeline.

----End

Configuring the Pipeline Triggering Policy

Choose **Continuous Delivery** > **Pipeline**. On the **Pipeline** page that is displayed, set the pipeline triggering policy as follows.

Table 7-3 Triggering policies

Policy	Mode	Description
Manual	-	Select the pipeline task to be triggered and click **Start** to manually start the pipeline.

Policy	Mode	Description
Automatic	-	Set the code source, corresponding namespace, repository name, and branch. When code is submitted to the corresponding branch of the source code repository, the pipeline is automatically triggered. You can set a maximum of eight trigger sources. The procedure is as follows: 1. Select a pipeline and choose **More** > **Triggering Policy**. 2. Set **Type** to **Automatic**. 3. Select **Source Code Repository** to push the code to the selected source code repository. 4. Click **OK**.
Scheduled	Single-time	Set the triggering time to trigger a single-time pipeline. The procedure is as follows: 1. Select a pipeline and choose **More** > **Triggering Policy**. 2. Set **Type** to **Scheduled**. 3. Specify **Triggered**. 4. Click **OK**.

Policy	Mode	Description
	Periodic	Set the triggering time segment, interval, and period to implement periodic pipeline triggering. The procedure is as follows: 1. Select a pipeline and choose **More** > **Triggering Policy**. 2. Set **Type** to **Scheduled**. 3. Enable **Periodic Triggering**. 4. Specify **Period**, **Triggered**, **Effective Time**, and **Period**. 5. Click **OK**.

Cloning a Pipeline

You can clone a pipeline to generate a new pipeline based on the existing pipeline configuration.

Step 1 Log in to ServiceStage and choose **Continuous Delivery** > **Pipeline**.

Step 2 Select a pipeline and choose **More** > **Clone**.

Step 3 ServiceStage automatically loads configurations of the clone pipeline. You can then modify the configurations as required by referring to **Creating a Pipeline**.

Step 4 Click **Create and Start** to start the pipeline.

Click **Create** to save the settings and do not execute the pipeline.

----End

Follow-Up Operations

After the pipeline is started, you can build and deploy applications in one-click mode. For details about maintenance operations after application components are deployed, see **Component O&M**.

7.7 Authorizing a Repository

You can create repository authorization so that build projects and application components can use the authorization information to access the software repository.

Step 1 Log in to ServiceStage, choose **Continuous Delivery** > **Repository Authorization**, and click **Create Authorization**.

Step 2 Configure authorization information by referring to the following table. Parameters marked with an asterisk (*) are mandatory.

Table 7-4 Authorization information

Parameter	Description
*Name	Authorization name, which cannot be changed after being created.
*Repository Type	The following supported official repository types vary depending on regions: • GitHub (**https://github.com**) Authorization mode: OAuth or private token. • Bitbucket (**https://bitbucket.org**) Authorization mode: OAuth or private Bitbucket. • GitLab (**https://gitlab.com**) Authorization mode: OAuth or private token. **NOTE** ServiceStage allows you to access official and private GitLab source code repositories using private tokens. – Visit the official GitLab source code repository, obtain and enter a private token as prompted, and select **Verify token (access the repository address from the public network)**. – Visit the private GitLab source code repository, enter the correct private GitLab source code repository address and private token as prompted. You do not need to select **Verify token (access the repository address from the public network)**.

Step 3 Click **Create**.

----End

8 Microservice Engine

8.1 Cloud Service Engine Overview

Cloud Service Engine (CSE) provides service registry, service governance, and configuration management. It allows you to quickly develop microservice applications and implement high-availability O&M, and supports multiple languages, multiple runtime systems, and Spring Cloud and Apache ServiceComb Java Chassis (Java chassis) frameworks.

You can use the professional microservice engine named "Cloud Service Engine" or create an exclusive microservice engine.

- An exclusive microservice engine is physically isolated. A tenant exclusively uses an exclusive microservice engine.
- The professional microservice engine does not support multiple AZs.
- You can configure multiple AZs when creating an exclusive engine.
- After a microservice engine is created, the AZ cannot be modified. Select a suitable AZ when creating a microservice engine.
- Exclusive microservice engines cannot run across CPU architectures.

8.2 Creating a Microservice Engine

This section describes how to create a microservice engine.

Prerequisites

A microservice engine runs on a VPC. Before creating a microservice engine, ensure that a VPC and subnet are available.

You have created a VPC. For details, see **Creating a VPC**.

If the engine is created using an account with the minimum permission for creating engines, for example, **cse:engine:create** in the **fine-grained permission dependencies of microservice engines**, the default VPC security group cse-engine-default-sg needs to be preset by the primary account and the rules listed in **Table 8-1** need to be added.

For details, see **Adding a Security Group Rule**.

Table 8-1 cse-engine-default-sg rules

Direction	Priority	Policy	Protocol and Port	Type	Source Address
Inbound	1	Allow	ICMP: all	IPv6	::/0
	1	Allow	TCP: 30100–30130	IPv6	::/0
	1	Allow	All	IPv6	cse-engine-default-sg
	1	Allow	TCP: 30100–30130	IPv4	0.0.0.0/0
	1	Allow	ICMP: all	IPv4	0.0.0.0/0
Outbound	100	Allow	All	IPv4	0.0.0.0/0
	100	Allow	All	IPv6	::/0

Procedure

Step 1 Go to the **Buy Exclusive Microservice Engine page**.

NOTE

- By default, a maximum of five exclusive microservice engines can be created for each project. To create more exclusive microservice engines, submit a service ticket to increase the quota. For details, see **Creating a Service Ticket**.
- For details about projects, see **Projects**.

Step 2 Set parameters according to the following table. Parameters marked with an asterisk (*) are mandatory.

Parameter	Description
*Billing Mode	Billing mode. Currently, **Pay-per-use** is supported.

Parameter	Description
*Enterprise Project	Select the project where the microservice engine is located. You can search for and select the required enterprise project from the drop-down list. Enterprise projects let you manage cloud resources and users by project. An enterprise project can be used after it is created and enabled. For details, see **Enabling the Enterprise Project Function**. By default, **default** is selected. **NOTE** • The enterprise project cannot be changed once the microservice engine is created. • When a microservice engine is in use, do not disable the enterprise project. Otherwise, the engine will not be displayed in the engine list, affecting normal use.
*Specification	Select the microservice instance quota. **NOTICE** The specification cannot be changed once the microservice engine is created.
*Engine Type	Microservice engine type. If the engine type is cluster, the engine is deployed in cluster mode and supports host-level DR.
*Name	Name of a microservice engine. The name contains 3 to 24 characters, including letters, digits, and hyphens (-), and starts with a letter but cannot end with a hyphen. This name cannot be changed after the engine is created. **NOTICE** Microservice engine name cannot be **default**.
*AZ	Availability zone. Select one or three AZs for the engine based on the number of AZs in the environment. • Select one AZ to provide host-level DR. • Select three AZs to provide AZ-level DR. **NOTE** • The AZ of a created microservice engine cannot be changed. • The AZs in one region can communicate with each other over an intranet. • Multiple AZs enhance DR capabilities.

Parameter	Description
*Network	Select a VPC and subnet to provision logically isolated, configurable, and manageable virtual networks for your engine. • To use a created VPC, search for and select a VPC created under the current account from the drop-down list. • To use a new VPC, click **Create VPC** in the drop-down list. For details, see **Creating a VPC**. • To use a shared VPC, select a VPC that another account shares with the current account from the drop-down list. VPC owners can share the subnets in a VPC with one or multiple accounts through **Resource Access Manager (RAM)**. Through VPC sharing, you can easily configure and manage multiple accounts' resources at low costs. For more information about VPC and subnet sharing, see **VPC Sharing**. **NOTE** The VPC cannot be changed once the engine is created.
Description	Click  and enter the engine description.
Authentication Mode	The exclusive microservice engine with security authentication enabled provides the system management function using the role-based access control (RBAC) through the microservice engine console. • Select **Enable security authentication**: 1. Determine whether to enable **Authenticate Programming Interface**. After it is enabled, you need to add the corresponding account and password to the microservice configuration file. Otherwise, the service cannot be registered with the engine. After it is disabled, you can register the service with the engine without configuring the account and password in the microservice configuration file, which improves the efficiency. You are advised to disable this function when accessing the service in a VPC. 2. Enter and confirm the password of user **root**. Keep the password secure. • Select **Disable security authentication**: Disable security authentication. You can enable it after the instance is created.

Step 3 Click **Buy**. The page for confirming the engine information is displayed.

Step 4 Click **Submit** and wait until the engine is created.

NOTE

- It takes about 31 minutes to create a microservice engine.
- After the microservice engine is created, its status is **Available**. For details about how to view the microservice engine status, see **Viewing Microservice Engine Information**.
- If the microservice engine fails to be created, view the failure cause on the **Operation** page and rectify the fault. Then, you can perform the following operations:
  - In the **Microservice Engine Information** area, click **Retry** to create an engine again.
  - If the retry fails, delete the microservice engine that fails to be created. For details, see **Deleting an Exclusive Microservice Engine**.

----End

8.3 Managing Microservice Engines

8.3.1 Viewing Microservice Engine Information

In the **Microservice Engine Information** area, you can view the engine information as shown in **Table 8-2**.

Procedure

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 In the **Microservice Engine Information** area, you can view the engine information as shown in **Table 8-2**.

Table 8-2 Engine details

Item	Description
Name	Engine name entered when **Creating a Microservice Engine**. Click to copy it. Click to change the engine name.
Engine ID	Engine ID. You can click to copy it.

Item	Description
Status	Engine status, which can be: • Creating • Available • Unavailable • Configuring • Deleting • Upgrading • Resizing • Creation failed • Deletion failed • Upgrade failed • Resizing failed • Frozen
Version	Engine version.
Engine Type	Engine type selected when **Creating a Microservice Engine**.
AZ	Availability zone selected when **Creating a Microservice Engine**.

----End

8.3.2 Obtaining the Service Center Address of a Microservice Engine

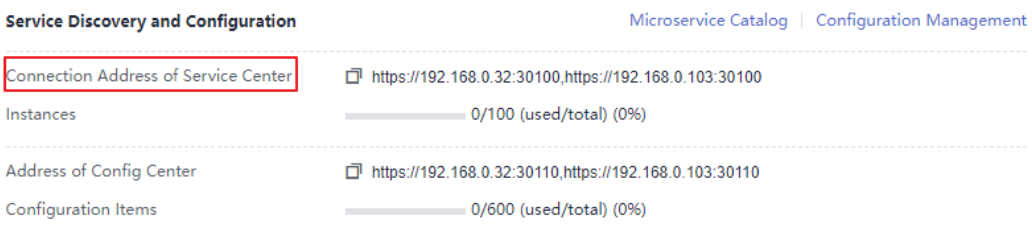
This section describes how to obtain the service center address of a microservice engine. The service center address cannot be changed after the engine is created.

Procedure

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 In the **Service Discovery and Configuration** area, view the service center address of the microservice engine.



----End

8.3.3 Obtaining the Configuration Center Address of a Microservice Engine

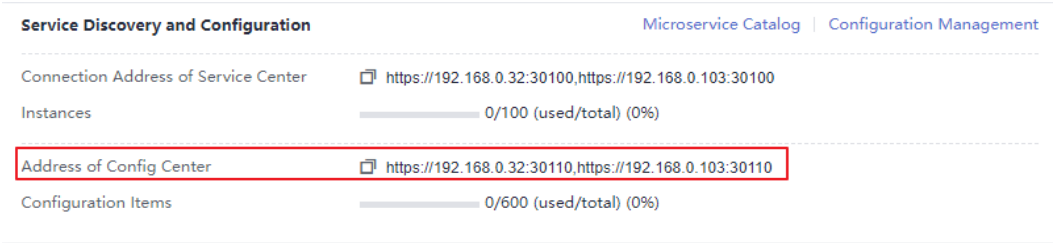
This section describes how to obtain the configuration center address of a microservice engine.

Procedure

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 In the **Service Discovery and Configuration** area, view the configuration center address of the microservice engine.



NOTE

- For microservice engine 1.x, the port number of the configuration center address is 30103.
- For microservice engine 2.x, the port number of the configuration center address is 30110.

----End

8.3.4 Viewing the Instance Quota of a Microservice Engine

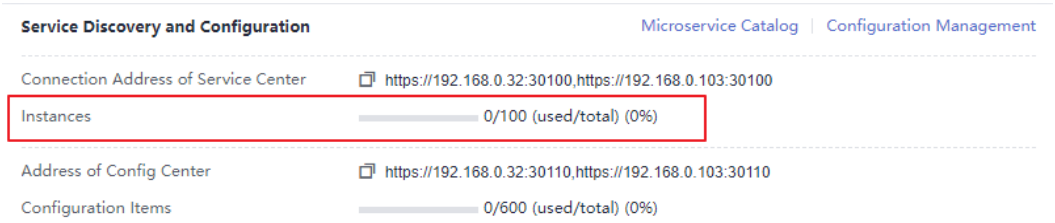
This section describes how to view the instance quota and quota usage of a microservice engine.

Procedure

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 In the **Service Discovery and Configuration** area, view the instance quota and quota usage of the microservice engine.



----End

8.3.5 Viewing the Configuration Item Quota of a Microservice Engine

This section describes how to view the configuration item quota and quota usage of a microservice engine.

NOTE

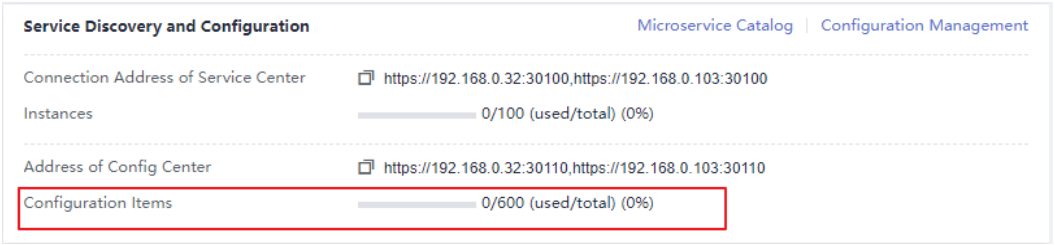
This section applies only to microservice engine 2.x.

Procedure

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 In the **Service Discovery and Configuration** area, view the configuration item quota and quota usage of the microservice engine.



----End

8.3.6 Configuring Backup and Restoration of a Microservice Engine

The ServiceStage console provides the backup and restoration functions. You can back up and restore microservice engine data, including microservices, contracts, configurations, and account roles.

You can customize backup policies to periodically or manually back up microservice engines.

Background

- Each exclusive microservice engine supports a maximum of 15 successful backups, including a maximum of 10 manual backups and a maximum of 5 automatic backups.

- The backup data will be stored for 10 days. Expired backup data will be deleted.

Automatic Backup

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 In the **Backup and Restoration** area, click **Automatic backup settings** and set backup parameters.

Table 8-3 Automatic backup parameters

Parameter	Description
Automatic Backup	After automatic backup is disabled, the previously set backup policy will be deleted. In this case, you need to set the backup policy again.
Backup Interval	Backup period. This parameter takes effect after **Automatic Backup** is enabled.
Trigger Time	Time at which a backup task starts. Only the hour is supported. This parameter takes effect after **Automatic Backup** is enabled.

Step 4 Click **OK**.

Once the backup policy is set, the backup task is triggered within one hour after the preset time.

----End

Manual Backup

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 In the **Backup and Restoration** area, click **Create Manual Backup** and set backup parameters.

Table 8-4 Manual backup parameters

Parameter	Description
Name	Name of a backup task. The name cannot be changed after the backup task is created.
Remarks	(Optional) Description about the backup task.

Step 4 Click **OK**.

----End

Restoring Backup Data

NOTICE

The backup data will overwrite the current data of the microservice engine. As a result, the microservice and service instances may be messed, and dynamic configurations may be lost. Exercise caution when performing this operation.

If security authentication is enabled, the backup data contains the account information. You are advised to disable security authentication before restoring the backup data. Otherwise, the authentication for accessing the microservice engine may fail after the restoration.

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 In the **Backup and Restoration** area, click **Restore** in the **Operation** column of the row that contains the specified backup data.

1. Select **I have read and fully understand the risks**.
2. Click **OK**. To view the restoration status, click **Restoration History** in the **Backup and Restoration** area.

----End

Deleting Backup Data

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 In the **Backup and Restoration** area, click **Delete** in the **Operation** column of the target backup data. In the displayed dialog box, enter **DELETE** and click **OK**.

----End

8.3.7 Managing Public Network Access for a Microservice Engine

8.3.7.1 Binding an EIP

Exclusive microservice engines that are bound with EIPs can be accessed from the public network.

NOTICE

Microservice engines with security authentication disabled do not have the authentication and authorization capabilities. Opening those engines to the public network may cause security risks and increases the system vulnerability. For example, data assets such as configurations and service information may be stolen.

Do not use this function in a production environment or a network environment with high security requirements.

Prerequisites

An EIP has been created.

For details about how to create an EIP, see **Assigning an EIP**.

Procedure

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 In the **Network Configuration and Security** area, click **Bind EIP**.

Step 4 Read the security risk prompt in the displayed dialog box and select **I understand the security risks**.

Step 5 In the **EIP** drop-down list, select the EIP to be bound.

Step 6 Click **OK**.

----End

8.3.7.2 Unbinding an EIP

If an EIP has been bound to an exclusive microservice engine, you can unbind the EIP from the engine to disable the public network access to the engine.

Procedure

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 In the **Network Configuration and Security** area, click **Unbind EIP**.

Step 4 In the displayed dialog box, click **OK**.

----End

8.3.8 Viewing Microservice Engine Operation Logs

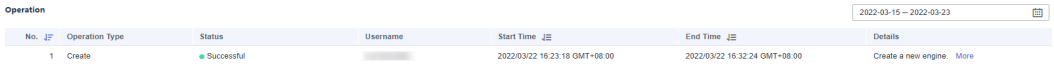
In the **Operation** area, you can view the operation logs of a microservice engine.

Procedure

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 In the **Operation** area, view the operation logs of a microservice engine.



- Click  in the upper right corner to view operation logs in a specified period.
- Click **More** in the **Details** column of a specified operation log to view details about the operation log.

----End

8.3.9 Upgrading a Microservice Engine Version

Microservice engines are created using the latest engine version. When a later version is released, you can upgrade your engine.

NOTICE

- During the microservice engine upgrade, the microservice and engine are intermittently disconnected, but services of running microservices are not affected. You are advised not to upgrade, restart, or change microservices when upgrading a microservice engine.
- Only exclusive microservice engines can be upgraded. Version rollback is not supported after the upgrade.
- For details about the precautions for upgrading an exclusive microservice engine from 1.x to 2.x, see **What Do I Need to Know Before Upgrading an Exclusive Microservice Engine?**

Background

During upgrade, two instances are upgraded in rolling mode without service interruptions. However, one of the two access addresses may be unavailable. In this case, you need to quickly switch to the other instance. Currently, ServiceComb SDK and Mesher support instance switching. If you call the APIs of the service center and configuration center for service registry and discovery, instance switching is required.

Procedure

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 In the **Microservice Engine Information** area, click **Upgrade**.

Step 4 Select **Target Version** and view the version description. Determine whether to upgrade the software to this version.

Step 5 Click **OK**.

If the upgrade fails, click **Retry** to perform the upgrade again.

----End

8.3.10 Deleting a Microservice Engine

You can delete an exclusive microservice engine if it is no longer used.

NOTICE

- Deleted engines cannot be recovered. Exercise caution when performing this operation.
- For engine 1.x, if the cse_admin_trust agency is missing, deleting the engine will cause residual DNS, VPC, and security group resources on the tenant side. You need to delete them by yourself.

Background

You can delete exclusive microservice engines in the following states:

- Available
- Unavailable
- Creation failed
- Resizing failed
- Upgrade failed

Procedure

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 In the **Microservice Engine Information** area, click **Delete**. In the displayed dialog box, enter **DELETE** and click **OK**.

NOTE

If the deletion fails, click **Force Delete**.

----End

8.3.11 Managing Security Authentication for a Microservice Engine

A microservice engine may be used by multiple users. Different users must have different microservice engine access and operation permissions based on their responsibilities and permissions. If security authentication is enabled for an exclusive microservice engine, grant different access and operation permissions to users based on the roles associated with the accounts used by the users to access the microservice engine.

For details about security authentication, see **System Management**.

Currently, Java chassis and Spring Cloud support security authentication for microservices. The Java chassis version must be 2.3.5 or later, and Spring Cloud must integrate Spring Cloud Huawei 1.6.1 or later.

You can enable or disable security authentication for the exclusive microservice engine based on service requirements.

- **Enabling Security Authentication**

  If a microservice engine is available with security authentication disabled, you can enable security authentication based on service requirements.

  After security authentication is enabled and programming interface authentication is also enabled, if security authentication parameters are not configured for the microservice components connected to the engine, or the security authentication account and password configured for the microservice components are incorrect, the heartbeat of the microservice components fails and the service is forced to go offline. Perform the following steps:

  - Spring Cloud: For details, see **Connecting Spring Cloud Applications to CSE Engines**.
  - Java Chassis: For details, see **Connecting Java Chassis Applications to CSE Engines**.

- **Disabling Security Authentication**

  If a microservice engine is available with security authentication enabled, you can disable security authentication based on service requirements.

  After security authentication is disabled for a microservice component, service functions of the microservice component are not affected no matter whether security authentication parameters are configured for the microservice component.

Enabling Security Authentication

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 In the **Network Configuration and Security** area, click **Set Authentication**.

- If the engine version is earlier than 1.2.0, go to **Step 4**.
- If the engine version is 1.2.0 or later, go to **Step 5**.

Step 4 Upgrade the engine to 1.2.0 or later.

1. Click **Upgrade**.
2. Select **Target Version** and view the version description. Determine whether to upgrade the software to this version. Then, click **OK**.
3. Select the upgraded microservice engine. In the **Network Configuration and Security** area, click **Set Authentication**.

Step 5 On the **System Management** page, enable security authentication.

- To enable security authentication for the first time, click **Enable security authentication**.

  You need to create user **root** first. Enter and confirm the password of user **root**. Then, click **Create Now**.
- Enable security authentication again and enter the name and password of the account associated with the **admin** role in the engine.

Step 6 (Optional) Create a role based on service requirements. For details, see **Roles**.

Step 7 (Optional) Create an account based on service requirements. For details, see **Accounts**.

Step 8 On the **System Management** page, click **Enable security authentication** and configure the security settings.

- If you enable **Authenticate Console**, go to **Step 10**.

  After **Authenticate Console** is enabled, you need to use an account and password to log in to the CSE console. The login user can only view and configure services on which the user has permission.
- If you enable **Authenticate Programming Interface**, go to **Step 9**.

  After **Authenticate Programming Interface** is enabled, **Authenticate Console** is automatically enabled.

  After it is enabled, you need to add the corresponding account and password to the microservice configuration file. Otherwise, the service cannot be registered with the engine.

  After it is disabled, you can register the service with the engine without configuring the account and password in the microservice configuration file, which improves the efficiency. You are advised to disable this function when accessing the service in a VPC.

Step 9 Configure the SDK. For microservice components that have been deployed but not configured with security authentication parameters, configure the account name and password for security authentication and then upgrade the component. For details, see **Configuring the Security Authentication Account and Password for a Microservice**.

Step 10 Click **OK**.

After the microservice engine is updated and the engine status changes from **Configuring** to **Available**, security authentication is enabled successfully.

----End

Disabling Security Authentication

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 In the **Network Configuration and Security** area, click **Set Authentication**.

Step 4 On the **System Management** page, click **Set Authentication**.

Step 5 On the **Security Settings** page, disable **Authenticate Console**.

Step 6 Click **OK**. After the microservice engine is updated and the engine status changes from **Configuring** to **Available**, security authentication is disabled successfully.

NOTE

After security authentication is disabled, accounts created on the engine will not be deleted.

----End

8.4 Using Microservice Engines

8.4.1 Using the Microservice Dashboard

You can view metrics related to microservices through the dashboard in real time. Based on abundant and real-time dashboard data, you can take corresponding governance actions for microservices.

Background

- If a microservice application is deployed on ServiceStage, you need to configure the microservice engine during application deployment. The application automatically obtains the service registry and discovery address, configuration center address, and dashboard address. You do not need to configure the monitor address.
- If the microservice application is locally started and registered with the microservice engine, manually configure the monitor address before using the dashboard.

  For details, see **Using Dashboard**.

Procedure

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 Choose **Dashboard**.

- For microservice engines with security authentication disabled, go to **Step 5**.
- For microservice engines with security authentication enabled, go to **Step 4**.

Step 4 In the displayed **Security Authentication** dialog box, enter the account name and password, and click **OK**.

NOTE

- If you connect to the microservice engine for the first time, enter the account name **root** and the password entered when **Creating a Microservice Engine**.
- For details about how to create an account, see **Adding an Account**.

Step 5 On the **Dashboard** page, select an application from the drop-down list box and enter a microservice name in the search box. The operating metrics of the microservice are displayed.

Click **View Diagram** to view the description of operating metrics.

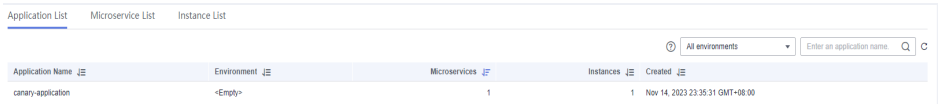
Step 6 Select a sorting order to sort the filtered microservices.

----End

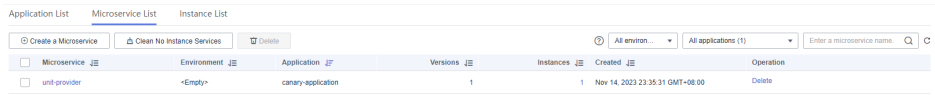
8.4.2 Managing Microservices

You can use the microservice catalog to view microservice details and search for target microservices to maintain microservices. The **Microservice Catalog** page contains the following tabs:

- **Application List**: displays all applications of the current microservice engine. You can search for the target application by application name, or filter applications by environment. For details, see **Viewing the Application List**.



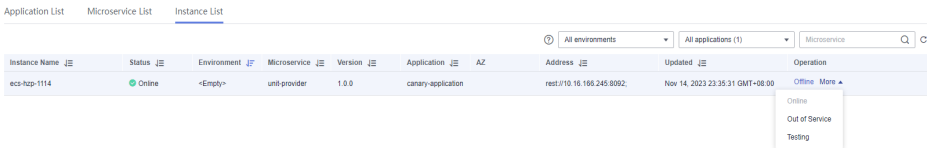
- **Microservice List**: For details about the operations supported by in **Microservice List**, see the following table.



Operation	Description
Viewing the Microservice List	Displays all microservices of the current microservice engine. You can search for the target microservice by microservice name, or filter microservices by environment and application.
Viewing Microservice Details	On the microservice details page, you can view the instance list, called services, calling services, dynamic configuration, and service contract.
Creating a Microservice	Creates a microservice.

Operation	Description
Cleaning Versions Without Instances	Cleans microservice versions that have no instances.
Deleting a Microservice	Deletes a microservice that is no longer used.
Dynamic Configuration	Creates a microservice-level configuration.
Dark Launch	In dark launch, new features are tested in a selected group of users. When the features become mature and stable, they are released to all users.

- **Instance List**: For details about the operations supported by in **Instance List**, see the following table.



Operation	Description
Viewing the Instance List	Displays all instances of the current microservice engine. You can search for the target instance by microservice name, or filter instances by environment and application.
Changing the Instance Status	**Status** indicates the status of a microservice instance.

Viewing the Application List

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 Choose **Microservice Catalog**.

- For engines with security authentication disabled, go to **Step 5**.
- For engines with security authentication enabled, go to **Step 4**.

Step 4 In the displayed **Security Authentication** dialog box, enter the account name and password, and click **OK**.

NOTE

- If you connect to the microservice engine for the first time, enter the account name **root** and the password entered when **Creating a Microservice Engine**.
- For details about how to create an account, see **Adding an Account**.

Step 5 Click **Application List** to view details about all applications of the current account under the engine.

You can search for the target application by application name, or filter applications by environment.

----End

Viewing the Microservice List

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 Choose **Microservice Catalog**.

- For engines with security authentication disabled, go to **Step 5**.
- For engines with security authentication enabled, go to **Step 4**.

Step 4 In the displayed **Security Authentication** dialog box, enter the account name and password, and click **OK**.

NOTE

- If you connect to the microservice engine for the first time, enter the account name **root** and the password entered when **Creating a Microservice Engine**.
- For details about how to create an account, see **Adding an Account**.

Step 5 Click **Microservice List** to view all microservices of the current account under the engine.

You can search for the target microservice by microservice name, or filter microservices by environment and application.

----End

Viewing Microservice Details

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 Choose **Microservice Catalog**.

- For microservice engines with security authentication disabled, go to **Step 5**.
- For engines with security authentication enabled, go to **Step 4**.

Step 4 In the displayed **Security Authentication** dialog box, enter the account name and password, and click **OK**.

NOTE

- If you connect to the microservice engine for the first time, enter the account name **root** and the password entered when **Creating a Microservice Engine**.
- For details about how to create an account, see **Adding an Account**.

Step 5 Click the microservice to be viewed in **Microservice List**. On the displayed page, view the instance list, called services, calling services, configurations, and service contract.

----End

Creating a Microservice

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 Choose **Microservice Catalog**.

- For engines with security authentication disabled, go to **Step 5**.
- For engines with security authentication enabled, go to **Step 4**.

Step 4 In the displayed **Security Authentication** dialog box, enter the account name and password, and click **OK**.

NOTE

- If you connect to the microservice engine for the first time, enter the account name **root** and the password entered when **Creating a Microservice Engine**.
- For details about how to create an account, see **Adding an Account**.

Step 5 Choose **Microservice List** > **Create a Microservice** and set microservice parameters by referring to the following table. Parameters marked with an asterisk (*) are mandatory.

Parameter	Description
*Microservice	Microservice name, for example, **myServiceName**.
*Application	Name of the application to which the microservice belongs. Microservices are isolated by applications.
*Version	Microservice version. The default value is **1.0.0**. **NOTE** The microservice version is in the format of X.Y.Z or X.Y.Z.B, where X, Y, Z, and B are digits and range from 0 to 32767. The value contains 3 to 46 characters.
*Environment	Environment where the microservice is located to isolate microservice data, including the version and instance.
Detail	Microservice description.

Step 6 Click **OK**.

Once the microservice is created, it will be displayed in **Microservice List**.

----End

Cleaning Versions Without Instances

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 Choose **Microservice Catalog**.

- For engines with security authentication disabled, go to **Step 5**.
- For engines with security authentication enabled, go to **Step 4**.

Step 4 In the displayed **Security Authentication** dialog box, enter the account name and password, and click **OK**.

NOTE

- If you connect to the microservice engine for the first time, enter the account name **root** and the password entered when **Creating a Microservice Engine**.
- For details about how to create an account, see **Adding an Account**.

Step 5 Choose **Microservice List** > **Clean No Instance Services**. Select the microservice version without instances to be cleaned.

You can search for the target microservice by microservice name, or filter microservices by environment and application.

Step 6 Click **OK**.

----End

Deleting a Microservice

NOTICE

- After a microservice is deleted, you can restore it by referring to **Restoring Backup Data**.
- If the service to be deleted has instances, delete the instances first. Otherwise, the service will be registered again.

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 Choose **Microservice Catalog**.

- For engines with security authentication disabled, go to **Step 5**.
- For engines with security authentication enabled, go to **Step 4**.

Step 4 In the displayed **Security Authentication** dialog box, enter the account name and password, and click **OK**.

NOTE

- If you connect to the microservice engine for the first time, enter the account name **root** and the password entered when **Creating a Microservice Engine**.
- For details about how to create an account, see **Adding an Account**.

Step 5 Click **Microservice List**.

- To delete microservices in batches, select the microservices to be deleted and click **Delete** above the microservices.
- To delete one microservice, locate the row that contains the microservice to be deleted and click **Delete** in the **Operation** column.

Step 6 In the displayed dialog box, enter **DELETE** to confirm the deletion and click **OK**.

----End

Dynamic Configuration

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 Choose **Microservice Catalog**.

- For engines with security authentication disabled, go to **Step 5**.
- For engines with security authentication enabled, go to **Step 4**.

Step 4 In the displayed **Security Authentication** dialog box, enter the account name and password, and click **OK**.

NOTE

- If you connect to the microservice engine for the first time, enter the account name **root** and the password entered when **Creating a Microservice Engine**.
- For details about how to create an account, see **Adding an Account**.

Step 5 Click **Microservice List**.

Step 6 Click the target microservice.

Step 7 Choose **Dynamic Configuration**. On the **Dynamic Configuration** tab, perform the following operations.

NOTICE

Configuration items are stored in plaintext. Do not include sensitive data.

Operation	Procedure
Creating a configuration item	See **Creating a Microservice-Level Configuration**. **Microservice-level** is selected for **Configuration Range** and **Microservices** is set to the current microservice.

Operation	Procedure
Viewing historical versions	Click **View Historical Version** in the **Operation** column of the target configuration item.
Disabling a Configuration Item	1. In the **Operation** column of the target configuration item, choose **More** > **Disable**. 2. Click **OK**.
Modifying a configuration item	1. Click **Edit** in the **Operation** column corresponding to the target configuration item. 2. On the configuration details page, click **Edit**. 3. On the **Configuration Details** tab, enter the new configuration. 4. Click **Save**.
Deleting a configuration item	1. In the **Operation** column of the target configuration item, choose **More** > **Delete**. 2. Click **OK**.

----End

Dark Launch

In dark launch, new features are tested in a selected group of users. When the features become mature and stable, they are released to all users. This ensures the smooth feature rollout.

NOTE

- For microservices developed based on the ServiceComb Java Chassis framework, add dependency **darklaunch** or **handler-router** to POM and add **servicecomb.router.type=router** to the configuration file.
- For microservices developed based on the Spring Cloud Huawei framework, add dependency **spring-cloud-starter-huawei-router** to POM.

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 Choose **Microservice Catalog**.

- For engines with security authentication disabled, go to **Step 5**.
- For engines with security authentication enabled, go to **Step 4**.

Step 4 In the displayed **Security Authentication** dialog box, enter the account name and password, and click **OK**.

NOTE

- If you connect to the microservice engine for the first time, enter the account name **root** and the password entered when **Creating a Microservice Engine**.
- For details about how to create an account, see **Adding an Account**.

Step 5 In the microservice list, click a microservice. On the displayed page, choose **Dark Launch**.

Step 6 Click **Add Launch Rule**.

- To add a launch rule by **Weight**:
  a. Click **Weight**.
  b. Set the following parameters.

Item	Description
Rule Name	Name of the rule.
Scope	▪ Microservice version to which the rule applies. ▪ Select **Do you want to add a customized version?** and add a new version as prompted.
Rule Configuration	Traffic allocation rate for the selected version. Traffic is evenly allocated to the selected service versions based on the configured value.

  c. Click **OK** to complete the weight rule configuration and dark launch.
- To add a launch rule by **Customization**:

  NOTE

  Dark launch rules can be delivered only after dark launch is implemented for microservices developed based on the ServiceComb Java Chassis framework using dependency **darklaunch** and microservices developed based on the Spring Cloud Huawei framework. Dark launch rules that depend on **handler-router** need to be manually delivered in the configuration center.

  a. Click **Custom**.
  b. Set the following parameters.

Item	Description
Rule Name	Name of the rule.
Scope	▪ Microservice version to which the rule applies. ▪ Select **Do you want to add a customized version?** and add a new version as prompted.

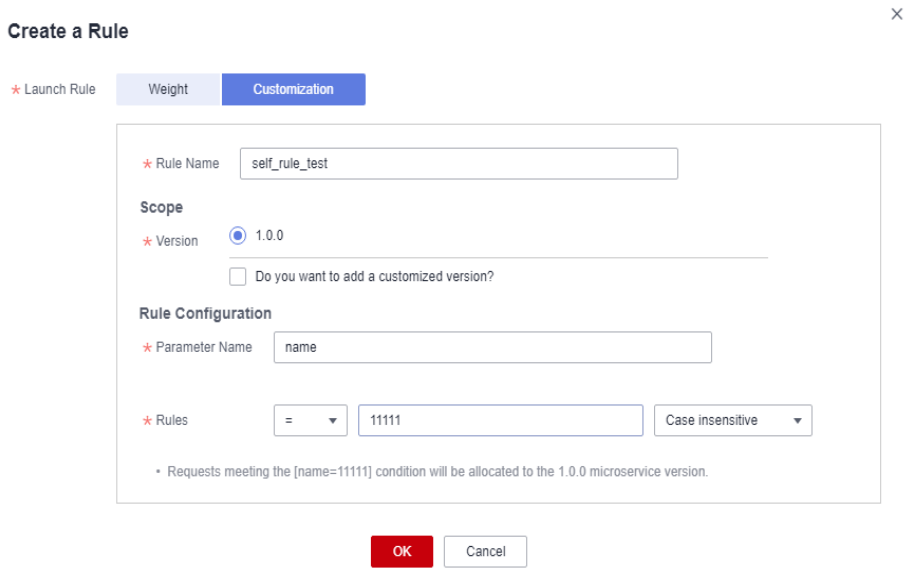
Item	Description
Rule Configuration	Configure the matching rule. When **darklaunch** is used to implement dark launch, this parameter configures **policyCondition**. Otherwise, it configures **Headers**. ▪ Parameter Name Set this parameter based on the parameter name of contract or the customized key of the header. ▪ Rules By selecting the matching character and the value corresponding to the key of contract or the key of the header, requests that meet the rules are allocated to the microservice version. **NOTE** ○ If **~** is selected from the drop-down list next to **Rules**, the asterisk (*) and question mark (?) in the **Rules** value can be used for fuzzy matching. The asterisk (*) indicates a character of any length, and the question mark (?) indicates one character. For example, if the rule value of **Name** is set to ***1000**, all **Name** fields ending with 1000 can be matched. ○ If **~** is not selected from the drop-down list next to **Rules**, the asterisk (*) and question mark (?) in the **Rules** value cannot be used for fuzzy matching.

c. Click **OK** to complete the customization rule configuration and dark launch.

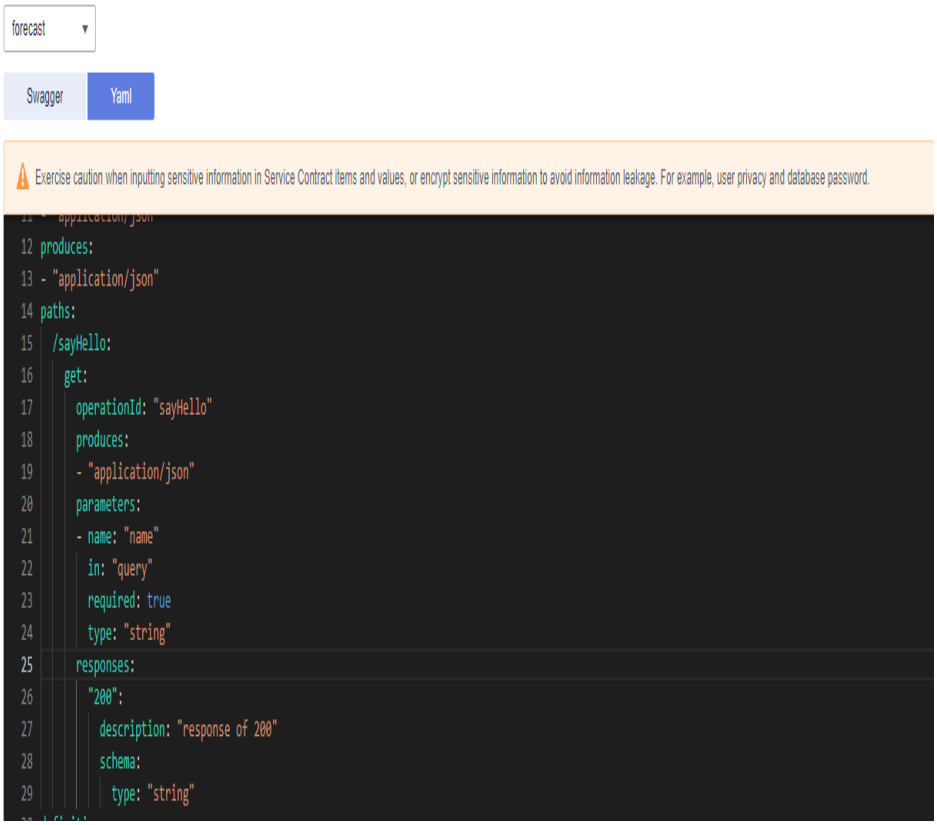
----End

Examples of delivering dark launch rules:

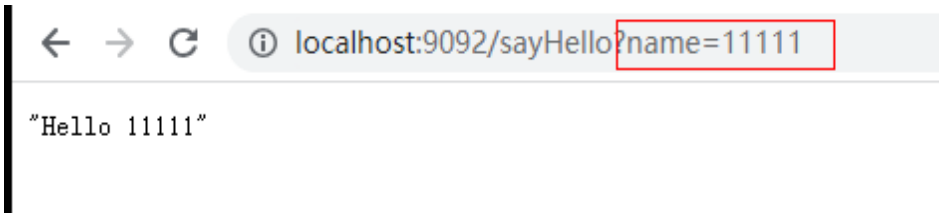
- For microservices developed based on the ServiceComb Java Chassis framework, rules are delivered based on dependency **darklaunch** on the microservice engine page. You can add dark launch rules in customized mode.



This key must exist in the contract. It is possible that the server API is **String paramA**, but **paramB** is actually generated after the annotation is added. Therefore, **paramB** should be set here.

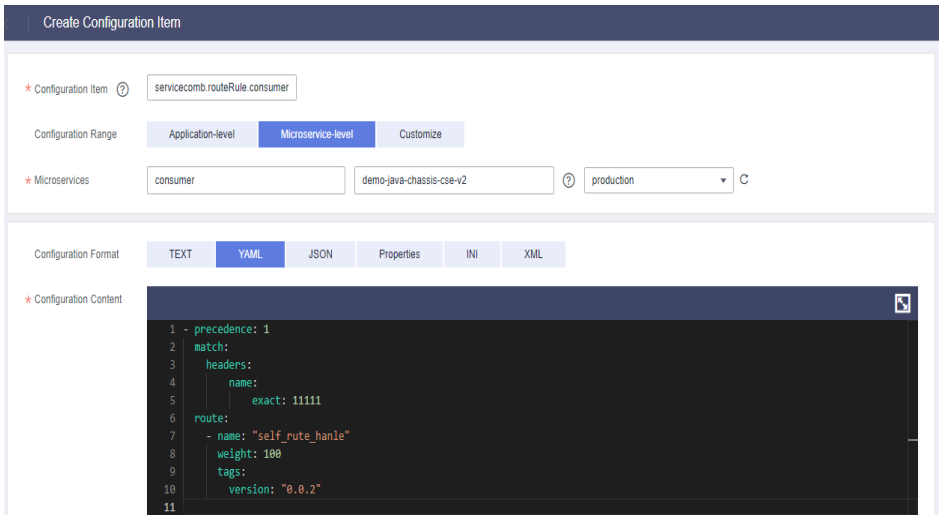


By selecting the matching character and the value corresponding to the key of contract, requests that meet the rules are allocated to the microservice version.

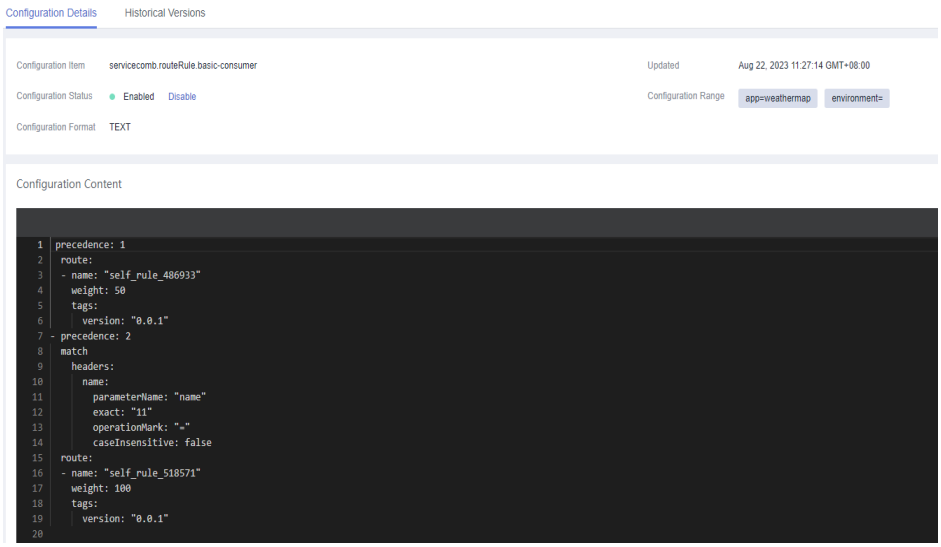


A delivered rule is as follows. The configuration item is **cse.darklaunch.policy.*${serviceName}***.

- For microservices developed based on the ServiceComb Java Chassis framework, dark launch rules that depend on **handler-router** need to be manually delivered in the configuration center. The configuration item is **servicecomb.routeRule.*${serviceName}***. The content is as follows:



- For microservices developed based on the Spring Cloud Huawei framework, dark launch rules delivered on the microservice engine page are as follows:



Viewing the Instance List

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 Choose **Microservice Catalog**.

- For microservice engines with security authentication disabled, go to **Step 5**.
- For microservice engines with security authentication enabled, go to **Step 4**.

Step 4 In the displayed **Security Authentication** dialog box, enter the account name and password, and click **OK**.

NOTE

- If you connect to the microservice engine for the first time, enter the account name **root** and the password entered when **Creating a Microservice Engine**.
- For details about how to create an account, see **Adding an Account**.

Step 5 Click **Instance List** to view all instances of the engine.

You can search for the target instance by microservice name, or filter instances by environment and application.

----End

Changing the Instance Status

Status indicates the status of a microservice instance.

NOTE

The status of microservice instances synchronized by binding microservice engines cannot be changed during component creation and deployment by referring to **Creating and Deploying a Component**.

The following table describes the microservice instance statuses.

Status	Description
Online	The instance is running and can provide services.
Offline	Before the instance process ends, the instance is marked as not providing services externally.
Out of Service	The instance has been registered with the microservice engine and does not provide services.
Testing	The instance is in the internal joint commissioning state and does not provide services.

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 Choose **Microservice Catalog**.

- For microservice engines with security authentication disabled, go to **Step 5**.
- For microservice engines with security authentication enabled, go to **Step 4**.

Step 4 In the displayed **Security Authentication** dialog box, enter the account name and password, and click **OK**.

NOTE

- If you connect to the microservice engine for the first time, enter the account name **root** and the password entered when **Creating a Microservice Engine**.
- For details about how to create an account, see **Adding an Account**.

Step 5 Click **Instance List**, select the target instance, and change the instance status.

- Offline

  In the **Operation** column, click **Offline**.
- Online

  In the **Operation** column, choose **More** > **Online**.
- Out of Service

  In the **Operation** column, choose **More** > **Out of Service**.
- Testing

  In the **Operation** column, choose **More** > **Testing**.

----End

8.4.3 Microservice Governance

8.4.3.1 Overview

If an application is developed using the microservice framework, the microservice is automatically registered with the corresponding microservice engine after the application is managed and started. You can perform service governance on the engine console by referring to **Governing Microservices**.

NOTE

This function is supported by microservice engine 1.x and 2.4.0 and later versions.

8.4.3.2 Governing Microservices

After a microservice is deployed, you can govern it based on its running statuses.

Prerequisites

- You can create a microservice in **Microservice List** from **Service Catalog** and start the microservice. After the microservice starts, the service instance is registered under the corresponding service based on configurations in the **.yaml** file.
- If the microservice is not created in advance or has been deleted, the microservice is automatically created when the service instance is registered.
- After a microservice is created, register the service instance before performing the corresponding operation.

Governance Policies

You can configure the following policies: Load Balancing, Rate Limiting, Fault Tolerance, Service Degradation, Circuit Breaker, Fault Injection, and Blacklist and Whitelist. For details, see the following table.

Name	Description
Load Balancing	• Application scenario Generally, multiple instances are deployed for a microservice. Load balancing controls the policy for a microservice consumer to access multiple instances of a microservice provider to balance traffic. It includes polling, random, response time weigh, and session stickiness. • For details about the configuration example of the governance policy and how to add dependencies to POM, see **Load Balancing**.
Rate Limiting	• Application scenario This policy controls the number of requests for accessing microservices to prevent the system from being damaged due to traffic impact. • For details about the configuration example of the governance policy and how to add dependencies to the POM, see **Rate Limiting**.

Name	Description
Service Degradation	• Application scenario When a microservice invokes other microservices, the default value is forcibly returned or an exception is thrown instead of sending the request to the target microservice. In this way, the access to the target microservice is shielded and the pressure on the target microservice is reduced. • For details about the configuration example of the governance policy and how to add dependencies to the POM, see **Service Degradation**.
Fault Tolerance	• Application scenario If an exception occurs when a microservice consumer accesses a provider, for example, the instance network is disconnected, the request needs to be forwarded to another available instance. Fault tolerance is often referred to as retry. • For details about the configuration example of the governance policy and how to add dependencies to POM, see **Fault Tolerance**.
Circuit Breaker	• Application scenario If an exception occurs when a microservice consumer accesses a provider, for example, the instance network is disconnected or the request times out, and the exception accumulates to a certain extent, the consumer needs to stop accessing the provider and return an exception or a default value to prevent the avalanche effect. Automatic circuit breaker is supported, which determines a circuit breaker according to the error rate. • For details about the configuration example and how to add dependencies to the POM, see **Circuit Breaker**.
Fault Injection	• Application scenario Fault injection can simulate an invoking failure, which is mainly used for function verification and fault scenario demonstration. • Governance of microservices connected to the Java Chassis development framework. For details about the configuration example of the governance policy and how to add dependencies to POM, see **Fault Injection**. **NOTE** This policy applies only to microservices accessed through Java chassis.

Name	Description
Blacklist and Whitelist	• Application scenario Based on the public key authentication mechanism, microservice engines provide the blacklist and whitelist functions. The blacklist and whitelist can be used to control which services can be accessed by microservices. • Governance of microservices accessed through Java chassis The blacklist and whitelist take effect only after public key authentication is enabled. For details, see **Configuring Public Key Authentication**. **NOTE** This policy applies only to microservices accessed through Java chassis.

Configuring Load Balancing

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 Choose **Microservice Governance**.

- For microservice engines with security authentication disabled, go to **Step 5**.
- For microservice engines with security authentication enabled, go to **Step 4**.

Step 4 In the displayed **Security Authentication** dialog box, enter the account name and password, and click **OK**.

NOTE

- If you connect to the microservice engine for the first time, enter the account name **root** and the password entered when **Creating a Microservice Engine**.
- For details about how to create an account, see **Adding an Account**.

Step 5 Click the microservice to be governed.

Step 6 Choose **Load Balancing**.

Step 7 Click **New**. Select the microservices to be governed and select a proper load balancing policy. For details, see the following table.

Policy	Description
Round robin	Supports routes according to the location information about service instances.
Random	Provides random routes for service instances.

Policy	Description
Response time weight	Provides weight routes with the minimum active number (latency) and supports service instances with slow service processing in receiving a small number of requests to prevent the system from stopping response. This load balancing policy is suitable for applications with low and stable service requests. **NOTE** This policy applies to microservices accessed through Java chassis.
Session stickiness	Provides a mechanism on the load balancer. In the specified session stickiness duration, this mechanism allocates the access requests related to the same user to the same instance. • **Stickiness Duration**: time limit for keeping a session. The value ranges from 0 to 86400, in seconds. • **Failures**: number of access failures. The value ranges from 0 to 10. If the upper limit of failures or the session stickiness duration exceeds the specified values, the microservice stops accessing this instance. **NOTE** This policy applies to microservices accessed through Java chassis.

Step 8 Click **OK**.

----End

Configuring Rate Limiting

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 Choose **Microservice Governance**.

- For microservice engines with security authentication disabled, go to **Step 5**.
- For microservice engines with security authentication enabled, go to **Step 4**.

Step 4 In the displayed **Security Authentication** dialog box, enter the account name and password, and click **OK**.

NOTE

- If you connect to the microservice engine for the first time, enter the account name **root** and the password entered when **Creating a Microservice Engine**.
- For details about how to create an account, see **Adding an Account**.

Step 5 Click the microservice to be governed.

Step 6 Click **Rate Limiting**.

Step 7 Click **New**. The following table describes configuration items of rate limiting.

Configuration Item	Description	Value Range
Rate Limiting Object	Other microservices that access the microservice. **NOTE** This configuration applies to microservices accessed through Java chassis.	Select an item from the drop-down list next to **Rate Limiting Object**.
Upstream Microservice	Configure rate limiting for the upstream microservice to invoke the service. **NOTE** This configuration applies to microservices accessed through Spring Cloud.	Select an item from the drop-down list next to **Upstream Microservice**.
QPS	Requests generated per second. When the number of requests sent by the rate limiting object to the current service instance exceeds the specified value, the current service instance no longer accepts requests from the rate limiting object.	Enter an integer ranging from 1 to 99999.

NOTE

If a microservice has three instances, the rate limiting of each instance is set to 2700 QPS, then the total QPS is 8100, and rate limiting is triggered only when the QPS exceeds 8100.

Step 8 Click **OK**.

----End

Configuring Service Degradation

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 Choose **Microservice Governance**.

- For microservice engines with security authentication disabled, go to **Step 5**.
- For microservice engines with security authentication enabled, go to **Step 4**.

Step 4 In the displayed **Security Authentication** dialog box, enter the account name and password, and click **OK**.

NOTE

- If you connect to the microservice engine for the first time, enter the account name **root** and the password entered when **Creating a Microservice Engine**.
- For details about how to create an account, see **Adding an Account**.

Step 5 Click the microservice to be governed.

Step 6 Click **Service Degradation**.

Step 7 Click **New** and select a proper policy. The following table describes the configuration items of service degradation.

Configuration Item	Description
Fallback Object	Microservice to be degraded.
Request Path	Click  and set **Method**, **Path**, and **Headers** to specify the request path. **NOTE** This configuration applies to microservices accessed through Spring Cloud.
Fallback	• Open • Close

Step 8 Click **OK**.

----End

Configuring Fault Tolerance

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 Choose **Microservice Governance**.

- For microservice engines with security authentication disabled, go to **Step 5**.
- For microservice engines with security authentication enabled, go to **Step 4**.

Step 4 In the displayed **Security Authentication** dialog box, enter the account name and password, and click **OK**.

NOTE

- If you connect to the microservice engine for the first time, enter the account name **root** and the password entered when **Creating a Microservice Engine**.
- For details about how to create an account, see **Adding an Account**.

Step 5 Click the microservice to be governed.

Step 6 Click **Fault Tolerance**.

Step 7 Click **New** and select a proper policy. The following table describes the configuration items of fault tolerance.

Configuration Item	Description
Downstream Microservice	Configure fault tolerance for the microservice to invoke the downstream microservice. You can select a value from the drop-down list. **NOTE** This configuration applies to microservices accessed through Spring Cloud.
Fault Tolerance Object	Microservice or method that the application relies on. **NOTE** This configuration applies to microservices accessed through Java chassis.
Fault Tolerance	**Open**: The system processes a request sent to the fault tolerance object based on the selected fault tolerance policy when the request encounters an error. **Close**: The system waits until the timeout interval expires and then returns the failure result even though the service request fails to be implemented.
FT Policy	This parameter is mandatory when **Fault Tolerance** is set to **Open**. For microservices accessed through Spring Cloud, set the following parameters: • Number of attempts to the same microservice instance • Number of attempts to the new microservice instance For microservices accessed through Java chassis, set the following parameters: • Failover The system attempts to reestablish connections on different servers. • Failfast The system does not attempt to reestablish a connection. After a request fails, a failure result is returned immediately. • Failback The system attempts to reestablish connections on the same server. • custom – Number of attempts to reestablish connections on the same server – Number of attempts to reestablish connections on new servers

Step 8 Click **OK**.

----End

Configuring Circuit Breaker

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 Choose **Microservice Governance**.

- For microservice engines with security authentication disabled, go to **Step 5**.
- For microservice engines with security authentication enabled, go to **Step 4**.

Step 4 In the displayed **Security Authentication** dialog box, enter the account name and password, and click **OK**.

NOTE

- If you connect to the microservice engine for the first time, enter the account name **root** and the password entered when **Creating a Microservice Engine**.
- For details about how to create an account, see **Adding an Account**.

Step 5 Click the microservice to be governed.

Step 6 Click **Circuit Breaker**.

Step 7 Click **New** and select a proper policy. The following table describes the configuration items of circuit breaker.

Configuration Item	Description
Downstream Microservice	Configure circuit breaker for the microservice to invoke the downstream microservice. **NOTE** This configuration applies to microservices accessed through Spring Cloud.
Circuit Breaker Object	Microservice or method invoked by the application. **NOTE** This configuration applies to microservices accessed through Java chassis.
Request Path	Click  and set **Method**, **Path**, and **Headers** to specify the request path. **NOTE** This configuration applies to microservices accessed through Spring Cloud.
Triggering Condition	• **Circuit Breaker Time Window**: circuit breaker duration. The system does not respond to requests within this time window. • **Request Failure Rate**: failure rate of window requests. • **Window Requests**: number of requests received by the window. Circuit breaker is triggered only when **Request Failure Rate** and **Window Requests** both reach their thresholds.

Step 8 Click **OK**.

----End

Configuring Fault Injection

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 Choose **Microservice Governance**.

- For microservice engines with security authentication disabled, go to **Step 5**.
- For microservice engines with security authentication enabled, go to **Step 4**.

Step 4 In the displayed **Security Authentication** dialog box, enter the account name and password, and click **OK**.

NOTE

- If you connect to the microservice engine for the first time, enter the account name **root** and the password entered when **Creating a Microservice Engine**.
- For details about how to create an account, see **Adding an Account**.

Step 5 Click the microservice to be governed.

Step 6 Click **Fault Injection**.

Step 7 Click **New** and select a proper policy. The following table describes the configuration items of fault injection.

Configuration Item	Description
Injection Object	Microservices for which fault injection is required. You can specify a method for this configuration item.
Type	Type of the fault injected to the microservice. • Delayed • Fault
Protocol	Protocol for accessing the microservice when latency or fault occurs. • Rest • Highway
Occurrence Probability	Probability of latency or fault occurrence.
Delay Time	Duration of the latency during microservice access. This parameter is required when **Type** is set to **Delayed**.

Configuration Item	Description
HTTP Error Code	HTTP error code during microservice access. This parameter is required when **Type** is set to **Fault**. This error code is an HTTP error code.

Step 8 Click **OK**.

----End

Configuring Blacklist and Whitelist

Based on the public key authentication mechanism, microservice engines provide the blacklist and whitelist functions. The blacklist and whitelist can be used to control which services can be accessed by microservices.

The blacklist and whitelist take effect only after public key authentication is enabled. For details, see **Configuring Public Key Authentication**.

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 Choose **Microservice Governance**.

- For microservice engines with security authentication disabled, go to **Step 5**.
- For microservice engines with security authentication enabled, go to **Step 4**.

Step 4 In the displayed **Security Authentication** dialog box, enter the account name and password, and click **OK**.

NOTE

- If you connect to the microservice engine for the first time, enter the account name **root** and the password entered when **Creating a Microservice Engine**.
- For details about how to create an account, see **Adding an Account**.

Step 5 Click the microservice to be governed.

Step 6 Click **Black and white list**.

Step 7 Click **New** to add a blacklist or whitelist for the application. The following table describes configuration items of blacklist and whitelist.

Configuration Item	Description
Type	• **Blacklist**: Microservices that match the matching rule are not allowed to access the current service. • **Whitelist**: Microservices that match the matching rule are allowed to access the current service.

Configuration Item	Description
Rule	Use a regular expression. For example, if **Rule** is set to **data***, services whose names start with **data** in the blacklist are not allowed to access the current service, or services whose names start with **data** in the whitelist are allowed to access the current service.

Step 8 Click **OK**.

----End

Configuring Public Key Authentication

Public key authentication is a simple and efficient authentication mechanism between microservices provided by CSE. Its security is based on the reliable interaction between microservices and the service center. That is, the authentication mechanism must be enabled between microservices and the service center. The procedure is as follows:

1. When the microservice starts, a key pair is generated and the public key is registered with the service center.
2. Before accessing the provider, the consumer uses its own private key to sign a message.
3. The provider obtains the public key of the consumer from the service center and verifies the signed message.

To enable public key authentication, perform the following steps:

1. Enable public key authentication for both the consumer and provider.
```
servicecomb:
  handler:
    chain:
      Consumer:
        default: auth-consumer
      Provider:
        default: auth-provider
```
2. Add the following dependency to the **pom.xml** file:
```
<dependency>
    <groupId>org.apache.servicecomb</groupId>
    <artifactId>handler-publickey-auth</artifactId>
  </dependency>
```

8.4.4 Configuration Management (Applicable to Engine 2.x)

Microservice engines define a configuration mechanism that is irrelevant to development frameworks. A configuration item consists of a key, label, and value. The label is used to identify whether a configuration item belongs to global configuration or microservice configuration. The label can also indicate the value type.

You can refer to the following table to select the operations to be performed.

Operation	Description
Creating an Application-Level Configuration	Associates the new configuration with an application, and adds the application name and environment label.
Creating a Microservice-Level Configuration	Associates the new configuration with a microservice, and adds the microservice name, application name, and environment.
Creating a Customized Configuration Item	If application-level and microservice-level configurations cannot meet service requirements, you can customize configuration files.
Importing Configurations	Imports the local configuration file.
Exporting Configurations	Exports the selected configuration file to the local host.
Comparing Configuration Versions	Compares differences between historical versions.
Rolling Back a Version	Rolls back to the selected historical version.
Viewing Historical Versions	Displays configurations of different historical versions.
Editing a Configuration Item	Edits a configuration item.
Disabling a Configuration Item	Disables a configuration item.

Operation	Description
Deleting a Configuration Item	Deletes a configuration item.

NOTE

When the configuration item quota specified by the engine specifications is about to be used up, the engine allows new configuration items that exceed the remaining quota to be created to ensure capacity availability. Expand the capacity of the engine as soon as possible to prevent configuration creation failures.

Creating an Application-Level Configuration

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 Choose **Configuration Management**.

- For microservice engines with security authentication disabled, go to **Step 5**.
- For microservice engines with security authentication enabled, go to **Step 4**.

Step 4 In the displayed **Security Authentication** dialog box, enter the account name and password, and click **OK**.

NOTE

- If you connect to the microservice engine for the first time, enter the account name **root** and the password entered when **Creating a Microservice Engine**.
- For details about how to create an account, see **Adding an Account**.

Step 5 Click **Create Configuration Item** and set parameters by referring to the following table. Parameters marked with an asterisk (*) are mandatory.

Parameter	Description
*Configuration Item	Enter a configuration item. The configuration item is the global ID of the configuration. In the coding phase, the configuration item is used to index and operate the configuration. You are advised to use the Java package naming rule (for example, **cse.service.registry.address**) to ensure the readability and uniqueness of the configuration. **NOTE** Configuration items starting with **servicecomb.matchGroup.** cannot be created during application-level configuration creation. Such configuration items conflict with the configuration generated during service scenario governance creation, so the service scenario cannot be displayed.

Parameter	Description
Configuration Range	Select **Application-level**.
*Application	1. Select or enter an application name. 2. Select an environment.
Configuration Format	Select a configuration format. Common configuration formats such as TEXT, YAML, JSON, Properties, INI and XML can be edited online. The default value is **TEXT**.
*Configuration Content	Enter the configuration content.
Enable Configuration	Determine whether to enable the configuration item. • **Enable now**: The configuration item takes effect immediately once being created. • **Not Enabled**: The configuration item does not take effect.

Step 6 Click **Create Now** to enable the configuration item.

----End

Creating a Microservice-Level Configuration

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 Choose **Configuration Management**.

- For microservice engines with security authentication disabled, go to **Step 5**.
- For microservice engines with security authentication enabled, go to **Step 4**.

Step 4 In the displayed **Security Authentication** dialog box, enter the account name and password, and click **OK**.

NOTE

- If you connect to the microservice engine for the first time, enter the account name **root** and the password entered when **Creating a Microservice Engine**.
- For details about how to create an account, see **Adding an Account**.

Step 5 Click **Create Configuration Item** and set parameters by referring to the following table. Parameters marked with an asterisk (*) are mandatory.

Parameter	Description
*Configuration Item	Enter a configuration item. The configuration item is the global ID of the configuration. In the coding phase, the configuration item is used to index and operate the configuration. You are advised to use the Java package naming rule (for example, **cse.service.registry.address**) to ensure the readability and uniqueness of the configuration.
Configuration Range	Select **Microservice-level**.
*Microservice	1. Select or enter a microservice name. 2. Select or enter an application name. 3. Select an environment.
Configuration Format	Select a configuration format. Common configuration formats such as TEXT, YAML, JSON, Properties, INI and XML can be edited online. The default value is **TEXT**.
*Configuration Content	Enter the configuration content.
Enable Configuration	Determine whether to enable the configuration item. • **Enable now**: The configuration item takes effect immediately once being created. • **Not Enabled**: The configuration item does not take effect.

Step 6 Click **Create Now** to enable the configuration item.

----End

Creating a Customized Configuration Item

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 Choose **Configuration Management**.

- For microservice engines with security authentication disabled, go to **Step 5**.
- For microservice engines with security authentication enabled, go to **Step 4**.

Step 4 In the displayed **Security Authentication** dialog box, enter the account name and password, and click **OK**.

NOTE

- If you connect to the microservice engine for the first time, enter the account name **root** and the password entered when **Creating a Microservice Engine**.
- For details about how to create an account, see **Adding an Account**.

Step 5 Click **Create Configuration Item** and set parameters by referring to the following table. Parameters marked with an asterisk (*) are mandatory.

Parameter	Description
*Configuration Item	Enter a configuration item. The configuration item is the global ID of the configuration. In the coding phase, the configuration item is used to index and operate the configuration. You are advised to use the Java package naming rule (for example, **cse.service.registry.address**) to ensure the readability and uniqueness of the configuration.
Configuration Range	Select **Customize**.
*Labels	If application-level and microservice-level configurations cannot meet service requirements, you can use labels to customize configurations.
Configuration Format	Select a configuration format.
*Configuration Content	Enter configuration content. Common configuration formats such as TEXT, YAML, JSON, Properties, INI and XML can be edited online. The default value is **TEXT**.
Enable Configuration	Determine whether to enable the configuration item. • **Enable now**: The configuration item takes effect immediately once being created. • **Not Enabled**: The configuration item does not take effect.

Step 6 Click **Create Now** to enable the configuration item.

----End

Importing Configurations

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 Choose **Configuration Management**.

- For microservice engines with security authentication disabled, go to **Step 5**.
- For microservice engines with security authentication enabled, go to **Step 4**.

Step 4 In the displayed **Security Authentication** dialog box, enter the account name and password, and click **OK**.

NOTE

- If you connect to the microservice engine for the first time, enter the account name **root** and the password entered when **Creating a Microservice Engine**.
- For details about how to create an account, see **Adding an Account**.

Step 5 Click **Import** in the upper right corner and set parameters by referring to the following table.

Parameter	Description
Import to a specific environment	• Disabled: The imported configuration does not change the environment label. • Enabled: Importing the configuration to a specific environment will change the environment label.
Same Configuration	• **Terminate**: If a configuration is the same as that in the system, the import terminates. • **Skip**: During import, if a configuration is the same as that in the system, the configuration is skipped and other configurations are imported. • **Overwrite**: During import, if a configuration is the same as that in the system, the value of the configuration will be replaced.
Configuration File	Click **Import** and select the target file. **NOTE** The file size cannot exceed 2 MB.

Step 6 Click **Close**.

----End

Exporting Configurations

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 Choose **Configuration Management**.

- For microservice engines with security authentication disabled, go to **Step 5**.
- For microservice engines with security authentication enabled, go to **Step 4**.

Step 4 In the displayed **Security Authentication** dialog box, enter the account name and password, and click **OK**.

NOTE

- If you connect to the microservice engine for the first time, enter the account name **root** and the password entered when **Creating a Microservice Engine**.
- For details about how to create an account, see **Adding an Account**.

Step 5 Select the configuration items to be exported and click **Export**. In the displayed dialog box, click **Export**. Alternatively, click **Export All** in the upper right corner to export all configurations.

----End

Comparing Configuration Versions

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 Choose **Configuration Management**.

- For microservice engines with security authentication disabled, go to **Step 5**.
- For microservice engines with security authentication enabled, go to **Step 4**.

Step 4 In the displayed **Security Authentication** dialog box, enter the account name and password, and click **OK**.

NOTE

- If you connect to the microservice engine for the first time, enter the account name **root** and the password entered when **Creating a Microservice Engine**.
- For details about how to create an account, see **Adding an Account**.

Step 5 Click the configuration item to be compared.

Step 6 Click **View Historical Version**.

Step 7 In **Historical Versions** on the left, select the historical version to be viewed.

In the **Configuration file** on the right, you can view the differences between the current and historical versions.

----End

Rolling Back a Version

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 Choose **Configuration Management**.

- For microservice engines with security authentication disabled, go to **Step 5**.
- For microservice engines with security authentication enabled, go to **Step 4**.

Step 4 In the displayed **Security Authentication** dialog box, enter the account name and password, and click **OK**.

NOTE

- If you connect to the microservice engine for the first time, enter the account name **root** and the password entered when **Creating a Microservice Engine**.
- For details about how to create an account, see **Adding an Account**.

Step 5 Click the target configuration item.

Step 6 Click **View Historical Version**.

Step 7 In **Historical Versions** on the left, select the target historical version.

Step 8 In **Configuration file** on the right, click **Roll Back to the Selected Version**.

----End

Viewing Historical Versions

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 Choose **Configuration Management**.

- For microservice engines with security authentication disabled, go to **Step 5**.
- For microservice engines with security authentication enabled, go to **Step 4**.

Step 4 In the displayed **Security Authentication** dialog box, enter the account name and password, and click **OK**.

NOTE

- If you connect to the microservice engine for the first time, enter the account name **root** and the password entered when **Creating a Microservice Engine**.
- For details about how to create an account, see **Adding an Account**.

Step 5 Click **View Historical Version** in the **Operation** column of a configuration item. On the **Historical Versions** page that is displayed, you can view the historical versions of the configuration item. On this page, you can compare the configuration version with the rollback version.

----End

Editing a Configuration Item

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 Choose **Configuration Management**.

- For microservice engines with security authentication disabled, go to **Step 5**.
- For microservice engines with security authentication enabled, go to **Step 4**.

Step 4 In the displayed **Security Authentication** dialog box, enter the account name and password, and click **OK**.

NOTE

- If you connect to the microservice engine for the first time, enter the account name **root** and the password entered when **Creating a Microservice Engine**.
- For details about how to create an account, see **Adding an Account**.

Step 5 Click **Edit** in the **Operation** column of the target configuration item. You can also click the target configuration item and then click **Edit** on the displayed configuration details page.

Step 6 Enter the configuration information in the **Configuration Content** text box and click **Save**.

----End

Disabling a Configuration Item

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 Choose **Configuration Management**.

- For microservice engines with security authentication disabled, go to **Step 5**.
- For microservice engines with security authentication enabled, go to **Step 4**.

Step 4 In the displayed **Security Authentication** dialog box, enter the account name and password, and click **OK**.

NOTE

- If you connect to the microservice engine for the first time, enter the account name **root** and the password entered when **Creating a Microservice Engine**.
- For details about how to create an account, see **Adding an Account**.

Step 5 In the **Operation** column of the target configuration item, click **More** > **Disable**.

Step 6 Click **OK**.

----End

Deleting a Configuration Item

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 Choose **Configuration Management**.

- For microservice engines with security authentication disabled, go to **Step 5**.
- For microservice engines with security authentication enabled, go to **Step 4**.

Step 4 In the displayed **Security Authentication** dialog box, enter the account name and password, and click **OK**.

NOTE

- If you connect to the microservice engine for the first time, enter the account name **root** and the password entered when **Creating a Microservice Engine**.
- For details about how to create an account, see **Adding an Account**.

Step 5 Click **Delete** in the **Operation** column of the target configuration item. You can also click the target configuration item and then click **Delete** on the displayed configuration details page.

Step 6 Click **OK**.

----End

8.4.5 Configuration Management (Applicable to Engine 1.x)

The configuration added here is a global configuration. After being added, the configuration takes effect immediately if all microservices registered with the engine use it.

If dynamic configuration is set for a single microservice, the dynamic configuration overwrites the global configuration. For details about how to set dynamic configuration, see **Dynamic Configuration**.

Creating a Configuration

Configuration management provides common configurations for microservices, such as log levels and running parameters. After being added, the configuration item is used as the default one if no same configuration items are defined for microservices.

NOTICE

Configuration items are stored in plaintext. Do not include sensitive data.

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 Choose **Configuration Management**.

- For microservice engines with security authentication disabled, go to **Step 5**.
- For microservice engines with security authentication enabled, go to **Step 4**.

Step 4 In the displayed **Security Authentication** dialog box, enter the account name and password, and click **OK**.

NOTE

- If you connect to the microservice engine for the first time, enter the account name **root** and the password entered when **Creating a Microservice Engine**.
- For details about how to create an account, see **Adding an Account**.

Step 5 Click **Create Configuration Item**.

Step 6 On the **Create Configuration Item** page, select a microservice environment and enter **Configuration Item** and **Value**.

Step 7 Click **OK**.

----End

Importing Configurations

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 Choose **Configuration Management**.

- For microservice engines with security authentication disabled, go to **Step 5**.
- For microservice engines with security authentication enabled, go to **Step 4**.

Step 4 In the displayed **Security Authentication** dialog box, enter the account name and password, and click **OK**.

NOTE

- If you connect to the microservice engine for the first time, enter the account name **root** and the password entered when **Creating a Microservice Engine**.
- For details about how to create an account, see **Adding an Account**.

Step 5 Click **Import**.

Step 6 Select a microservice environment, click **Import**, and select the target file.

NOTE

A maximum of 150 configuration items can be imported at a time.

Step 7 Click **Close**.

----End

Exporting Configurations

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 Choose **Configuration Management**.

- For microservice engines with security authentication disabled, go to **Step 5**.
- For microservice engines with security authentication enabled, go to **Step 4**.

Step 4 In the displayed **Security Authentication** dialog box, enter the account name and password, and click **OK**.

NOTE

- If you connect to the microservice engine for the first time, enter the account name **root** and the password entered when **Creating a Microservice Engine**.
- For details about how to create an account, see **Adding an Account**.

Step 5 Click **Export All**.

----End

Deleting a Configuration

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 Choose **Configuration Management**.

- For microservice engines with security authentication disabled, go to **Step 5**.
- For microservice engines with security authentication enabled, go to **Step 4**.

Step 4 In the displayed **Security Authentication** dialog box, enter the account name and password, and click **OK**.

NOTE

- If you connect to the microservice engine for the first time, enter the account name **root** and the password entered when **Creating a Microservice Engine**.
- For details about how to create an account, see **Adding an Account**.

Step 5 Select the target configuration item and click **Delete**. You can also click **Delete** in the **Operation** column of the target configuration item.

Step 6 Click **OK**.

----End

Editing a Configuration

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 Choose **Configuration Management**.

- For microservice engines with security authentication disabled, go to **Step 5**.
- For microservice engines with security authentication enabled, go to **Step 4**.

Step 4 In the displayed **Security Authentication** dialog box, enter the account name and password, and click **OK**.

NOTE

- If you connect to the microservice engine for the first time, enter the account name **root** and the password entered when **Creating a Microservice Engine**.
- For details about how to create an account, see **Adding an Account**.

Step 5 Click **Edit** in the **Operation** column of the target configuration item and edit the values of the configuration item.

Step 6 Click **OK**.

----End

8.4.6 System Management

8.4.6.1 Overview

A microservice engine may be used by multiple users. Different users must have different microservice engine access and operation permissions based on their responsibilities and permissions.

The exclusive microservice engine with security authentication enabled provides the system management function using the role-based access control (RBAC) through the microservice console.

The exclusive microservice engine with security authentication enabled supports the access of Spring Cloud and Java chassis microservice frameworks.

NOTE

- The RBAC-based system management function is irrelevant to IAM permission management. It is only an internal permission management mechanism of CSE.
- To operate a microservice engine on CSE, you must have both the IAM and RBAC permissions, and the IAM permission takes precedence over the RBAC permission.
- If you perform operations on a microservice engine through APIs or the microservice framework, you only need to have the RBAC permissions.

1. You can use an account associated with the **admin** role to create an account and associate a proper role with the account based on service requirements. The user who uses this account has the access and operation permissions on the microservice engine.
   - When you create an exclusive microservice engine with security authentication enabled, the system automatically creates the **root** account associated with the **admin** role. The **root** account cannot be edited or deleted.
   - You can create an account using the **root** account of the microservice engine or an account associated with the **admin** role of the microservice engine. For details about how to create and manage an account, see **Accounts**.
2. You can create a custom role using an account associated with the **admin** role and grant proper microservice engine access and operation permissions to the role based on service requirements.
   - The system provides two default roles: administrator (**admin**) and developer (**developer**). Default roles cannot be edited or deleted.
   - You can create a custom role using the **root** account of the microservice engine or an account associated with the **admin** role of the microservice engine. For details about how to create and manage a role, see **Roles**.
   - For details about role permissions, see **Table 8-5**.

Table 8-5 Role permissions

Role	Permission Description
Admin	Full permissions for all microservices, accounts, and roles of the microservice engine.
Developer	Full permissions for all microservices of the microservice engine.

Role	Permission Description
Custom role	You can create roles based on service requirements and grant microservice operation and configuration permissions to the roles.

8.4.6.2 Accounts

You can use an account associated with the **admin** role to log in to the microservice engine console and create an account or manage a specified account created in the engine based on service requirements.

Table 8-6 Account management operations

Operation	Description
Adding an Account	Creates an account and associates a proper role with the account. Users who use the account have the access and operation permissions on the microservice engine. You can create up to 1000 accounts, including new accounts and imported IAM account.
Importing an IAM Account	Imports an IAM account and associates roles with it. Users using this IAM account have the access and operation permissions on the microservice engine. If the imported IAM account needs to connect microservice applications to the engine through programming interface authentication, **reset** its password and then use the new password to configure security authentication parameters. When you use this IAM account to log in to the ServiceStage console with security authentication enabled, you do not need to enter the account and password. However, a password is still required after the VDC read-only user is imported. ServiceStage can manage up to 1000 accounts, including new accounts and imported IAM account. **NOTE** The IAM account can be imported only to microservice engine 2.5.0 or later with security authentication enabled.
Viewing Role Permissions	Displays the permissions of the role associated with a specified account.
Editing an Account	Adds or deletes roles for an account. The **root** account cannot be edited.

Operation	Description

Changing the Password	Changes the password of an account that has logged in to the engine based on service requirements or security regulations.
	NOTICE
	• If the account and password are used to register a microservice in the SDK, changing the account and password may affect the service running of the microservice (the microservice cannot be registered with the microservice engine). As a result, the service system will be damaged. Exercise caution when performing this operation.
	• After the password is changed, update the microservice authentication configuration in a timely manner.
	  • Spring Cloud: see **Connecting Spring Cloud Applications to CSE Engines**.
	  • Java Chassis: see **Connecting Java Chassis Applications to CSE Engines**.
	• After the password is changed, the account may be locked due to three consecutive incorrect password attempts. The account will be unlocked after 15 minutes.

Resetting a Password	Based on service requirements or security regulations, you can use the account that has logged in to the microservice engine to reset the passwords of other accounts under the microservice engine.
	NOTICE
	• If the account and password are used to register a microservice in the SDK, resetting the account and password may affect the service running of the microservice (the microservice cannot be registered with the microservice engine). As a result, the service system will be damaged. Exercise caution when performing this operation.
	• After the password is reset, update the microservice authentication configuration in a timely manner.
	  • Spring Cloud: see **Connecting Spring Cloud Applications to CSE Engines**.
	  • Java Chassis: see **Connecting Java Chassis Applications to CSE Engines**.
	• After the password is reset, the account may be locked due to three consecutive incorrect password attempts. The account will be unlocked after 15 minutes.

Deleting an Account	Deletes an account that is no longer used. The **root** account cannot be deleted. **NOTICE** If the account and password are used to register a service in the SDK, deleting the account will affect the service running (the account cannot be registered with the engine) and damage the service system. Exercise caution when performing this operation.

Adding an Account

Before adding an account, you can create a role based on service requirements. For details, see **Creating a Role**.

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine with security authentication enabled from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 Choose **System Management**.

Step 4 In the displayed **Security Authentication** dialog box, enter the name and password of the account associated with the **admin** role under the microservice engine, and click **OK**.

NOTE

If you connect to the microservice engine for the first time, enter the account name **root** and the password entered when **Creating a Microservice Engine**.

Step 5 Choose **Accounts** > **Create Account** and configure account parameters by referring to the following table:

Parameter	Description
Account	Enter an account name. **NOTE** The account name cannot be changed once the account is created.
Role	Select a role based on service requirements. **NOTE** An account can be associated with up to five roles.
Password	Enter the password.
Confirm Password	Enter the password again.

Step 6 Click **OK**.

----End

Importing an IAM Account

Before importing an IAM account, you can create a role based on service requirements. For details, see **Creating a Role**.

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 Choose **System Management**. In the displayed **Security Authentication** dialog box, enter the name and password of the account associated with the **admin** role under the microservice engine, and click **OK**.

NOTE

- If you connect to the microservice engine for the first time, enter the account name **root** and the password entered when **Creating a Microservice Engine**.
- For details about how to create an account, see **Adding an Account**.

Step 4 Choose **Accounts** > **Import IAM User Name**.

Step 5 Select the IAM account to be imported and select account roles.

NOTE

An account can be associated with up to five roles.

Step 6 Click **Confirm**.

NOTE

You cannot use the passwords of the imported IAM user names to log in to the system. **Resetting a Password** first if you want to use the imported IAM user names to connect microservice applications to the engine through programming interface security authentication.

----End

Viewing Role Permissions

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine with security authentication enabled from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 Choose **System Management**.

Step 4 In the displayed **Security Authentication** dialog box, enter the name and password of the account associated with the **admin** role under the microservice engine, and click **OK**.

NOTE

- If you connect to the microservice engine for the first time, enter the account name **root** and the password entered when **Creating a Microservice Engine**.
- For details about how to create an account, see **Adding an Account**.

Step 5 Click the role in the **Role** column of the account to be viewed in the account list. On the displayed page, view the role and permission configuration associated with the account.

----End

Editing an Account

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine with security authentication enabled from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 Choose **System Management**.

Step 4 In the displayed **Security Authentication** dialog box, enter the name and password of the account associated with the **admin** role under the microservice engine, and click **OK**.

NOTE

- If you connect to the microservice engine for the first time, enter the account name **root** and the password entered when **Creating a Microservice Engine**.
- For details about how to create an account, see **Adding an Account**.

Step 5 On the **Accounts** tab page, click **Edit Account** in the **Operation** column of the account to be edited.

Step 6 Select a role based on service requirements.

NOTE

An account can be associated with up to five roles.

Step 7 Click **Save**.

----End

Changing the Password

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine with security authentication enabled from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 Choose **System Management**.

Step 4 In the displayed **Security Authentication** dialog box, enter the account name and password, and click **OK**.

NOTE

- If you connect to the microservice engine for the first time, enter the account name **root** and the password entered when **Creating a Microservice Engine**.
- The account for connecting to the microservice engine is not associated with the admin role. You can only change the password of the current login account.
- The account for connecting to the microservice engine is associated with the admin role. You can change the passwords of all accounts of the microservice engine.
- For details about how to create an account, see **Adding an Account**.

Step 5 On the **Accounts** tab, select the account for logging in to the microservice engine and click **Reset Own Password** in the **Operation** column.

1. Enter the old password and a new password, and confirm the password.
2. After confirming that the password needs to be changed, select **I Understand**.

NOTE

You can also click **Reset Own Password** in the upper right corner of the **System Management** page to change the password of the current login account.

Step 6 Click **Save**.

----End

Resetting a Password

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine with security authentication enabled from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 Choose **System Management**.

Step 4 In the displayed **Security Authentication** dialog box, enter the name and password of the account associated with the **admin** role under the microservice engine, and click **OK**.

NOTE

- If you connect to the microservice engine for the first time, enter the account name **root** and the password entered when **Creating a Microservice Engine**.
- For details about how to create an account, see **Adding an Account**.

Step 5 On the **Accounts** tab page, select the account whose password is to be reset, and click **Reset Password** in the **Operation** column.

1. Enter a new password and confirm the password.
2. After confirming that the password needs to be reset, select **I Understand**.

Step 6 Click **Save**.

----End

Deleting an Account

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine with security authentication enabled from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 Choose **System Management**.

Step 4 In the displayed **Security Authentication** dialog box, enter the name and password of the account associated with the **admin** role under the microservice engine, and click **OK**.

NOTE

- If you connect to the microservice engine for the first time, enter the account name **root** and the password entered when **Creating a Microservice Engine**.
- For details about how to create an account, see **Adding an Account**.

Step 5 On the **Accounts** tab page, click **Delete** in the **Operation** column of the account to be deleted.

Step 6 In the displayed dialog box, enter **DELETE** and click **OK**.

----End

8.4.6.3 Roles

In addition to the default roles **admin** and **developer**, you can use a microservice engine account associated with the **admin** role to log in to the CSE console and perform operations listed in **Table 8-7** based on service requirements.

Table 8-7 Role management operations

Operation	Description
Creating a Role	Creates a role and configures permission actions for the role in different service and configuration groups. A maximum of 100 roles can be created.
Editing a Role	Modifies the permissions of the created role.
Deleting a Role	Deletes a role that is no longer used. **NOTE** • Deleted roles cannot be restored. Exercise caution when performing this operation. • Before deleting a role, ensure that the role is not associated with any account. For details about how to cancel the association between a role and an account, see **Editing an Account**.
Viewing a Role	Displays the created roles of the microservice engine based on the keyword of the role name.

Creating a Role

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine with security authentication enabled from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 Choose **System Management**.

Step 4 In the displayed **Security Authentication** dialog box, enter the name and password of the account associated with the **admin** role under the microservice engine, and click **OK**.

NOTE

- If you connect to the microservice engine for the first time, enter the account name **root** and the password entered when **Creating a Microservice Engine**.
- For details about how to create an account, see **Adding an Account**.

Step 5 On the **Roles** tab page, click **Create Role**.

Step 6 Enter a role name.

NOTE

The role name cannot be changed once the role is created.

Step 7 Configure permissions.

1. Set **Permission Group**.
   a. Set the service permissions.
      - If you select **All Services**:
        You can perform corresponding permission actions on all microservices of the microservice engine.
      - If you select **Custom Service Groups**, set the parameters according to **Table 8-8**.

Table 8-8 Custom service group operations

Operation	Description
Adding a Matching Rule	Click **Add Service Group Matching Rule**. Select **Application**, **Environment**, and **Service** based on service requirements to filter the microservices on which the role can perform permission actions. **NOTE** **Application**, **Environment**, and **Service** are three parameters of a microservice: ○ If only one parameter is set for a single matching rule, the role has the operation permission on the microservice that matches the parameter value. For example, if you add **Environment: production**, the role has the operation permission only on the microservice whose environment name is **production**. ○ If more than one parameter is set for a single matching rule, the role has the operation permission on the microservices that match all parameter values. For example, if you add **Environment: production** **Application: abc**, the role has the operation permission on the microservice whose environment name is **production** and application name is **abc**. ○ When automatic discovery is enabled, microservices query the instance addresses of services such as the registry center, configuration center, and dashboard through the registry center. When you grant the query permission to a microservice, the permission of the default application must be included. In this case, add the matching rule **Application: default**. After the microservice matching rule is set, click **OK**.
Editing a Matching Rule	Click  next to the matching rule to be edited. You can reconfigure **Service Group** and **Action** of the matching rule based on service requirements. After the service group matching rule is set, click **OK**.

Operation	Description
Deleting a Matching Rule	Click  next to the matching rule to be deleted. You can delete the matching rule based on service requirements.

NOTE

A maximum of 20 microservice matching rules can be set for a custom service group.

If multiple matching rules are set for a custom service group, the role has the operation permission on the microservice as long as the microservice meets any of the matching rules.

b. Set the configuration permissions.

- If you select **All Configurations**:

  You can perform corresponding permission actions on all microservices of the microservice engine.

- If you select **Custom Configuration Groups**, set the parameters according to **Table 8-9**.

Table 8-9 Custom configuration group operations

Operation	Description
Adding a Matching Rule	Click **Add Configuration Group Matching Rule**. Select **Application**, **Environment**, and **Service** based on service requirements to filter the configurations on which the role can perform permission actions. If application-level and microservice-level configurations cannot meet service requirements, you can customize a matching rule to match the configured custom labels and filter the permission actions that can be performed by the role. **NOTE** **Application**, **Environment**, and **Service** are three parameters of a configuration: ○ If only one parameter is set for a single matching rule, the role has the operation permission on the configuration that matches the parameter value. For example, if you add **Environment: production**, the role has the operation permission only on the configuration whose environment name is **production**. ○ If more than one parameter is set for a single matching rule, the role has the operation permission on the configurations that match all parameter values. For example, if you add **Environment: production** **Application: abc**, the role has the operation permission on the configuration whose environment name is **production** and application name is **abc**. After the configuration matching rule is set, click **OK**.
Editing a Matching Rule	Click  next to the matching rule to be edited. You can reconfigure **Configuration Group** and **Action** of the matching rule based on service requirements. After the configuration group matching rule is set, click **OK**.
Deleting a Matching Rule	Click  next to the matching rule to be deleted. You can delete the matching rule based on service requirements.

NOTE

A maximum of 20 matching rules can be set for a custom configuration group.

If multiple matching rules are set for a configuration service group, the role has the operation permission on the configuration as long as the configuration meets any of the matching rules.

2. Set **Action**.

Configure the permission actions that can be performed by the role on the selected service group and configuration group based on service requirements. You can select multiple permission actions.

- **All**: Add, delete, modify, and query resources in the service group and configuration group.
- **Add**: Add resources to the service group and configuration group.
- **Delete**: Delete resources from the service group and configuration group.

  NOTE

  If only **Delete** is selected, you cannot delete resources in the service group and configuration group. You must select **View** at the same time.
- **Modify**: Modify resources in the service group.

  NOTE

  If only **Modify** is selected, you cannot modify resources in the service group and configuration group. You must select **View** at the same time.
- **View**: View resources in the service group and configuration group.

Step 8 Click **Create**.

----End

Editing a Role

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine with security authentication enabled from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 Choose **System Management**.

Step 4 In the displayed **Security Authentication** dialog box, enter the name and password of the account associated with the **admin** role under the microservice engine, and click **OK**.

NOTE

- If you connect to the microservice engine for the first time, enter the account name **root** and the password entered when **Creating a Microservice Engine**.
- For details about how to create an account, see **Adding an Account**.

Step 5 On the **Roles** tab page, click **Edit** in the **Operation** column of the role to be edited.

Step 6 Modify **Service Group**, **Configuration Group**, and **Action** based on service requirements.

Step 7 Click **Save**.

----End

Deleting a Role

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine with security authentication enabled from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 Choose **System Management**.

Step 4 In the displayed **Security Authentication** dialog box, enter the name and password of the account associated with the **admin** role under the microservice engine, and click **OK**.

NOTE

- If you connect to the microservice engine for the first time, enter the account name **root** and the password entered when **Creating a Microservice Engine**.
- For details about how to create an account, see **Adding an Account**.

Step 5 On the **Roles** tab page, click **Delete** in the **Operation** column of the role to be deleted. In the displayed dialog box, enter **DELETE** and click **OK**.

NOTE

- Deleted roles cannot be restored. Exercise caution when performing this operation.
- Before deleting a role, ensure that the role is not associated with any account. For details about how to cancel the association between a role and an account, see **Editing an Account**.

----End

Viewing a Role

Step 1 Log in to ServiceStage and choose **Cloud Service Engine** > **Engines**.

Step 2 Select the target microservice engine with security authentication enabled from the **Microservice Engine** drop-down list in the upper part of the page.

Step 3 Choose **System Management**.

Step 4 In the displayed **Security Authentication** dialog box, enter the name and password of the account associated with the **admin** role under the microservice engine, and click **OK**.

NOTE

- If you connect to the microservice engine for the first time, enter the account name **root** and the password entered when **Creating a Microservice Engine**.
- For details about how to create an account, see **Adding an Account**.

Step 5 On the **Roles** tab page, click ⌄ next to the role to be viewed to expand the role details.

Service Group, **Configuration Group**, and **Action** of the role are displayed.

----End

9 Key Operations Recorded by CTS

9.1 ServiceStage Operations That Can Be Recorded by CTS

Cloud Trace Service (CTS) records ServiceStage application management operations, enabling you to query, audit, and review operations.

After CTS is **enabled**, the system starts recording operations on ServiceStage resources. You can view the operation records of the last seven days on the CTS console.

Table 9-1 ServiceStage operations that can be recorded by CTS

Operation	Resource Type	Event Name
Creating a component	component	createComponent
Deleting a component	component	deleteComponent
Upgrading a component	component	updateComponent
Starting a component	component	startComponent
Stopping a component	component	stopComponent
Restarting a component	component	restartComponent
Scaling a component	component	scaleComponent

Operation	Resource Type	Event Name
Rolling back a component	component	rollbackComponent
Deploying a component	component	provisionComponent
Uninstalling a component	component	deprovisionComponent
Creating an application	application	createApplication
Deleting an application	application	deleteApplication
Updating an application	application	updateApplication
Registering a VM agent	vmagent	registerVmagent
Deregistering a VM agent	vmagent	unregisterVmagent
Creating an environment	environment	createEnvironment
Deleting an environment	environment	deleteEnvironment

9.2 Querying Real-Time Traces

Scenarios

After you enable CTS and the management tracker is created, CTS starts recording operations on cloud resources. After a data tracker is created, the system starts recording operations on data in OBS buckets. CTS stores operation records generated in the last seven days.

This section describes how to query and export operation records of the last seven days on the CTS console.

- **Viewing Real-Time Traces in the Trace List of the New Edition**
- **Viewing Real-Time Traces in the Trace List of the Old Edition**

Constraints

- Traces of a single account can be viewed on the CTS console. Multi-account traces can be viewed only on the **Trace List** page of each account, or in the

OBS bucket or the **CTS/system** log stream configured for the management tracker with the organization function enabled.

- You can only query operation records of the last seven days on the CTS console. To store operation records for more than seven days, you must configure an OBS bucket to transfer records to it. Otherwise, you cannot query the operation records generated seven days ago.
- After performing operations on the cloud, you can query management traces on the CTS console 1 minute later and query data traces on the CTS console 5 minutes later.

Viewing Real-Time Traces in the Trace List of the New Edition

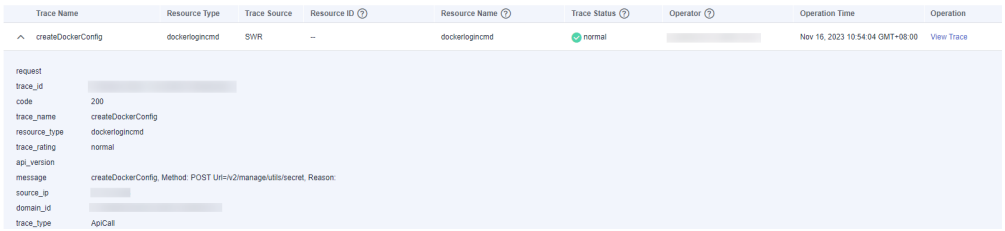
1. Log in to the management console.
2. Click  in the upper left corner and choose **Management & GovernanceManagement & Deployment** > **Cloud Trace Service**. The CTS console is displayed.
3. Choose **Trace List** in the navigation pane on the left.
4. On the **Trace List** page, use advanced search to query traces. You can combine one or more filters.
   - **Trace Name**: Enter a trace name.
   - **Trace ID**: Enter a trace ID.
   - **Resource Name**: Enter a resource name. If the cloud resource involved in the trace does not have a resource name or the corresponding API operation does not involve the resource name parameter, leave this field empty.
   - **Resource ID**: Enter a resource ID. Leave this field empty if the resource has no resource ID or if resource creation failed.
   - **Trace Source**: Select a cloud service name from the drop-down list.
   - **Resource Type**: Select a resource type from the drop-down list.
   - **Operator**: Select one or more operators from the drop-down list.
   - **Trace Status**: Select **normal**, **warning**, or **incident**.
     - **normal**: The operation succeeded.
     - **warning**: The operation failed.
     - **incident**: The operation caused a fault that is more serious than the operation failure, for example, causing other faults.
   - Time range: Select **Last 1 hour**, **Last 1 day**, or **Last 1 week**, or specify a custom time range.
5. On the **Trace List** page, you can also export and refresh the trace list, and customize the list display settings.
   - Enter any keyword in the search box and click  to filter desired traces.
   - Click **Export** to export all traces in the query result as an .xlsx file. The file can contain up to 5000 records.
   - Click  to view the latest information about traces.

- Click  to customize the information to be displayed in the trace list. If **Auto wrapping** is enabled (), excess text will move down to the next line; otherwise, the text will be truncated. By default, this function is disabled.

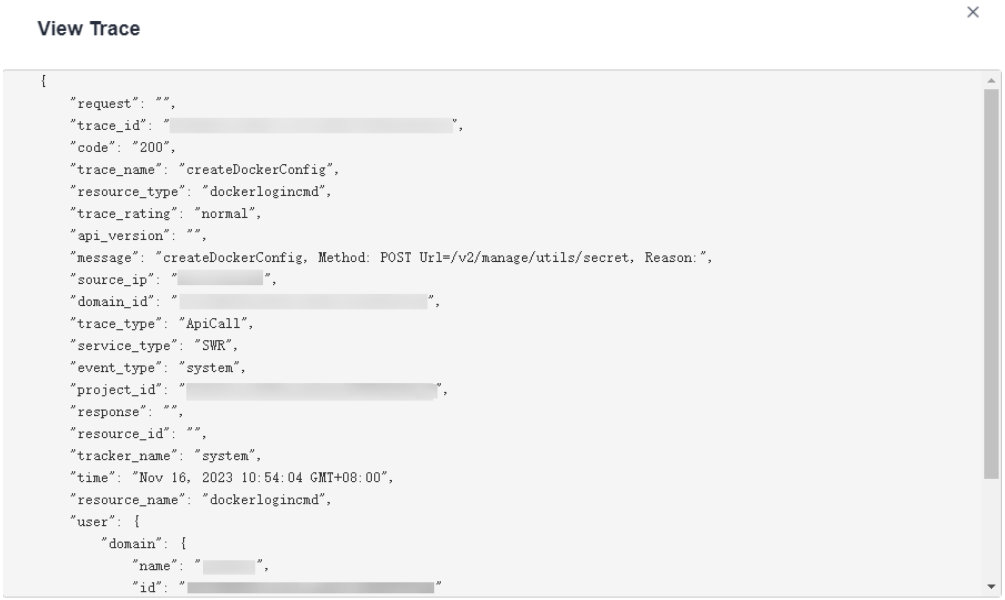
6. For details about key fields in the trace structure, see **Trace Structure**section "Trace References" > "Trace Structure" and **Example Traces**section "Trace References" > "Example Traces".
7. (Optional) On the **Trace List** page of the new edition, click **Go to Old Edition** in the upper right corner to switch to the **Trace List** page of the old edition.

Viewing Real-Time Traces in the Trace List of the Old Edition

1. Log in to the management console.
2. Click  in the upper left corner and choose **Management & GovernanceManagement & Deployment** > **Cloud Trace Service**. The CTS console is displayed.
3. Choose **Trace List** in the navigation pane on the left.
4. Each time you log in to the CTS console, the new edition is displayed by default. Click **Go to Old Edition** in the upper right corner to switch to the trace list of the old edition.
5. Set filters to search for your desired traces. The following filters are available:
   - **Trace Type**, **Trace Source**, **Resource Type**, and **Search By**: Select a filter from the drop-down list.
     - If you select **Resource ID** for **Search By**, specify a resource ID.
     - If you select **Trace name** for **Search By**, specify a trace name.
     - If you select **Resource name** for **Search By**, specify a resource name.
   - **Operator**: Select a user.
   - **Trace Status**: Select **All trace statuses**, **Normal**, **Warning**, or **Incident**.
   - Time range: You can query traces generated during any time range in the last seven days.
   - Click **Export** to export all traces in the query result as a CSV file. The file can contain up to 5000 records.
6. Click **Query**.
7. On the **Trace List** page, you can also export and refresh the trace list.
   - Click **Export** to export all traces in the query result as a CSV file. The file can contain up to 5000 records.
   - Click  to view the latest information about traces.
8. Click on the left of a trace to expand its details.



9. Click **View Trace** in the **Operation** column. The trace details are displayed.



10. For details about key fields in the trace structure, see **Trace Structure**section "Trace References" > "Trace Structure" and **Example Traces**section "Trace References" > "Example Traces" in the *CTS User Guide*.
11. (Optional) On the **Trace List** page of the old edition, click **New Edition** in the upper right corner to switch to the **Trace List** page of the new edition.

10 Viewing Monitoring Metrics and Alarms

Introduction

Application Operations Management (AOM) monitors and displays the running status of ServiceStage and the usage of each metric, and creates alarm rules for monitoring items.

After you use ServiceStage to deploy components, AOM can associate monitoring metrics of the components to help you master the performance metrics of the components in real time and accurately master the running status of the components.

Setting Monitoring and Alarms

ServiceStage supports container-based and VM-based deployment modes.

- Container-based deployment

  CCE works with AOM to comprehensively monitor clusters. When a node is created, the ICAgent (the DaemonSet named **icagent** in the kube-system namespace of the cluster) of AOM is installed by default. The ICAgent collects monitoring data of underlying resources and workloads running on the cluster, and uploads the data to AOM. In addition, after **Customizing Component Running Metrics**, the ICAgent can collect monitoring data of user-defined load metrics and upload the data to AOM.

  After **Configuring Alarm Thresholds for Resource Monitoring**, alarms generated during component running are reported to AOM.

- VM-based deployment

  Before deploying components on a VM, install the VM agent on the VM. During the installation, the AOM ICAgent is installed by default. The monitoring metrics of the VM-deployed components are uploaded to AOM.

Supported Metrics

Metrics reflect the resource performance or status.

- Metrics supported by the container-deployed components

Basic resource monitoring includes CPU, memory, and disk monitoring. For details, see **Table 10-1**.

Table 10-1 Resource metrics

Metric	Description	Value Range	Unit
Total CPU cores (cpuCoreLimit)	Total number of CPU cores that have been applied for a measured object	≥1	Cores
Used CPU cores (cpuCoreUsed)	Number of CPU cores used by a measured object	≥0	Cores
CPU usage (cpuUsage)	CPU usage of a measured object, that is, the ratio of the used CPU cores to the total CPU cores.	0%–100%	%
Total physical memory (memCapacity)	Total physical memory that has been applied for a measured object	≥0	MB
Physical memory usage (memUsage)	Percentage of the used physical memory to the total physical memory	0%–100%	%
Used physical memory (memUsed)	Used physical memory of a measured object	≥0	MB
Disk read rate (diskReadRate)	Volume of data read from a disk per second	≥0	KB/s
Disk write rate (diskWriteRate)	Volume of data written into a disk per second	≥0	KB/s
Downlink rate (recvPackRate)	Number of data packets received by the NIC per second	≥0	Packets per second (PPS)

Metric	Description	Value Range	Unit
Total file system (filesystemCapacity)	Total file system capacity of a measured object. This metric is available only for containers using the Device Mapper storage drive in the Kubernetes cluster of version 1.11 or later.	≥0	MB
Downlink rate (recvBytesRate)	Inbound traffic rate of a measured object	≥0	Byte per second (BPS)
Downlink error rate (recvErrPackRate)	Number of error packets received by an NIC per second	≥0	PPS
Uplink rate (sendPackRate)	Outbound traffic rate of a measured object	≥0	BPS
Uplink error rate (sendErrPackRate)	Number of error packets sent by the NIC per second	≥0	PPS
Uplink rate (sendBytesRate)	Outbound traffic rate of a measured object	≥0	BPS
Error packets (rxPackErrors)	Number of error packets received by a measured object	≥0	Packets
Threads (threadsCount)	Number of threads created on a host	≥0	N/A
Available file system (filesystemAvailable)	Available file system capacity of a measured object. This metric is available only for containers using the Device Mapper storage drive in the Kubernetes cluster of version 1.11 or later.	≥0	MB

Metric	Description	Value Range	Unit
File system usage (filesystemUsage)	File system usage of a measured object, that is, the ratio of the used file system to the total file system. This metric is available only for containers using the Device Mapper storage drive in the Kubernetes cluster of version 1.11 or later.	≥0	%
Handles (handleCount)	Number of handles used by a measured object	≥0	N/A
Component status (status)	Status of an application group	• 0: normal • 1: abnormal	N/A
Total virtual memory (virMemCapacity)	Total virtual memory that has been applied for a measured object	≥0	MB

- Metrics supported by the VM-deployed components

  In AOM, VM-deployed components refer to processes, and VM-deployed component metrics refer to process metrics. For details, see **Table 10-2**.

  Table 10-2 Process metrics

Metric	Description	Value Range	Unit
Total CPU cores (cpuCoreLimit)	Total number of CPU cores that have been applied for a measured object	≥1	Cores
Used CPU cores (cpuCoreUsed)	Number of CPU cores used by a measured object	≥0	Cores
CPU usage (cpuUsage)	CPU usage of a measured object, that is, the ratio of the used CPU cores to the total CPU cores.	0%–100%	%
Handles (handleCount)	Number of handles used by a measured object	≥0	N/A

Metric	Description	Value Range	Unit
Total physical memory (memCapacity)	Total physical memory that has been applied for a measured object	≥0	MB
Physical memory usage (memUsage)	Percentage of the used physical memory to the total physical memory	0%–100%	%
Used physical memory (memUsed)	Used physical memory of a measured object	≥0	MB
Status (status)	Process status	• 0: normal • 1: abnormal	N/A
Threads (threadsCount)	Number of threads used by a measured object	≥0	N/A
Total virtual memory (virMemCapacity)	Total virtual memory that has been applied for a measured object	≥0	MB

