ALIBABA CLOUD

Alibaba Cloud

Container Service
User Guide

Document Version: 20220630

Alibaba Cloud

Legal disclaimer

Alibaba Cloud reminds you to carefully read and fully understand the terms and conditions of this legal disclaimer before you read or use this document. If you have read or used this document, it shall be deemed as your total acceptance of this legal disclaimer.

1. You shall download and obtain this document from the Alibaba Cloud website or other Alibaba Cloud-authorized channels, and use this document for your own legal business activities only. The content of this document is considered confidential information of Alibaba Cloud. You shall strictly abide by the confidentiality obligations. No part of this document shall be disclosed or provided to any third party for use without the prior written consent of Alibaba Cloud.

2. No part of this document shall be excerpted, translated, reproduced, transmitted, or disseminated by any organization, company or individual in any form or by any means without the prior written consent of Alibaba Cloud.

3. The content of this document may be changed because of product version upgrade, adjustment, or other reasons. Alibaba Cloud reserves the right to modify the content of this document without notice and an updated version of this document will be released through Alibaba Cloud-authorized channels from time to time. You should pay attention to the version changes of this document as they occur and download and obtain the most up-to-date version of this document from Alibaba Cloud-authorized channels.

4. This document serves only as a reference guide for your use of Alibaba Cloud products and services. Alibaba Cloud provides this document based on the "status quo", "being defective", and "existing functions" of its products and services. Alibaba Cloud makes every effort to provide relevant operational guidance based on existing technologies. However, Alibaba Cloud hereby makes a clear statement that it in no way guarantees the accuracy, integrity, applicability, and reliability of the content of this document, either explicitly or implicitly. Alibaba Cloud shall not take legal responsibility for any errors or lost profits incurred by any organization, company, or individual arising from download, use, or trust in this document. Alibaba Cloud shall not, under any circumstances, take responsibility for any indirect, consequential, punitive, contingent, special, or punitive damages, including lost profits arising from the use or trust in this document (even if Alibaba Cloud has been notified of the possibility of such a loss).

5. By law, all the contents in Alibaba Cloud documents, including but not limited to pictures, architecture design, page layout, and text description, are intellectual property of Alibaba Cloud and/or its affiliates. This intellectual property includes, but is not limited to, trademark rights, patent rights, copyrights, and trade secrets. No part of this document shall be used, modified, reproduced, publicly transmitted, changed, disseminated, distributed, or published without the prior written consent of Alibaba Cloud and/or its affiliates. The names owned by Alibaba Cloud shall not be used, published, or reproduced for marketing, advertising, promotion, or other purposes without the prior written consent of Alibaba Cloud. The names owned by Alibaba Cloud include, but are not limited to, "Alibaba Cloud", "Aliyun", "HiChina", and other brands of Alibaba Cloud and/or its affiliates, which appear separately or in combination, as well as the auxiliary signs and patterns of the preceding brands, or anything similar to the company names, trade names, trademarks, product or service names, domain names, patterns, logos, marks, signs, or special descriptions that third parties identify as Alibaba Cloud and/or its affiliates.

6. Please directly contact Alibaba Cloud for any errors of this document.

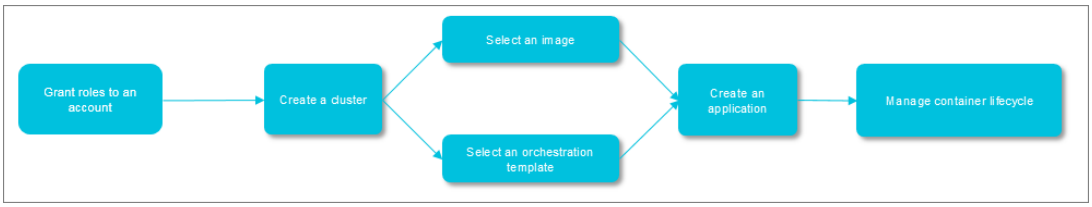
Document conventions

Style	Description	Example
Danger	A danger notice indicates a situation that will cause major system changes, faults, physical injuries, and other adverse results.	**Danger:** Resetting will result in the loss of user configuration data.
Warning	A warning notice indicates a situation that may cause major system changes, faults, physical injuries, and other adverse results.	**Warning:** Restarting will cause business interruption. About 10 minutes are required to restart an instance.
Notice	A caution notice indicates warning information, supplementary instructions, and other content that the user must understand.	**Notice:** If the weight is set to 0, the server no longer receives new requests.
Note	A note indicates supplemental instructions, best practices, tips, and other content.	**Note:** You can use Ctrl + A to select all files.
>	Closing angle brackets are used to indicate a multi-level menu cascade.	Click **Settings**> **Network**> **Set network type**.
Bold	Bold formatting is used for buttons , menus, page names, and other UI elements.	Click **OK**.
`Courier font`	Courier font is used for commands	Run the `cd /d C:/window` command to enter the Windows system folder.
Italic	Italic formatting is used for parameters and variables.	`bae log list --instanceid` *Instance_ID*
[] or [a|b]	This format is used for an optional value, where only one item can be selected.	`ipconfig [-all|-t]`
{} or {a|b}	This format is used for a required value, where only one item can be selected.	`switch {active|stand}`

Table of Contents

1.Workflow

The complete workflow for Container Service is as follows.



Step 1: Grant roles to an account.

For more information, see Role authorization.

Step 2: Create a cluster.

You can select the network environment of the cluster, and set the number of nodes and configurations for the cluster.

Step 3: Create an application by using an image or orchestration template.

Select an existing image or orchestration template, or create a new image or orchestration template.

If your application is composed of services supported by multiple images, create the application by using an orchestration template.

Step 4: Check the application status and the information of relevant services and containers after the deployment.

2.Authorizations

2.1. Role authorization

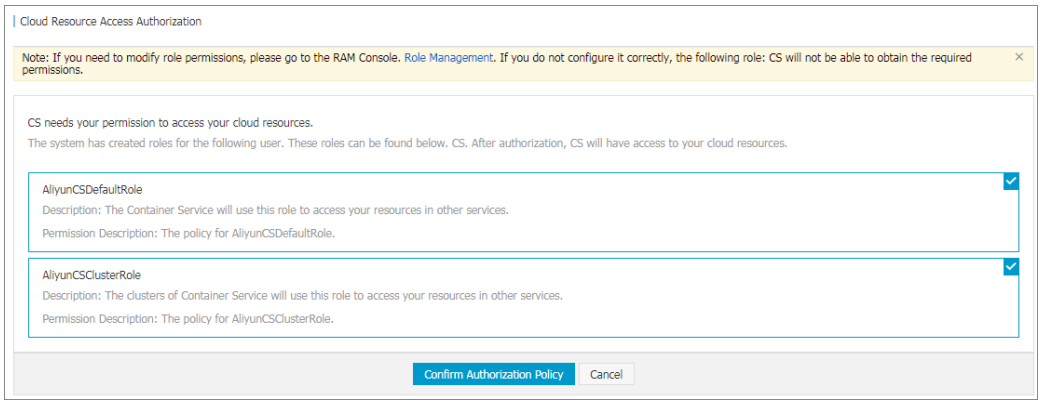
Grant the system default roles AliyunCSDefaultRole and AliyunCSClusterRole to the service account when you activate Container Service. Only after the roles are correctly granted, Container Service can normally call the services such as Elastic Compute Service (ECS), Object Storage Service (OSS), NAS, and Server Load Balancer (SLB), create clusters, and store logs.

Instructions

- If you have used Container Service before 15 January 2018, the system completes the role authorization by default. For more information about the permissions associated with each role, see the API documents of each product. If you used Container Service with a Resource Access Management (RAM) user before, upgrade the authorization policy for the RAM user. For more information, see 子账号策略升级.
- On 15 January 2018, Container Service is fully accessed to the cross-service authorization. New users who use the primary account can use Container Service only after having the cross-service authorization completed. If new users need to authorize RAM users to use Container Service, go to the RAM console to authorize the RAM users. For more information, see Use sub-accounts.

Procedure

1. If you have not granted the default roles to the service account correctly, the Cloud Resource Access Authorization page appears after you log on to the Container Service console. Click **Confirm Authorization Policy**.



> **Note** Container Service has configured the default role permissions. To modify the role permissions, go to the User Management page of the RAM console. Note that incorrect configurations might cause Container Service cannot obtain the required permissions.

2. After completing the authorization, refresh the Container Service console and then perform the operations.

   To view the policy details of the roles AliyunCSDefaultRole and AliyunCSClusterRole, log on to the RAM console.

Default role permissions

For more information about permissions of each role, see the API documents of each product.

AliyunCSDefaultRole permissions

The default role AliyunCSDefaultRole contains the following main permissions:

- ECS-related permissions

Action	Description
ecs:RunInstances	Query ECS instance information.
ecs:RenewInstance	Renew ECS instances.
ecs:Create*	Create ECS-related resources, such as instances and disks.
ecs:AllocatePublicIpAddress	Allocate public IP addresses.
ecs:AllocateEipAddress	Allocate Elastic IP (EIP) addresses.
ecs:Delete*	Delete ECS instances.
ecs:StartInstance	Start ECS-related resources.
ecs:StopInstance	Stop ECS instances.
ecs:RebootInstance	Restart ECS instances.
ecs:Describe*	Query ECS-related resources.
ecs:AuthorizeSecurityGroup	Configure inbound security group rules.
ecs:RevokeSecurityGroup	Revoke security group rules.
ecs:AuthorizeSecurityGroupEgress	Configure outbound security group rules.
ecs:AttachDisk	Add disks.
ecs:DetachDisk	Clean up disks.
ecs:AddTags	Add tags.
ecs:ReplaceSystemDisk	Change system disks of ECS instances.
ecs:ModifyInstanceAttribute	Modify ECS instance attributes.
ecs:JoinSecurityGroup	Add ECS instances to specified security groups.
ecs:LeaveSecurityGroup	Remove ECS instances from specified security groups.
ecs:UnassociateEipAddress	Unbind EIP addresses.
ecs:ReleaseEipAddress	Release EIP addresses.

- Virtual Private Cloud (VPC)-related permissions

Action	Description
vpc:Describe*	Query information of VPC-related resources.
vpc:DescribeVpcs	Query VPC information.
vpc:AllocateEipAddress	Allocate EIP addresses.
vpc:AssociateEipAddress	Associate with EIP addresses.
vpc:UnassociateEipAddress	Do not associate with EIP addresses.
vpc:ReleaseEipAddress	Release EIP addresses.
vpc:CreateRouteEntry	Create router interfaces.
vpc:DeleteRouteEntry	Delete router interfaces.

- SLB-related permissions

Action	Description
slb:Describe*	Query information related to Server Load Balancer.
slb:CreateLoadBalancer	Create Server Load Balancer instances.
slb:DeleteLoadBalancer	Delete Server Load Balancer instances.
slb:RemoveBackendServers	Unbind Server Load Balancer instances.
slb:StartLoadBalancerListener	Start specified listeners.
slb:StopLoadBalancerListener	Stop specified listeners.
slb:CreateLoadBalancerTCPListener	Create TCP-based listening rules for Server Load Balancer instances.
slb:AddBackendServers	Add backend servers.

AliyunCSClusterRole permissions

The default role AliyunCSClusterRole contains the following main permissions:

- OSS-related permissions

Action	Description
oss:PutObject	Upload files or folders.
oss:GetObject	Retrieve files or folders.
oss:ListObjects	Query file list information.

- NAS-related permissions

Action	Description
nas:Describe*	Return NAS-related information.
nas:CreateAccessRule	Create permission rules.

- SLB-related permissions

Action	Description
slb:Describe*	Query information related to Server Load Balancer.
slb:CreateLoadBalancer	Create Server Load Balancer instances.
slb:DeleteLoadBalancer	Delete Server Load Balancer instances.
slb:RemoveBackendServers	Unbind Server Load Balancer instances.
slb:StartLoadBalancerListener	Start specified listeners.
slb:StopLoadBalancerListener	Stop specified listeners.
slb:CreateLoadBalancerTCPListener	Create TCP-based listening rules for Server Load Balancer instances.
slb:AddBackendServers	Add backend servers.
slb:DeleteLoadBalancerListener	Delete listening rules of Server Load Balancer instances.
slb:CreateVServerGroup	Create VServer groups and add backend servers.
slb:ModifyVServerGroupBackendServers	Change backend servers in VServer groups.
slb:CreateLoadBalancerHTTPListener	Create HTTP-based listeners for Server Load Balancer instances.
slb:SetBackendServers	Configure backend servers and set the weight for a group of ECS instances at the Server Load Balancer instance backend.
slb:AddTags	Add tags for Server Load Balancer instances.

2.2. Use sub-accounts

Grant the sub-accounts the corresponding permissions before using the sub-accounts to log on to the Container Service console and perform the operations.

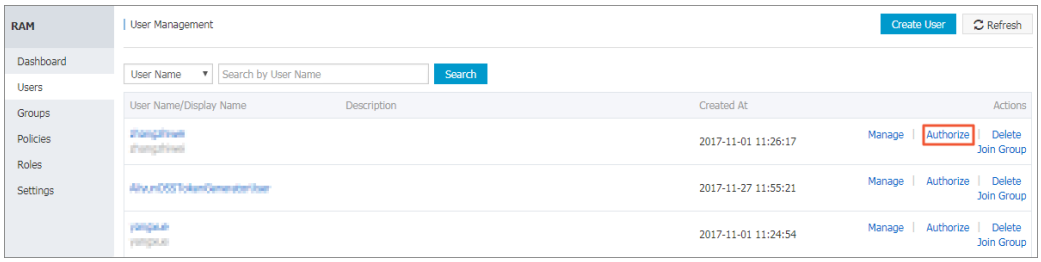
Step 1 Create sub-accounts and enable console logon

1. Log on to the RAM console.

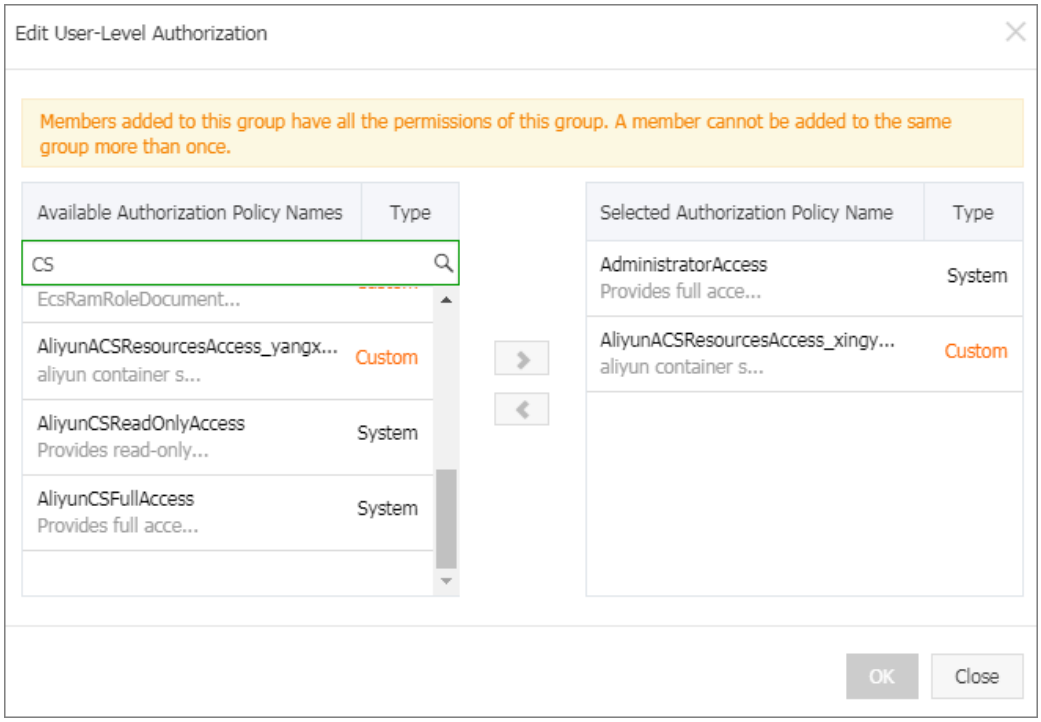
2. Click **Users** in the left-side navigation pane. Click **Create User** in the upper-right corner.
3. Enter the username of the sub-account and then click **OK**.
4. On the User Management page, click **Manage** at the right of the created sub-account.
5. Click **Enable Console Logon** in the **Web Console Logon Management** section.
6. Enter the logon password in the appeared dialog box and click **OK**.

Step 2 Grant sub-accounts permissions to access Container Service

1. On the User Management page, click **Authorize** at the right of the created sub-account.



2. Select the authorization policy and click 1 to add the policy to the Selected Authorization Policy Name.



You can use the following system default authorization policies:

- AliyunCSFullAccess: Provides full access to Container Service.
- AliyunCSReadOnlyAccess: Provides read-only access to Container Service.

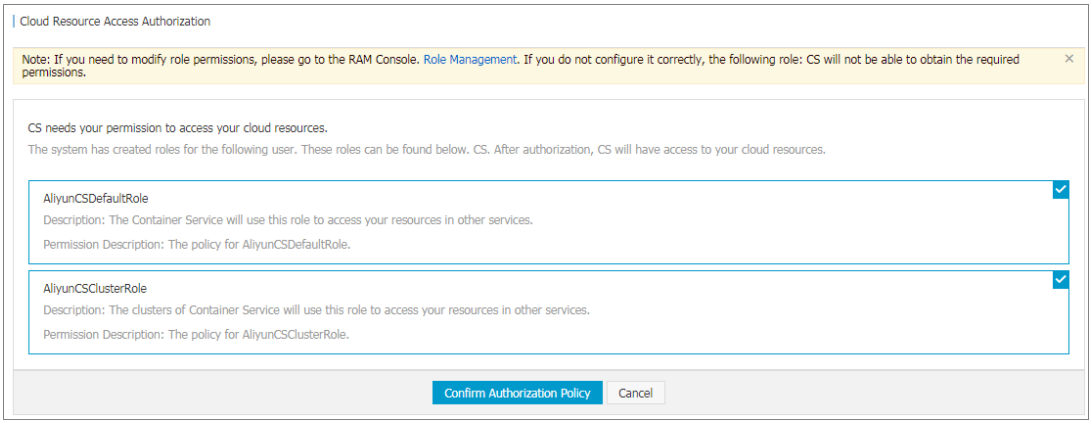
You can also create custom authorization policies as per your needs and grant the policies to the sub-accounts. For more information, see Create custom authorization policies.

Step 3 Log on to Container Service console with sub-accounts

Log on to the Container Service console with a sub-account.

If you have granted the AliyunCSDefaultRole and AliyunCSClusterRole roles to the main account, you can use the sub-account directly to log on to the Container Service console and perform the operations.

If you have not granted the AliyunCSDefaultRole or AliyunCSClusterRole roles to the main account before, click **Confirm Authorization Policy** in the appeared Cloud Resource Access Authorization page.



Then, refresh the Container Service console to perform the operations.

2.3. Create custom authorization policies

The authorization granularity of the system authorization policies provided by Container Service is coarse. If these authorization policies with coarse granularity cannot satisfy your requirements, create the custom authorization policies. For example, to control the permissions to a specific cluster, you must use the custom authorization policy to meet the requirements with fine granularity.

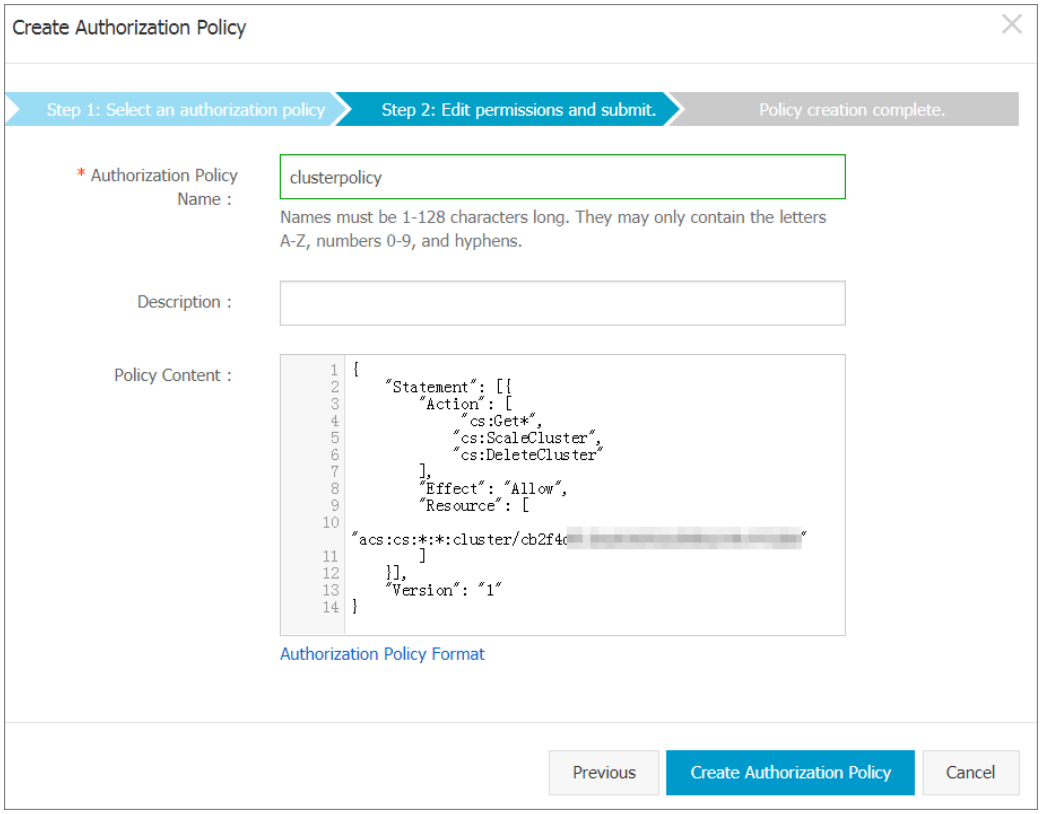
Create custom authorization policies

Get to know the basic structure and syntax of the authorization policy language before creating custom authorization policies. For more information, see Authorization policy language descriptions.

This document introduces how to grant Resource Access Management (RAM) users permissions to query, expand, and delete clusters.

Procedure

1. Log on to the RAM console with the primary account.
2. Click **Policies** in the left-side navigation pane. Click **Create Authorization Policy** in the upper-right corner.
3. Select a template. Enter the authorization policy name and the policy content.




```
"Statement": [{
    "Action": [
        "cs:Get*",
        "cs:ScaleCluster",
        "cs:DeleteCluster"
    "Effect": "Allow",
    "Resource": [
        "acs:cs:*:*:cluster/cluster ID"
"Version": "1"
```

Wherein:

- `Action:` Enter the permission that you want to grant.

  > **Note** All the Actions support wildcards.

- `Resource` supports the following configuration methods.
  - Grant permissions of a single cluster

    ```
    "Resource": [
         "acs:cs:*:*:cluster/cluster ID"
    ```

- Grant permissions of multiple clusters

```
"Resource": [
     "acs:cs:*:*:cluster/cluster ID",
     "acs:cs:*:*:cluster/cluster ID"
```

- Grant permissions of all your clusters

```
"Resource": [
```

You must replace `cluster ID` with your actual cluster ID.

4. Click **Create Authorization Policy** after completing the configurations.

Container Service RAM action

Action	Description
CreateCluster	Create clusters.
AttachInstances	Add existing Elastic Compute Service (ECS) instances to clusters.
ScaleCluster	Expand clusters.
GetClusters	View cluster list.
GetClusterById	View cluster details.
ModifyClusterName	Modify cluster names.
DeleteCluster	Delete clusters.
UpgradeClusterAgent	Upgrade cluster Agent.
GetClusterLogs	View cluster operation logs.
GetClusterEndpoint	View cluster access point.
GetClusterCerts	Download cluster certificate.
RevokeClusterCerts	Revoke cluster certificate.
BindSLB	Bind Server Load Balancer instances to clusters.
UnBindSLB	Unbind Server Load Balancer instances from clusters.
ReBindSecurityGroup	Rebind security groups to clusters.
CheckSecurityGroup	Check existing security group rules of clusters.
FixSecurityGroup	Fix cluster security group rules.
ResetClusterNode	Reset cluster nodes.

Action	Description
DeleteClusterNode	Delete cluster nodes.
CreateAutoScale	Create node auto scaling rules.
UpdateAutoScale	Update node auto scaling rules.
DeleteAutoScale	Delete node auto scaling rules.
GetClusterProjects	View applications in clusters.
CreateTriggerHook	Create triggers for applications.
GetTriggerHook	View application trigger list.
RevokeTriggerHook	Delete application triggers.
CreateClusterToken	Create tokens.

3.Clusters

3.1. Cluster introduction

A cluster is a collection of cloud resources that are required to run containers. It is associated with several Elastic Compute Service (ECS) nodes, Server Load Balancer, and other cloud resources.

Create a cluster

You can create a cluster by using the following methods:

Method 1: Create a cluster and several ECS instances.

You can directly create a cluster with several new ECS instances by using Container Service.

The ECS instances created using this method are all Pay-As-You-Go instances. If you want to use monthly or yearly subscription ECS instances, buy them separately and then follow **Method 2**.

Method 2: Create a zero-node cluster and add existing ECS instances to the cluster.

1. Create a zero-node cluster.

   If you have purchased several ECS instances from the ECS service, create a zero-node cluster in Container Service. Method 1 except that you need to select Do not Add when creating the cluster to add existing ECS instances instead of creating some new ones.

   The operations are the same as **Method 1** except that you need to select **Do not Add** when creating the cluster to add existing ECS instances instead of creating some new ones.

2. Add existing ECS instances.

   You can add an existing ECS instance to Container Service in the following ways:

   - Reset the image of the ECS instance and add the ECS instance to the cluster automatically.

     As this method will reset the image and system disk of the ECS instance, proceed with caution. However, ECS instances added by using this method are cleaner.

   - Run scripts on the ECS instance and manually add the ECS instance to the cluster.

     This method is applicable to images that do not require a reset of the ECS instance.

   Add an existing ECS instance.

Manage a cluster

You can search for, expand, connect to, clean up, or delete a cluster. For more information, see the following documents:

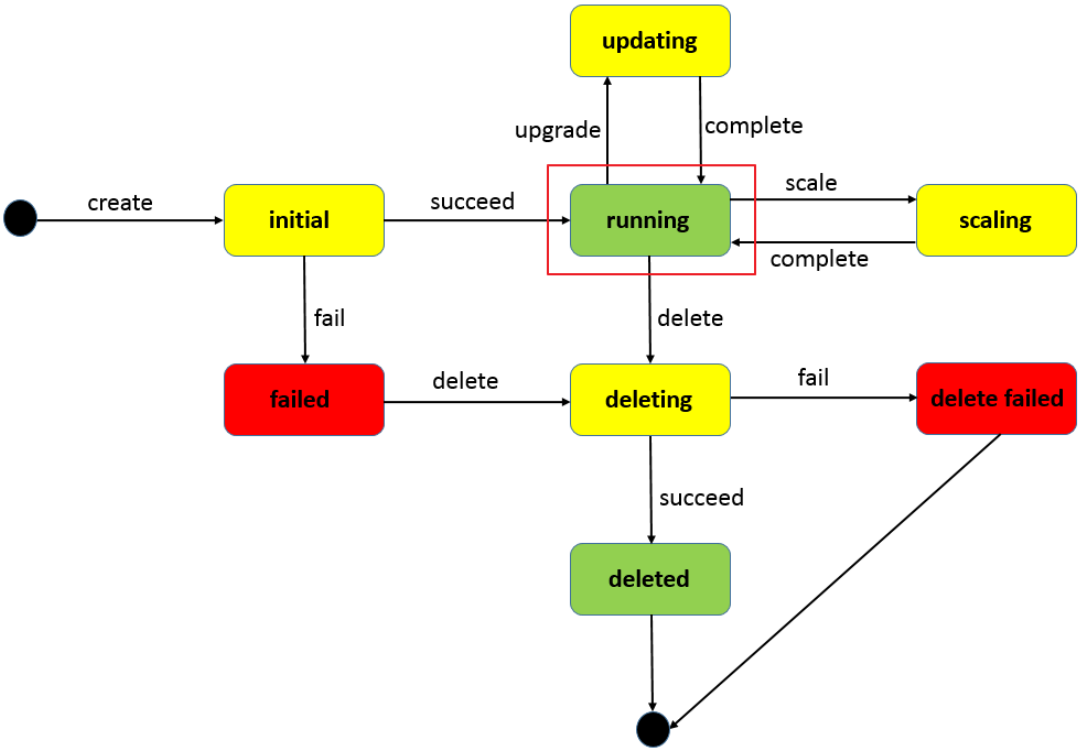
- Search for a cluster
- Expand a cluster
- Download cluster certificate
- Clean up a cluster disk
- Delete a cluster

3.2. Cluster lifecycle

A complete cluster lifecycle includes the following statuses.

Status	Description
inactive	The successfully created cluster does not contain any node.
initial	The cluster is applying for corresponding cloud resources.
running	The cluster successfully applied for the cloud resources.
updating	The cluster is upgrading the Agent.
scaling	Change the number of cluster nodes.
failed	The cluster application for cloud resources failed.
deleting	The cluster is being deleted.
delete_failed	The cluster failed to be deleted.
deleted (invisible to users)	The cluster is successfully deleted.

Cluster status flow



3.3. Create a cluster

You can specify the configurations and the number of Elastic Compute Service (ECS) instances when creating clusters. You can also create a zero-node cluster, and then bind it with other ECS instances.

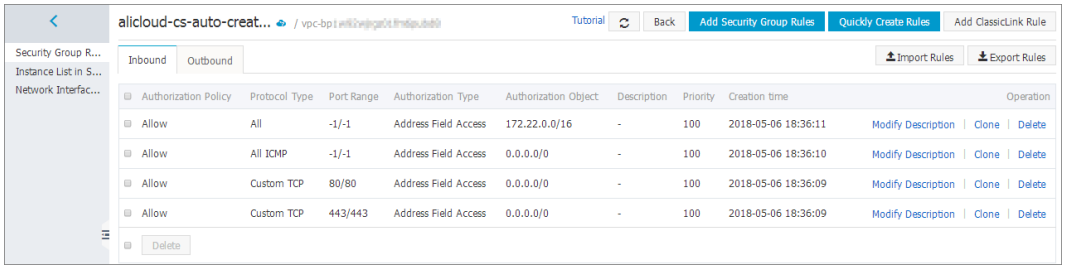
> **Note** The zero-node cluster is in the Inactive status after the creation and is activated with the Running status after you add ECS instances to it. For how to add existing ECS instances to the cluster, see Add an existing ECS instance.

Instructions

Container Service performs the following operations when creating a cluster:

- Create a Server Load Balancer instance with 80:9080 configured as the listener if the **Automatically Create Server Load Balancer** check box is selected.
- Create a security group. The security group rules are as follows.

  Virtual Private Cloud (VPC) inbound



- Create a Resource Access Management (RAM) user if you have activated the RAM service.
- Create the ECS instances and distribute the Internet IP address to the ECS instances if you select **Add** in the Add Node field. (If the Network Type is VPC, distribute the Elastic IP (EIP) to the ECS instances and create the corresponding routing rules.)
- Use the configured **Logon Password** to configure the ECS instances.

  > **Note** Container Service does not save this password.

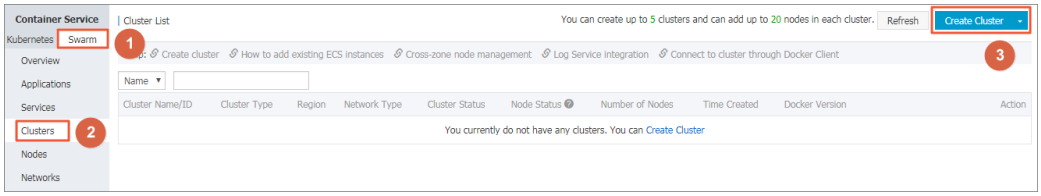
- If the VPC node configuration fails, Container Service collects the standard output of the node creation and initialization. You can view the information in the cluster logs.

Limits

- Server Load Balancer instances created with clusters are only available in Pay-As-You-Go mode.
- By default, each account has a certain quota for the cloud resources they can create. The cluster fails to be created if the quota is exceeded. Make sure you have enough quota before creating the cluster. To increase your quota, open a ticket.
  - By default, each account can create at most five clusters in all regions and add up to 20 nodes to each cluster.
  - By default, each account can create at most 100 security groups.
  - By default, each account can create at most 60 Pay-As-You-Go Server Load Balancer instances.
  - By default, each account can create at most 20 EIPs.

Procedure

1. Log on to the Container Service console.
2. Click Swarm > **Clusters** in the left-side navigation pane. Click **Create Cluster** in the upper-right corner.



3. Complete the following configurations.
   - **Cluster Name**: Enter the name of the cluster. It can be 1-63 characters long and contain numbers, Chinese characters, English letters, and hyphens (-).

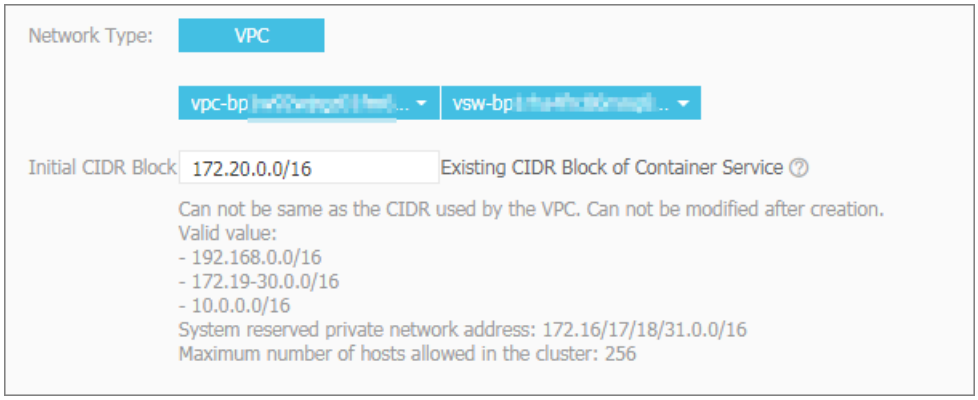
     > **Note** The cluster name must be unique under the same account and the same region.

   - **Region**: Select the region in which the cluster is to be deployed.
   - **Zone**: Select the zone for the cluster.

     > **Note** You can select the region and zone according to the distribution of your servers.

4. Select the network type of the cluster. Currently, Container Service only supports VPC.

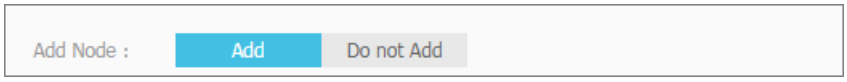
   Complete the corresponding configurations.



   VPC enables you to build an isolated network environment based on Alibaba Cloud. You can have full control over your own virtual network, including a free IP address range, Classless Inter-Domain Routing (CIDR) block division, and the configurations of route table and gateway.

   Specify a VPC, a VSwitchId, and the initial CIDR block of containers (the subnet CIDR block to which the Docker containers belong. For ease of IP management, containers of different virtual machines belong to different CIDR blocks, and container subnet CIDR block cannot conflict with virtual machine CIDR block). We recommend that you build your own VPC/VSwitchId for the cluster to prevent issues such as network conflicts.

5. Select whether or not to add nodes.



You can create a cluster with several new ECS instances, or create a zero-node cluster and then add existing ECS instances to the cluster. For how to add existing ECS instances to the cluster, see Add an existing ECS instance.

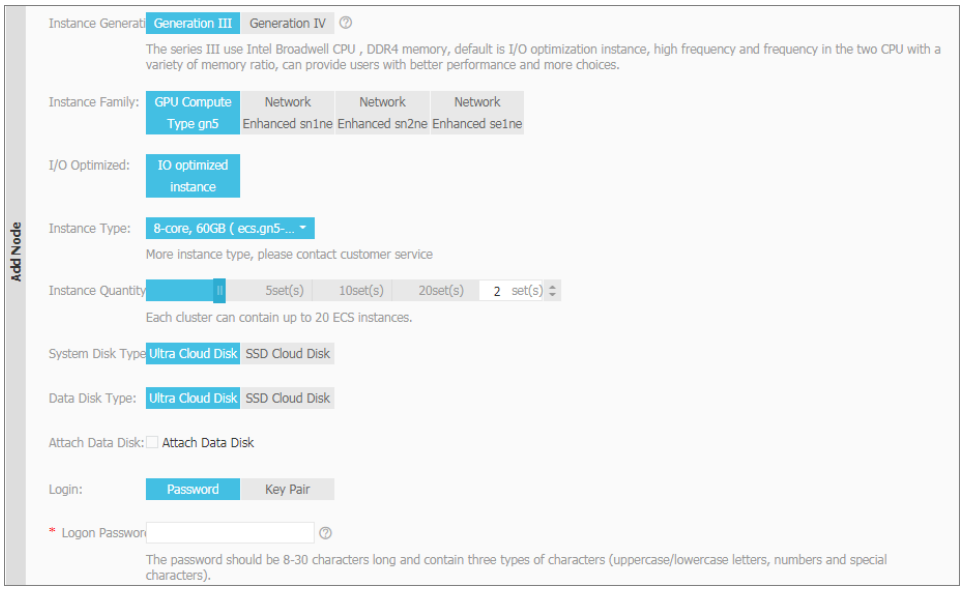
- **Add**

  a. Select the operating system for the node.

  Currently, Ubuntu 14.04/16.04 64bit and CentOS 7.4 64bit are supported.

  b. Configure the ECS instance specifications.

  You can select the generation, family, type, and quantity of the instance, disk type and capacity (the ECS instance has a 20 GB system disk by default), and logon password. Container Service uses the configured **Logon Password** to configure the ECS instances when creating the cluster, but does not save this password.

  > **Note**
  >
  > - The data disk is mounted to *the /var/lib/docker* directory and used for the storage of Docker images and containers if you select the Attach Data Disk check box.
  > - In terms of performance and management, we recommend that you mount an independent data disk to the host and manage the persistent data of containers by using Docker volumes.

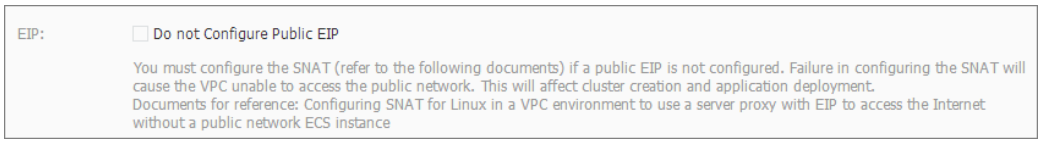
- **Do not Add**

You can click **Add Existing Instance** to add existing ECS instances to the cluster, or click Add Existing Instances on the Cluster List page to add existing ECS instances to the cluster after the cluster is created. For more information, see Add an existing ECS instance.

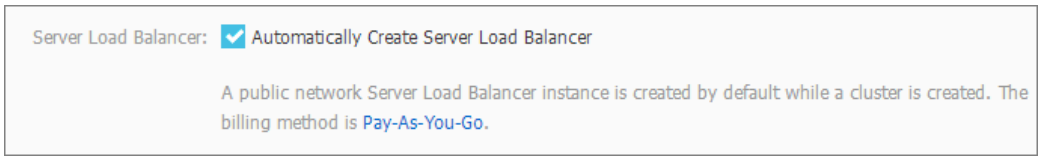
6. Select whether or not to configure public EIP.

   If you select VPC as the network type, Container Service configures an EIP for each ECS instance in the VPC environment by default. If this is not required, select the **Do not Configure Public EIP** check box and then configure the SNAT gateway.

   > **Note** You can apply for up to 20 EIPs per account. To use VPC and create EIP automatically when creating a cluster, the cluster fails to be created if the number of EIPs under your account reaches its quota.

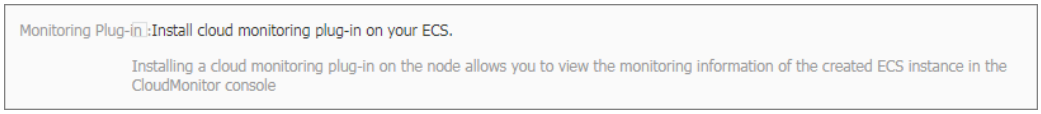
7. Select whether or not to create a Server Load Balancer instance.

   The Automatically Create Server Load Balancer check box is selected by default. With this check box selected, a Server Load Balancer instance is created after the cluster is created. You can access the container applications in the cluster by means of this Server Load Balancer instance. The created Server Load Balancer instance is in the Pay-As-You-Go mode.

8. Select whether or not to install cloud monitoring plug-in on your ECS instances.

   To view the monitoring information of the created ECS instances in the CloudMonitor console, select the Install cloud monitoring plug-in on your ECS check box.

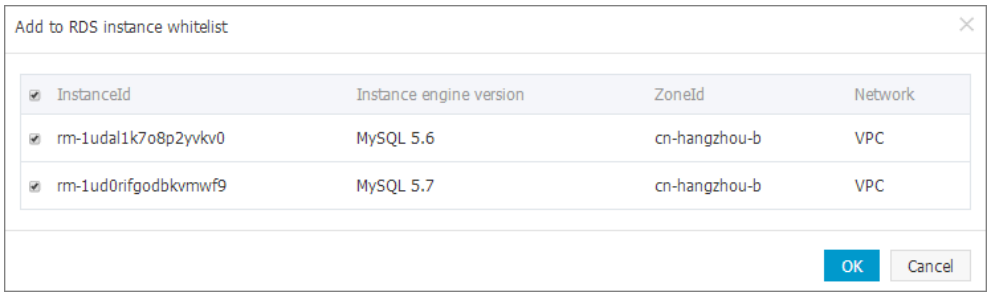
9. You can select to add the IP addresses of the ECS instances to the RDS instance whitelist.

   Adding the IP addresses of the ECS instances to the RDS instance whitelist facilitates the ECS instances to access the RDS instances.

> **Note**
>
> - We recommend that you configure the RDS Whitelist when **Add** is selected for Add Node.
> - If **Do not Add** is selected for Add Node and you want to configure the RDS Whitelist, add the existing ECS instances on the Create Cluster page. The RDS Whitelist cannot be configured if you create a zero-node cluster and add existing ECS instances after the cluster creation.
> - The ECS instance must be in the same region as the RDS instance so that the IP address of the ECS instance can be added to the RDS instance whitelist.

i. Click **Select RDS Instances**. The Add to RDS instance whitelist dialog box appears.

ii. Select the RDS instances and then click **OK**.

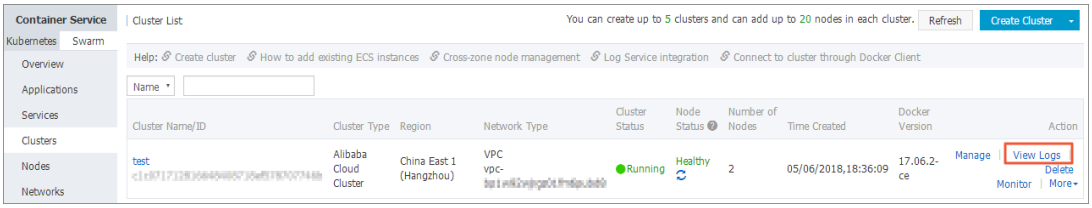


10. Click **Create Cluster**.

    After the cluster is successfully created, you can configure the ECS instance or Server Load Balancer instance in the corresponding console.

Subsequent operations

On the **Cluster List** page, you can click **View Logs** at the right of the cluster to view the creation process logs of the cluster.



You can create applications in the created cluster. For more information, see Create an application.

3.4. Cluster parameter configurations

This document aims to help you understand what the parameters on the page mean when you create a cluster. Then, you can configure the parameters smoothly. For some parameters, some documents are provided for your reference.

Cluster Name

Configure the cluster name.

- The name can be 1-63 characters long and contain numbers, Chinese characters, English letters, and hyphens (-), but cannot start with a hyphen (-).
- You can modify the cluster name on the Cluster List page after creating the cluster.

Region and Zone

Container Service authorizes to create the region and zone of the Elastic Compute Service (ECS) instances. Currently, the regions and zones supported by Container Service belong to the subset of ECS product. For more information, see Regions and zones.

Network Type

Select VPC as the network type of the ECS instances. Alibaba Cloud Virtual Private Cloud (VPC) allows you to create a custom VPC. Layer-2 logical isolation exists between different VPCs. You can plan the Classless Inter-Domain Routing (CIDR) block of each cluster flexibly. VPC is applicable to a scenario with large-scale container clusters and provides higher security and flexibility. To better guarantee the system security and the support of hybrid cloud business, Container Service does not support creating clusters whose network type is classic network or with non-I/O optimized instance since January 1, 2018.

Initial CIDR Block of Container Service

Configure this parameter only when you select VPC. When planning the CIDR block, make sure the container initial CIDR block does not overlap with the VPC CIDR block.

- You can only specify one CIDR block for each VPC. 172.16.0.0/12 is the default VPC CIDR block.
- Specify the corresponding container CIRD block when creating a Container Service cluster. Currently, Container Service supports the following container CIDR blocks: 192.168.1.0/24 and 172.[16-31].1.0/24

Add Node

Container Service has two ways to add nodes: create nodes and add existing nodes. If you select Add, Container Service is authorized to automatically create ECS instances when the cluster is created and automatically add the created ECS instances to the created cluster. If you select Do not Add, the existing ECS instances are added to the cluster. You can add the existing ECS instances on the Create Cluster page directly or create a zero-node cluster and then add the existing ECS instances on the Cluster List page. For more information, see Add an existing ECS instance.

Node Type

The node type is Pay-As-You-Go by default. After creating the ECS instances, you can go to the ECS console to change the Pay-As-You-Go ECS instances to monthly or yearly subscription ECS instances.

Operating System

Select the operating system installed in the ECS instances. We recommend that you use Ubuntu 14.04 64 bit and CentOS 7.4 64 bit.

Instance Generation and Instance Family

Different instance generations correspond to different instance families. ECS instances provide you with corresponding computing capabilities based on the instance specifications. ECS instances can be divided into many generations and families according to the business scenarios and usage scenarios. For the specific scenarios for each instance generation and family, see Instance family.

Instance Type

ECS instance type defines two basic attributes: the CPU configuration and memory configuration of the instance. However, ECS instances can determine the specific service pattern of an instance only by working together with the disk, image, and network type.

Instance Quantity

The number of the ECS instances to be created. The number of ECS instances in one cluster cannot exceed 20. To enhance the cluster availability, we do not recommend that you create a cluster with one node. 2 sets is the default value in the console.

System Disk Type

Select the cloud disk type of the installation system. Select Ultra Cloud Disk or SSD Cloud Disk according to your requirements on the system performance of the ECS instances. For the performance indicator comparison between these two types of cloud disks, see EBS performance.

Data disk configurations

Select the type of the data disk that is to be mounted to the container. Select the **Attach Data Disk** check box and select the data disk capacity. The data disk is mounted to the */var/lib/docker* directory of the container to store the image data and container data.

Logon Password and Confirm Password

Enter and confirm the logon password of the ECS instances. The password is 8-30 characters long and must contain uppercase letters/lowercase letters, numbers, and special characters at the same time. This password is required when you log on to the ECS console or log on to the ECS instance by using SSH.

> Note
>
> - Container Service uses this password to configure the ECS instances when creating the cluster, but does not save this password.
> - Keep this password properly for the initialization usage.

EIP

The Elastic IP (EIP) is used to access the Internet. By default, Container Service retains the EIP. If you select to not retain the EIP, the cluster releases the EIP after the instance initialization. You can access the Internet by using the What is NAT Gateway? on your own.

Server Load Balancer

An Internet Server Load Balancer instance is created by default if a cluster is created. The billing method is Pay-As-You-Go. The created Server Load Balancer instance is used to distribute the traffic to control the services and implement the service high availability.

Monitoring Plug-in

Select the check box to install the cloud monitoring plug-in on the ECS instances. Then, the operating system-level performance indicators of the ECS instances in the cluster can be monitored.

RDS Whitelist

You can select to add the IP addresses of the created nodes to the RDS instance whitelist, which facilitates the ECS instances to access the RDS instances.

- We recommend that you configure the RDS Whitelist when **Add** is selected for Add Node.
- If **Do not Add** is selected for Add Node and you want to configure the RDS Whitelist, add the existing ECS instances on the Create Cluster page. The RDS Whitelist cannot be configured if you create a zero-node cluster and add existing ECS instances after the cluster creation.
- The ECS instance must be in the same region as the RDS instance so that the IP address of the ECS instance can be added to the RDS instance whitelist.

Security Group

Container Service configures the default security group and only sets the inbound security group rules. You can configure the security group according to your business scenarios after the cluster is created successfully. For more information, see Container Service security group rules

- Ports 443 and 80 can be opened or closed as per your needs.
- We recommend that you retain the ICMP rules for communication between nodes and the convenience of troubleshooting. Some tools also depend on ICMP.
- Make sure you open all the ports you need. Otherwise, some services become inaccessible. The port that is accessed by using Server Load Balancer is not required to be opened.

3.5. Add an existing ECS instance

You can add a purchased Elastic Compute Service (ECS) instance to a specified cluster.

> **Note** At most 20 ECS instances can be added to a cluster by default. To add more ECS instances, open a ticket.

You can add an existing ECS instance in the following ways:

- **Add ECS instances automatically:** The image and system disk of the ECS instance are reset by using this method. You can add one or more ECS instances to the cluster at a time.
- **Add the ECS instance manually:** Manually add the ECS instance by running scripts on the ECS instance. You can only add one ECS instance to the cluster at a time.

Prerequisites

If you have not created a cluster before, create a cluster first.

Instructions

- The ECS instance to be added must be in the same region and use the same network type (Virtual Private Cloud (VPC)) as the cluster.
- When adding an existing ECS instance, make sure that your ECS instance has an Elastic IP (EIP) for the network type VPC, or the corresponding VPC has configured the NAT gateway. In short, make sure the corresponding node can access public network normally. Otherwise, the ECS instance fails to be added.
- The ECS instance to be added must be under the same account as the cluster.
- If you select to **manually add** the ECS instance, note that:

- If you have already installed Docker on your ECS instance, the ECS instance may fail to be added. We recommend that you uninstall Docker and remove the Docker folders before adding the ECS instance by running the following command:

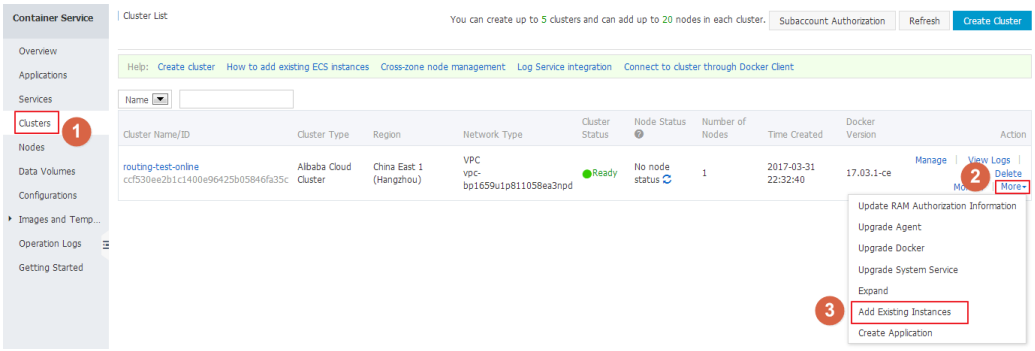
  Ubuntu: `apt-get remove -y docker-engine` , `rm -fr /etc/docker/ /var/lib/docker /etc/default/docker`

  CentOS: `yum remove -y docker-engine` , `rm -fr /etc/docker /var/lib/docker`

- Container Service nodes have special requirements for the operating system of the ECS instance. We recommend that you use Ubuntu 14.04/16.04 or CentOS 7 as the operating system. We have strictly tested the stability and compatibility of these operating systems.

Procedure

1. Log on to the Container Service console.
2. Click Swarm > **Clusters** in the left-side navigation pane.
3. Click **More** at the right of the cluster that you want to add ECS instances and then select **Add Existing Instances** from the drop-down list.



4. Add ECS instances.

   The ECS instances displayed are filtered and synchronized from your ECS instance list according to the region and network type defined by the cluster.

   Add the ECS instances in the following ways:

   - Add ECS instances automatically.

     > **Note** As this method will reset the image and system disk of the ECS instance, proceed with caution. Create a snapshot to back up your data before adding the ECS instance. For information about how to create a snapshot, see Create a snapshot of a disk.

     a. Select the ECS instances you want to add to the cluster and click **Next Step**.

        You can add one or more ECS instances at a time.

     b. Configure the instance information. Click **Next Step** and then click **Confirm** in the confirmation dialog box.

     c. Click **Finish**.

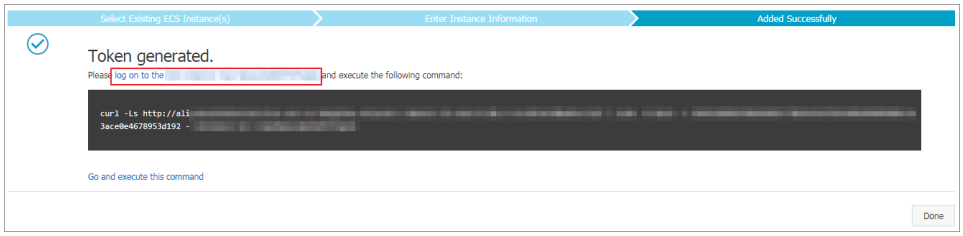
   - Manually add the ECS instance by running scripts on the ECS instance.

     a. Select **Manually Add**. Select an ECS instance, and then click **Next Step**.

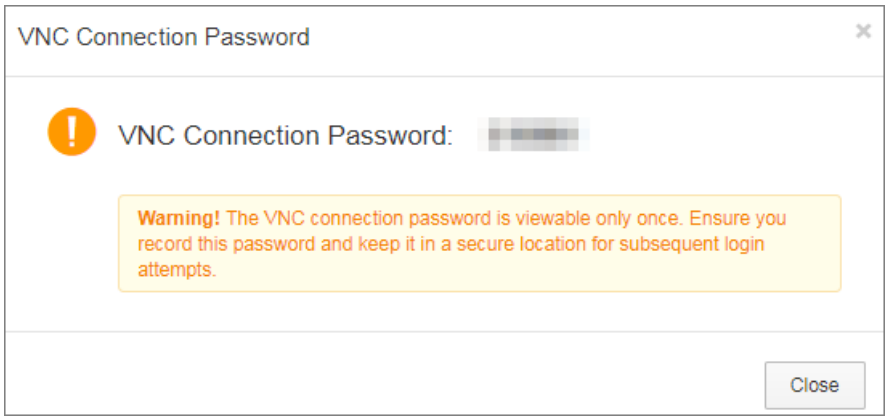
        You can only add one ECS instance at a time.

b. Confirm the instance information and click **Next Step**.

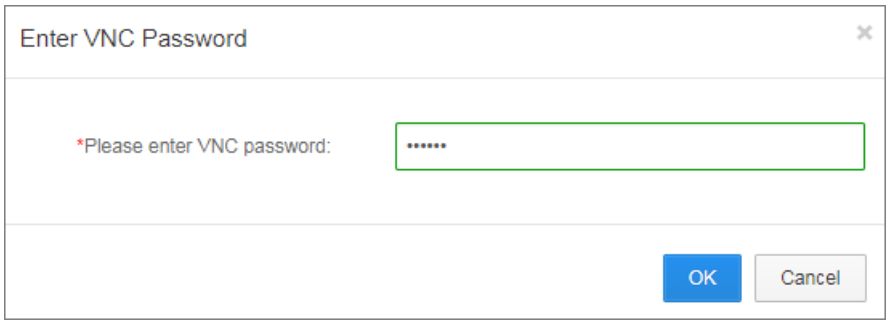
c. The scripts unique to this ECS instance are displayed. Click **log on to the ECS instance xxxxxx**.



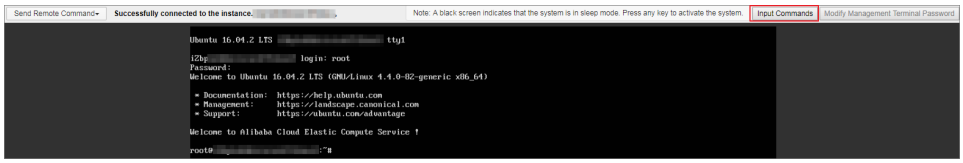
d. The VNC connection password is displayed in the dialog box. Copy the password and click **Close**.



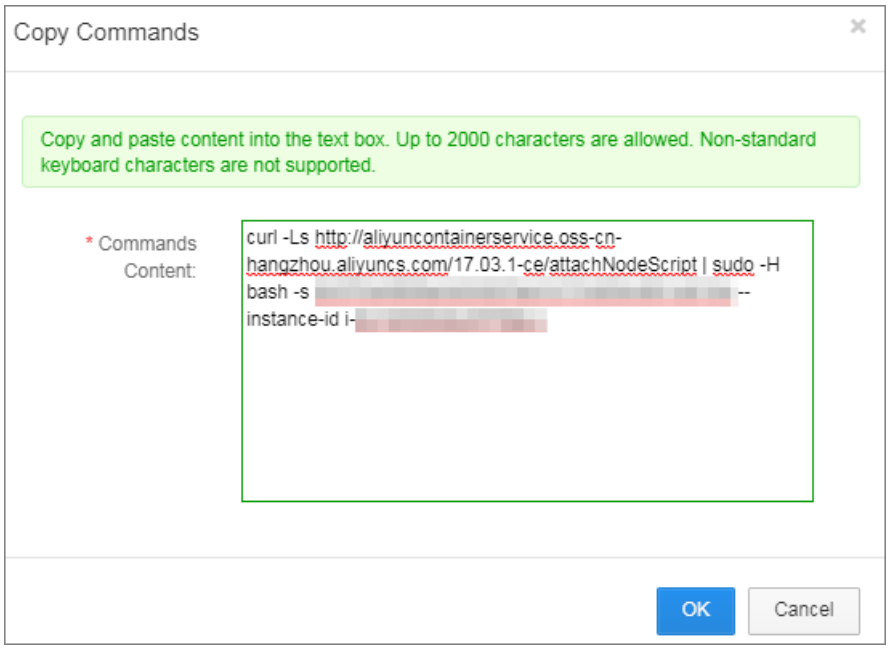
e. In the dialog box, enter the VNC connection password and click **OK**.



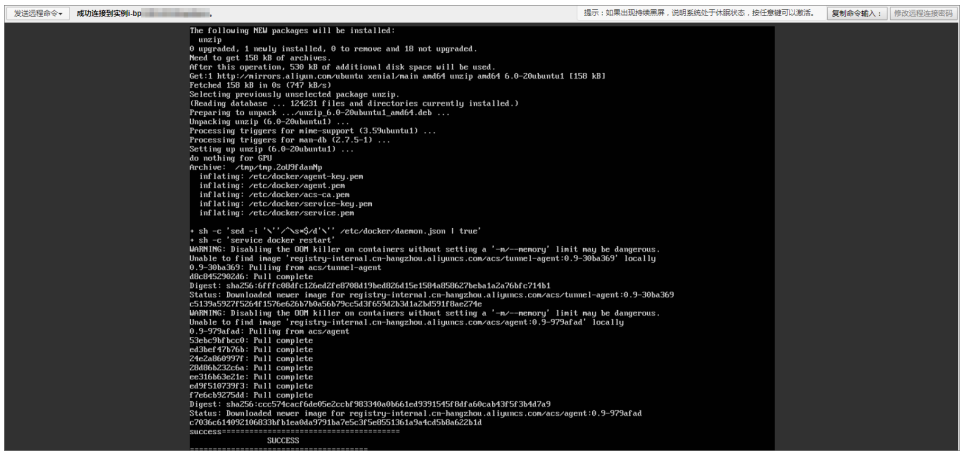
f. Enter the logon account (root) and password of the ECS instance, and press Enter to log on to the ECS instance.



g. Click **Input Commands**. Paste the preceding scripts into the dialog box, click **OK** and press Enter.

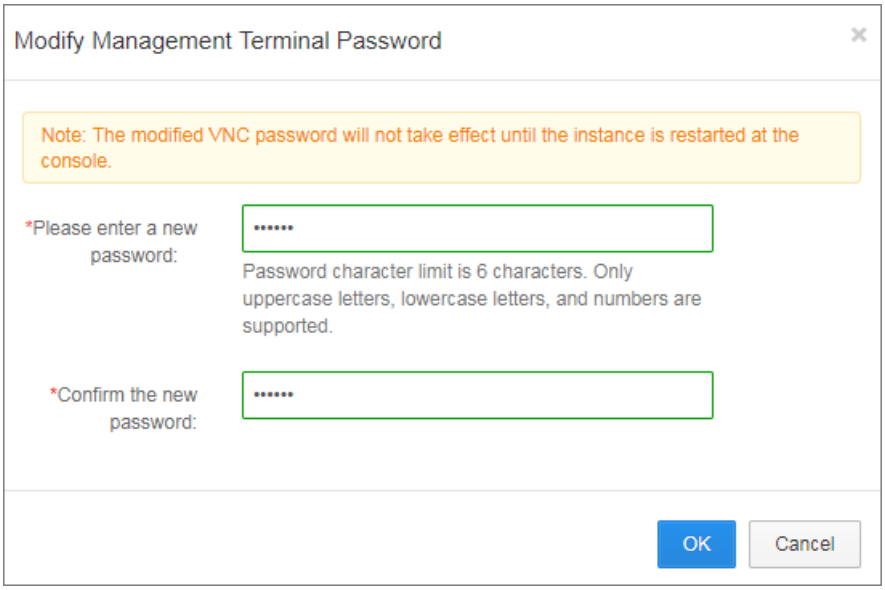


The system runs the scripts. Wait until the scripts are successfully run. A success message is displayed. The ECS instance is successfully added.



Related operation

You can modify the VNC connection password of the ECS instance in the remote terminal connection page. Click **Modify Management Terminal Password**, enter the new password and click **OK** in the dialog box.



3.6. Manage cross-zone nodes

To enhance the high availability of applications, you can distribute multiple nodes in different zones when creating a cluster.

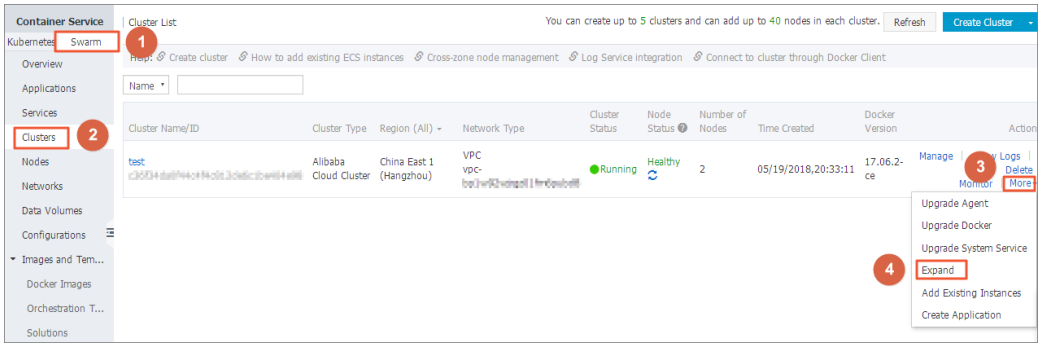
You can create a cluster with one node or a zero-node cluster. Then, add nodes of different zones by expanding the cluster or adding existing Elastic Compute Service (ECS) instances.

> **Note**
>
> - Nodes added by expanding the cluster are Pay-As-You-Go ECS instances.
> - Nodes added by adding existing ECS instances can be Pay-As-You-Go ECS instances or monthly/yearly subscription ECS instances.

Add nodes of different zones by expanding the cluster

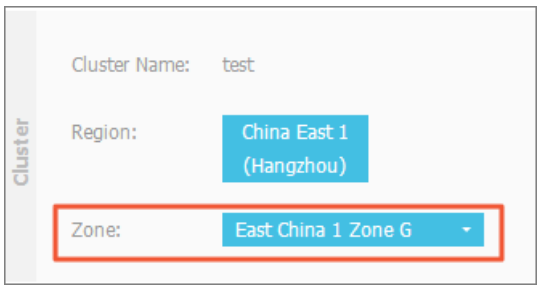
Procedure

1. Log on to the Container Service console.
2. Click Swarm > **Clusters** in the left-side navigation pane.
3. Click **More** at the right of the cluster that you want to expand and then select > **Expand** . As shown in the following figure.



4. The Expand page appears. Configure the specifications of the new nodes.

   You can create nodes of different zones by setting **Zone**.



5. Click **Expand** to add the new nodes to the cluster.
6. Repeat the preceding steps to create and add nodes of different zones to the cluster.

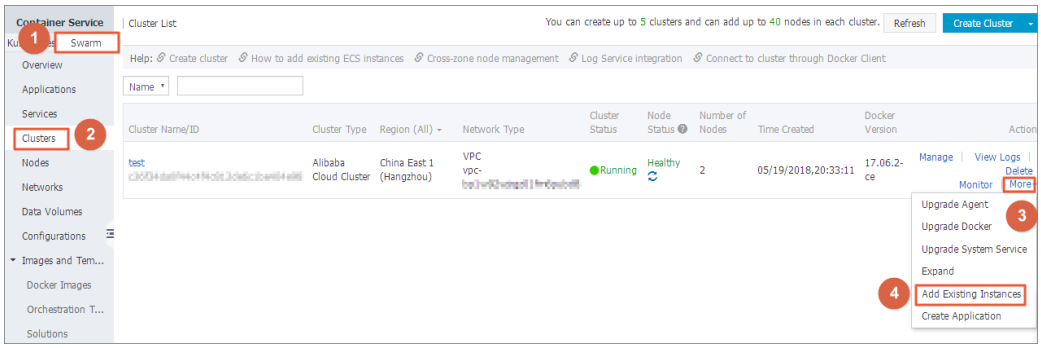
Add nodes of different zones by adding existing ECS instances

Prerequisites

To add nodes by using this method, purchase ECS instances from the ECS purchase page first, and select different zones for them during the purchase.

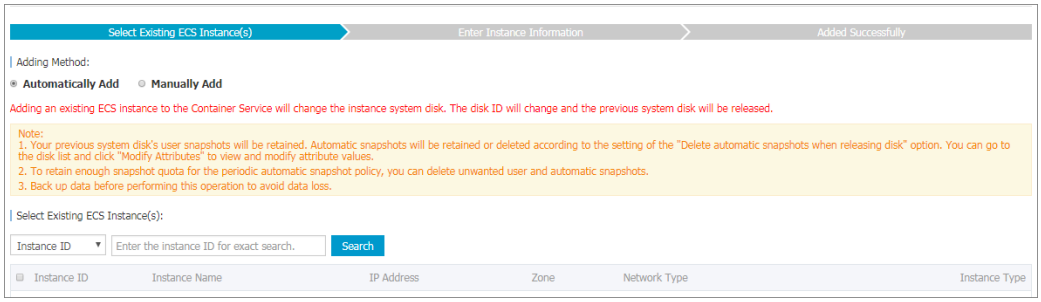
Procedure

1. Log on to the Container Service console.
2. Click Swarm > **Clusters** in the left-side navigation pane.
3. Click **More at the right of the cluster that you want to add existing ECS instances and then select > Add Existing Instances** . As shown in the following figure.



4. Select ECS instances of different zones and add them manually or automatically to the cluster.

   For more information, see Add an existing ECS instance.



5. Repeat the preceding steps to add ECS instances of different zones to the cluster.

3.7. Bind and unbind a Server Load Balancer instance

You can automatically create a Pay-As-You-Go Server Load Balancer instance when creating a cluster, or bind a monthly/yearly subscription or Pay-As-You-Go Server Load Balancer instance to a cluster after creating the cluster.

Container Service supports binding an Internet Server Load Balancer instance, a VPC Server Load Balancer instance, or an intranet Server Load Balancer instance in a classic network to a cluster.

Limits

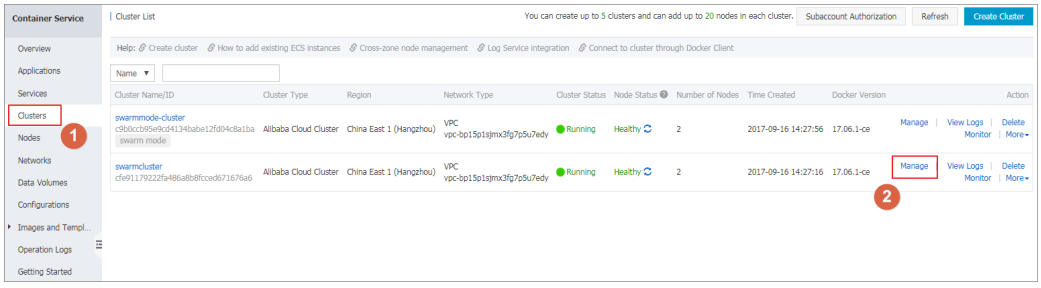
- You can only bind a Server Load Balancer instance to a cluster of the same region.
- You can only bind a Server Load Balancer instance to a cluster created by the same account.
- A VPC cluster can bind an Internet Server Load Balancer instance or a VPC Server Load Balancer instance.
- One cluster can only bind one Server Load Balancer instance.
- Two clusters cannot share one Server Load Balancer instance.

Prerequisites

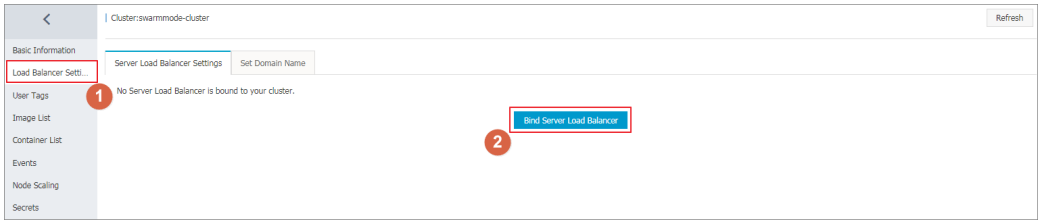
You have created a Server Load Balancer instance in the Server Load Balancer console and configured the TCP 9080 port for the instance to listen to backend servers.

Bind a Server Load Balancer instance

1. Log on to the Container Service console.
2. Click **Manage** at the right of the cluster that you want to bind a Server Load Balancer instance. The cluster details page appears.



3. Click **Load Balancer Settings** in the left-side navigation pane > and then click **Bind Server Load Balancer** .



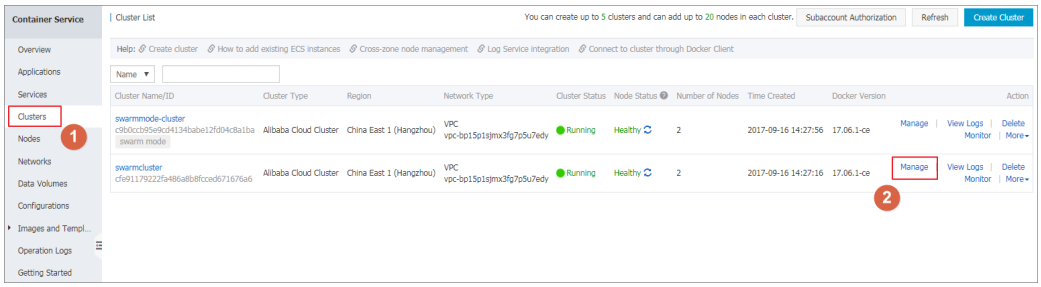
4. Select the Server Load Balancer instance that you want to bind to the cluster from the Server Load Balancer ID list and click **OK**.

> **Note** If the selected Server Load Balancer instance has been bound to a backend server, the system will prompt you that "This Server Load Balancer instance is already bound to a backend server". You need to select another Server Load Balancer instance that has not been bound to any backend server.

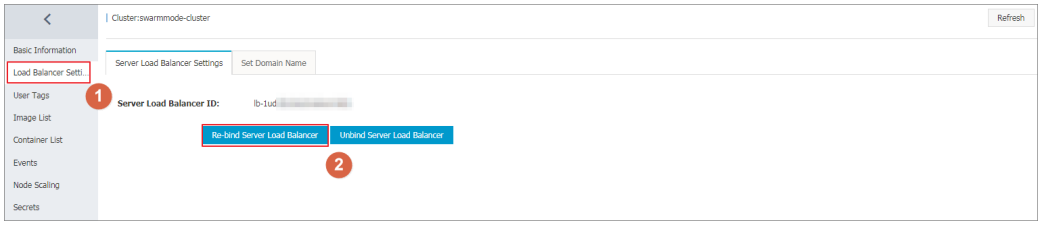
Rebind a Server Load Balancer instance

You can change the Server Load Balancer instance bound to your cluster per your needs.

1. Log on to the Container Service console.
2. Click **Clusters** in the left-side navigation pane.
3. Click **Manage** at the right of the cluster that you want to re-bind a Server Load Balancer instance. The cluster details page appears.



4. Click **Load Balancer Settings** in the left-side navigation pane, > and click **Re-bind Server Load Balancer** .

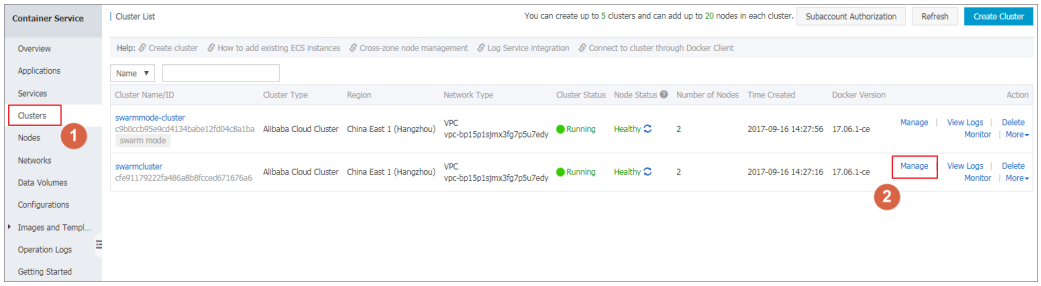


5. Select the Server Load Balancer instance that you want to bind to the cluster from the Server Load Balancer ID list and click **OK**.

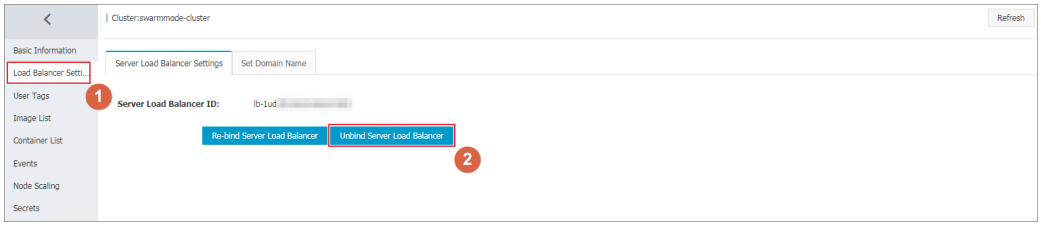
Unbind a Server Load Balancer instance

You can unbind a Server Load Balancer instance in the Container Service console if the instance is not required.

1. Log on to the Container Service console.
2. Click **Clusters** in the left-side navigation pane.
3. Click **Manage** at the right of the cluster that you want to unbind a Server Load Balancer instance. The cluster details page appears.



4. Click **Load Balancer Settings** in the left-side navigation pane, > and click **Unbind Server Load Balancer** .



3.8. Set the root domain name of a cluster

Context

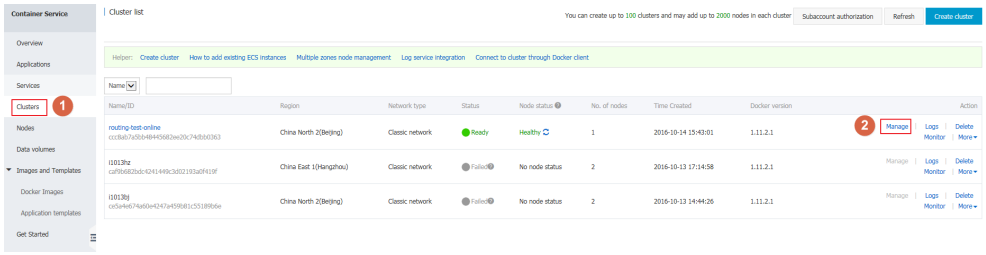
When you Create an Nginx webserver from an image and configure the web routing rules, you are only required to enter the domain name prefix `nginx` . Then, you can obtain `the domain name in the format of` $cluster_id.$region_id.alicontainer.com. You can replace this domain name by setting a root domain name (`51ili.com` is used in this example) of the cluster. When you redeploy the application `nginx` , the domain name changes from `nginx.cd5b226071936493b89e75bbe8841664c.cn-hangzhou.alicontainer.com to nginx.51ili.com` , which makes it convenient for you to access the cluster applications with your own root domain name.

> **Note** To guarantee the normal operation of the following example, upgrade the Agent to the latest version first.

Procedure

1. Bind a Server Load Balancer instance.
   i. Log on to the Container Service console.
   ii. Log on to the Container Service console.
   iii. Click **Clusters** in the left-side navigation pane.

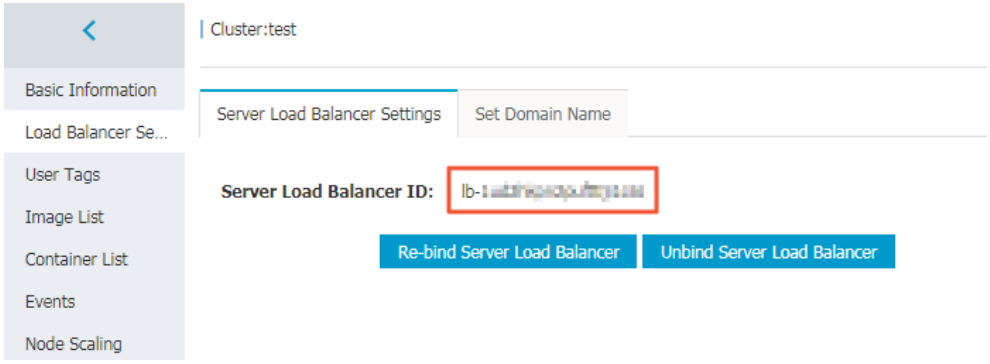
iv. Click **Manage** at the right of the cluster (`routing-test-online` in this example) that you want to configure.



v. Click **Load Balancer Settings** in the left-side navigation pane.

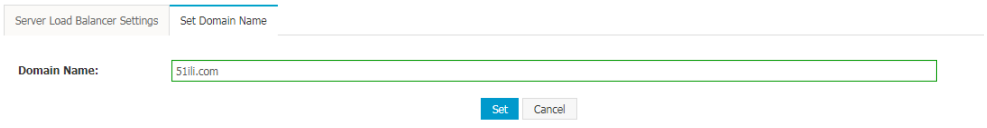
If no Server Load Balancer instance is bound to this cluster, log on to the Server Load Balancer console and create a Server Load Balancer instance. Then, return to this page and bind the instance to this cluster.

> **Note** For more information about how to bind and unbind a Server Load Balancer instance to and from a cluster and the limits in Container Service, see Bind and unbind a Server Load Balancer instance.



2. Set the domain name.

i. Click the Set Domain Name tab and enter the root domain name you bought in the Domain Name field. In this example, 51ili.com is entered.

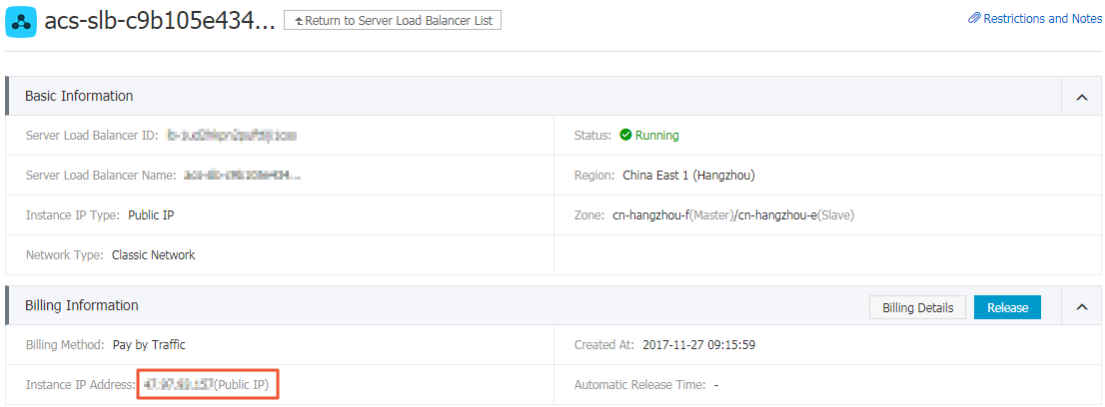


ii. Click **Set**.

3. Resolve the domain name to the bound Server Load Balancer instance.

i. Log on to the Server Load Balancer console. Click Instances in the left-side navigation pane, and then click the ID of the Server Load Balancer instance bound to the cluster routing-test-online.

ii. View the instance details. Find the instance IP address.



iii. Log on to the Alibaba Cloud DNS console and add record A to resolve `*.51ili.com` to the Server Load Balancer VIP address.

4. Redeploy the nginx application.

i. Click Redeploy at the right of nginx. The service access endpoint of the application nginx is changed.

The access endpoint before setting the root domain name.

The access endpoint after setting the root domain name.

ii. Access the latest access endpoint http://nginx.51ili.com.

3.9. Download cluster certificate

Context

With the downloaded certificate, you can connect to the endpoint exposed from the cluster by using Docker Swarm API or Docker client. For more information, see Connect to a cluster by using Docker tools.

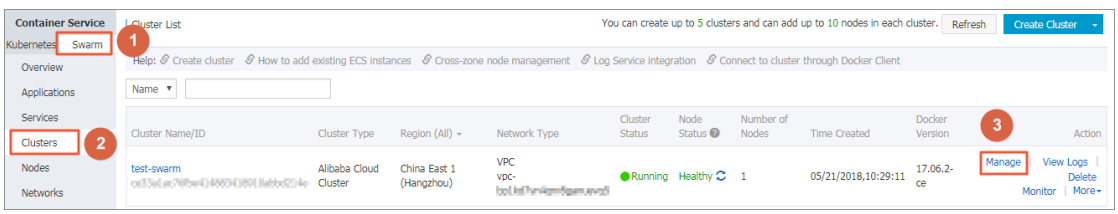
Procedure

1. Obtain the access address.

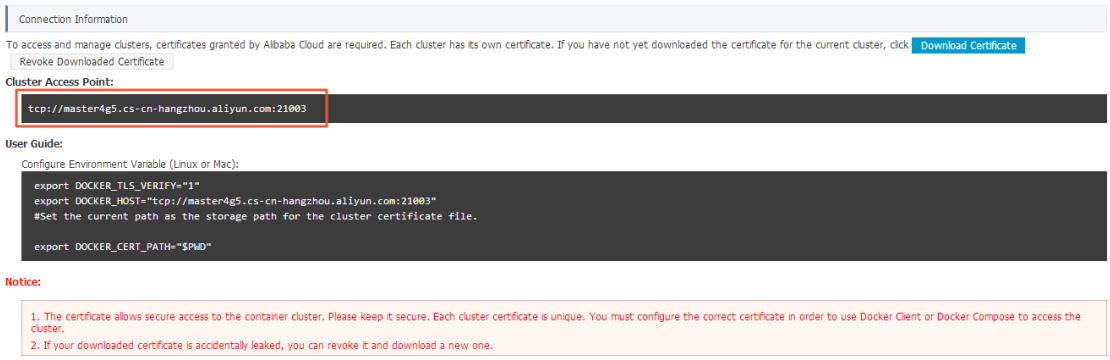
i. Log on to the Container Service console.

ii. Log on to the Container Service console.

iii. Click **Clusters** in the left-side navigation pane. On the Cluster List page, click **Manage** at the right of a cluster.



iv. The cluster details page is displayed, showing the cluster connection information.



2. Download and save the TLS certificate.

   Configure a TLS certificate before you use the preceding access address to access the Docker cluster.

   Click **Download Certificate** in the cluster details page to download the TLS certificate. The certFiles.zip file is downloaded. certificate. The `certFiles.zip` file is downloaded. In the following example, the downloaded certificate is saved to the *~/.acs/certs/ClusterName/* directory. `ClusterName` indicates the name of your cluster. You can save the certificate to a different directory, but we recommend using the *~/.acs/certs/ClusterName/* directory for easy management.

   ```
   mkdir ~/.acs/certs/ClusterName/ #Replace ClusterName with your cluster name
    cd ~/.acs/certs/ClusterName/
    cp /path/to/certFiles.zip .
    unzip certFiles.zip
   ```

   The *certFiles.zip* file contains *ca.pem*, *cert.pem*, and *key.pem*.

3.10. Expand a cluster

Prerequisites

A cluster can contain up to 20 nodes.

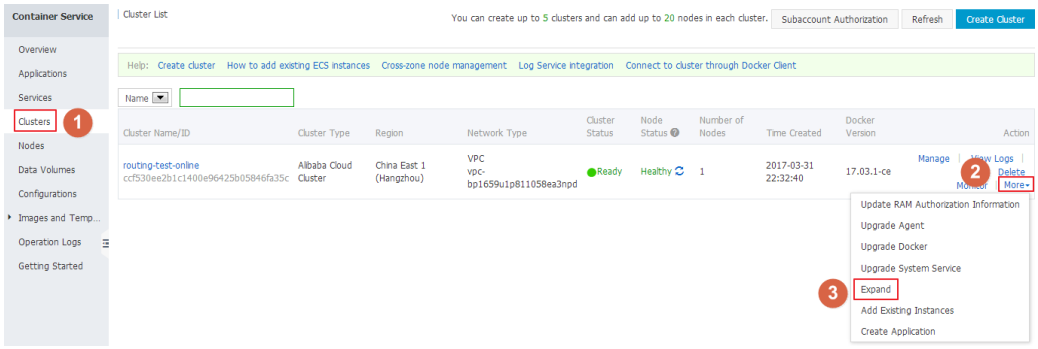
Context

You can expand your cluster according to your business needs.

> **Note** Elastic Compute Service (ECS) instances added by expanding the cluster are Pay-As-You-Go instances.

Procedure

1. Log on to the Container Service console.
2. Click **Clusters** in the left-side navigation pane.
3. On the Cluster List page, click **More** at the right of the cluster that you want to expand and then select **Expand** from the list.



4. In the displayed dialog box, configure the specifications of the new node.

   You can select the number and the specifications of the ECS instances you are about to add to the cluster.

5. Click Expand.

3.11. Migrate a cluster

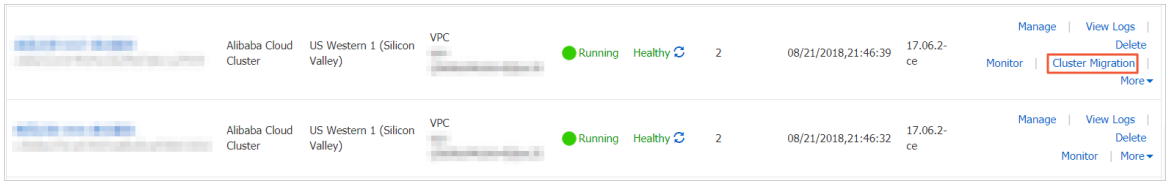
For a Swarm cluster created earlier, you can guarantee the performance and stability of the cluster by migrating the cluster.

Context

- The latest time for migrating a cluster is displayed through SMS, station message, or email. Complete the Swarm cluster migration before the latest time. The system automatically migrates the cluster if you do not migrate the cluster before the latest time.
- Cluster migration rebuilds connections from cluster nodes to the container server without affecting applications deployed in the cluster, nor adding or modifying any data. Make sure that you perform this operation during the low peak period of your business because unpredictable risks might still exist throughout the migration process.

Procedure

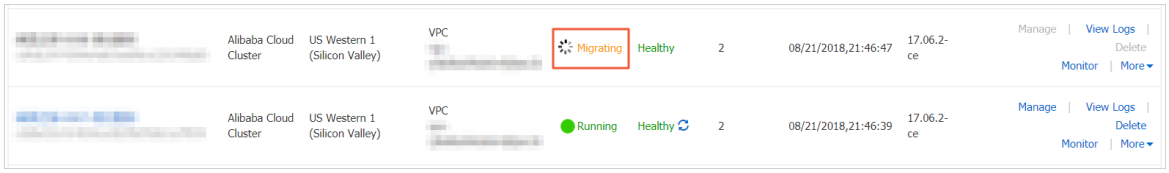
1. Log on to the Container Service console.
2. Under the Swarm menu, click **Clusters**.
3. Click **Cluster Migration** in the action column at the right of the cluster to be migrated.



4. Click **OK** in the **Prompt** dialog box.

> **Note** During cluster migration:
>
> - Information query, deployment, upgrade, and other operations cannot be performed in the console.
> - The cluster cannot be connected to through the cluster access point API.
> - The data and application status in the cluster remain unchanged. Applications deployed on the cluster are still accessible.
> - The migration process takes about three minutes.

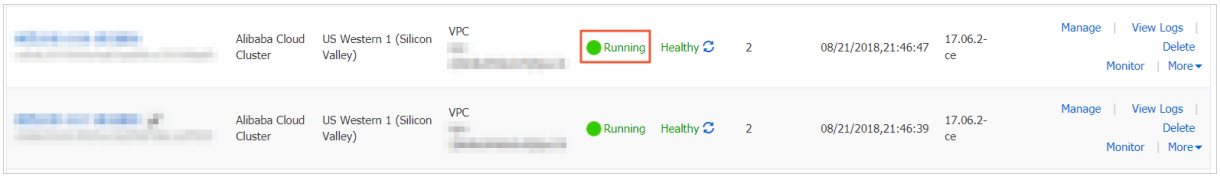
On the **Cluster List** page, **Migrating** is displayed in the **Cluster Status** column.



Result

After cluster migration is completed, on the **Cluster List** page, **Running** is displayed in the **Cluster Status** column.

> **Note**
>
> - The cluster ID, access point address, and other attributes remain unchanged.
> - Please be sure to confirm that your business is running properly.
> - During the migration process, if you have any questions, please open a ticket in which you include the cluster ID and state whether your deployed applications are normal.

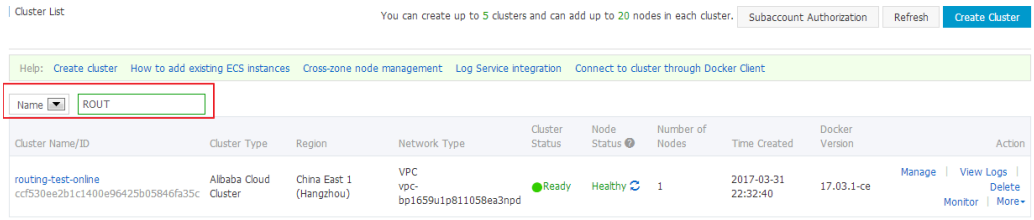


3.12. Search for a cluster

Procedure

1. Log on to the Container Service console.
2. Click Swarm > **Clusters** in the left-side navigation pane.
3. Enter the cluster name or keywords of the cluster name in the search box. Clusters with the keywords in their names are displayed. As shown in the following figure.

> **Note** The search is case insensitive.



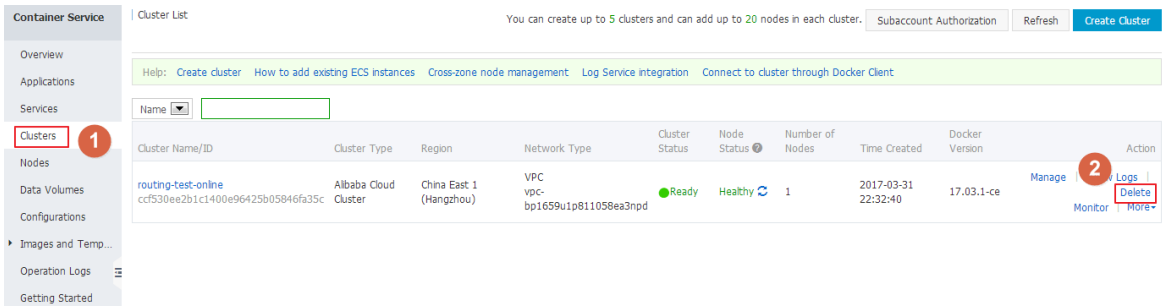
3.13. Delete a cluster

Context

You can delete clusters from Container Service. Deleting the cluster also deletes its associated Elastic Compute Service (ECS) instances, Server Load Balancer instance, and other cloud resources, so proceed with caution.

Procedure

1. Log on to the Container Service console.
2. Click Swarm > **Clusters** in the left-side navigation pane.
3. Click **Delete** at the right of the cluster you are about to delete.



4. In the displayed window, select whether or not to keep the Server Load Balancer instance and click **OK**.

3.14. Clean up a cluster disk

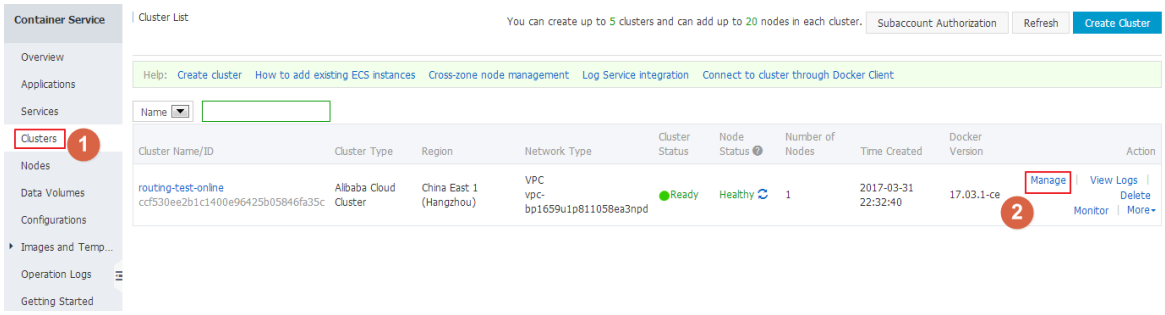
Context

Cleaning up disk clears the dirty data on each server in your cluster. Dirty data is limited to:

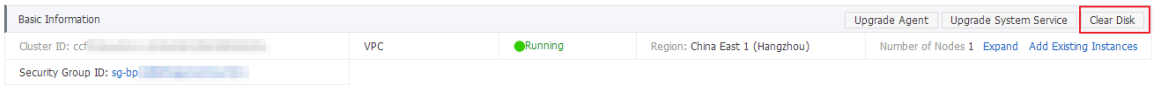
- Docker images downloaded locally but not used.
- Volume directory once attached to a container but not cleaned up after the destruction of the container.

Procedure

1. Log on to the Container Service console.
2. Click **Clusters** in the left-side navigation pane.
3. On the Cluster List page, click **Manage** at the right of the cluster that you want to clean up the disk.



4. Click **Clear Disk** on the cluster details page.



3.15. Log on to image repository

Prerequisites

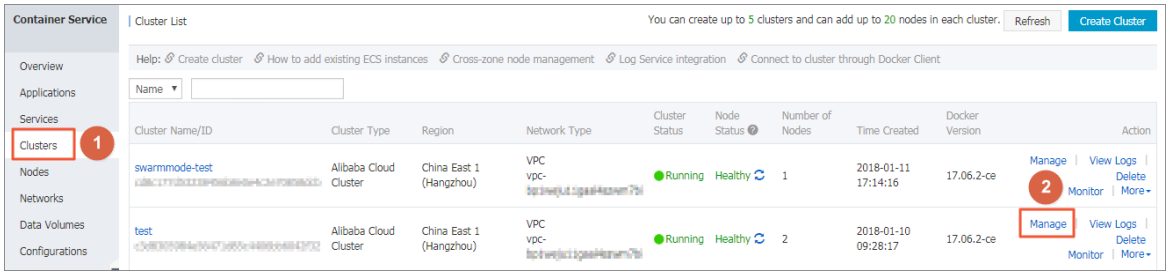
- Prepare an available image repository. Use the Docker Hub official service in this example, which requires you to register a Docker ID and build an available repository in it.
- Configure the independent logon password for the repository. In this example, log on to the Container Registry console to configure or modify the repository logon password. Note that you are configuring the password when you modify the repository logon password for the first time.

Context

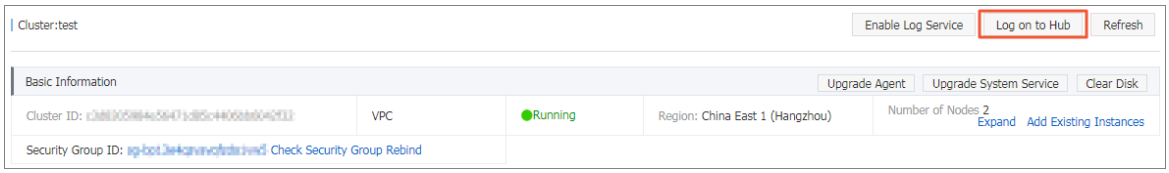
You can log on to the image repository in a cluster to provide the related cluster logon information, which facilitates you to manage clusters by using cluster management tools.

Procedure

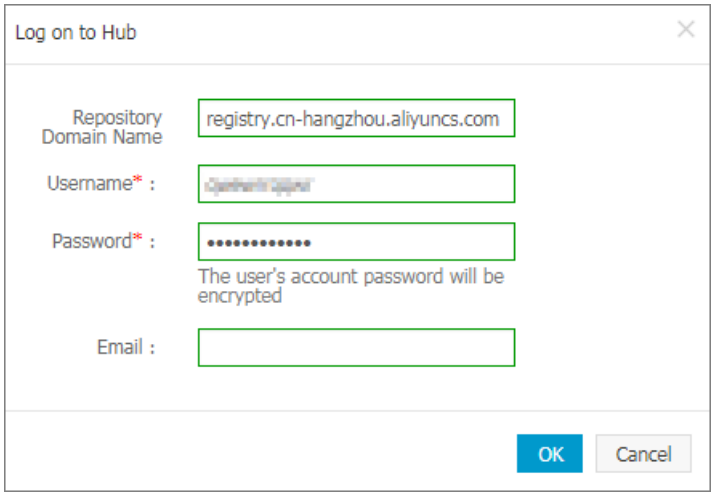
1. Log on to the Container Service console.
2. Click **Clusters** in the left-side navigation pane.
3. Click **Manage** at the right of the cluster you want to configure.



4. Click **Log on to Hub**.



5. Configure the parameters in the displayed dialog box.



- Repository Domain Name: Enter the hub domain name of the image repository. Take the image address `registry.cn-hangzhou.aliyuncs.com/acs/agent:0.8` as an example. The repository domain name is `registry.cn-hangzhou.aliyuncs.com` .
- Username: Enter the username of the image repository. In this example, enter the Docker ID registered in Docker Hub.
- Password: Enter the independent logon password of the image repository. In this example, enter the logon password set when you registered in Docker Hub. Registry's login password is set and modified on the container mirroring Service's console.
- Email: Enter the email set when you registered the image repository. In this example, enter the email set when you registered in Docker Hub.

6. Click **OK**. You have successfully logged on to the image repository if no error message appears.

3.16. Upgrade Agent

Context

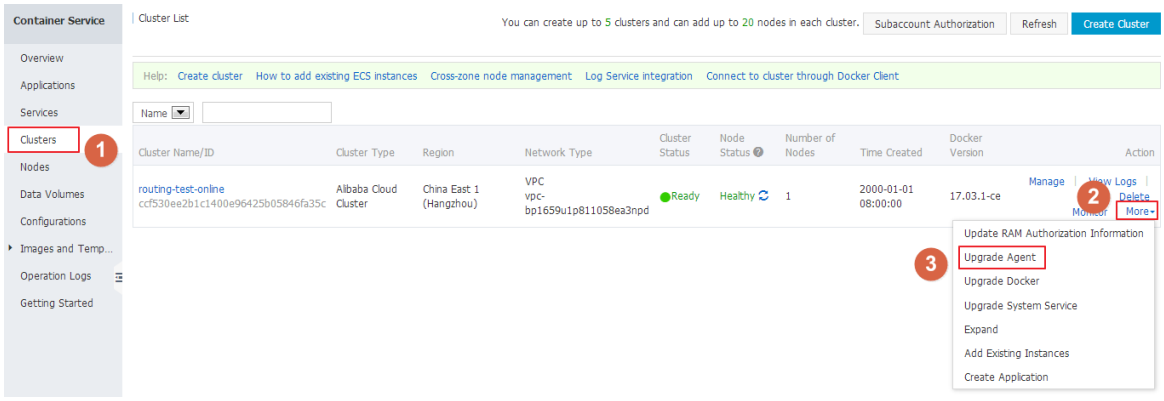
> **Note** Your applications are not affected during the upgrade, but you can neither manage the cluster by using the Web interface, nor use Docker client to connect to the cluster access port for about 2 minutes.

The Agent of Container Service, which is installed on each server in the cluster, receives commands issued by the Container Service control system.

New functions are regularly added to Container Service. If you need the latest functions, upgrade the Agent of the cluster.

Procedure

1. Log on to the Container Service console.
2. Click **Clusters** in the left-side navigation pane.
3. On the Cluster List page, **click More at the right of the cluster that you want to upgrade the Agent > and then select Upgrade Agent from the list** .



4. Click **OK** in the displayed dialog box.

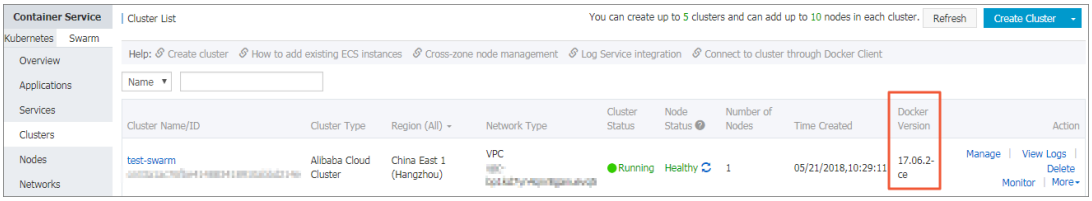
3.17. Upgrade Docker daemon

Context

Standard Docker daemon is installed on each server in the cluster to manage containers.

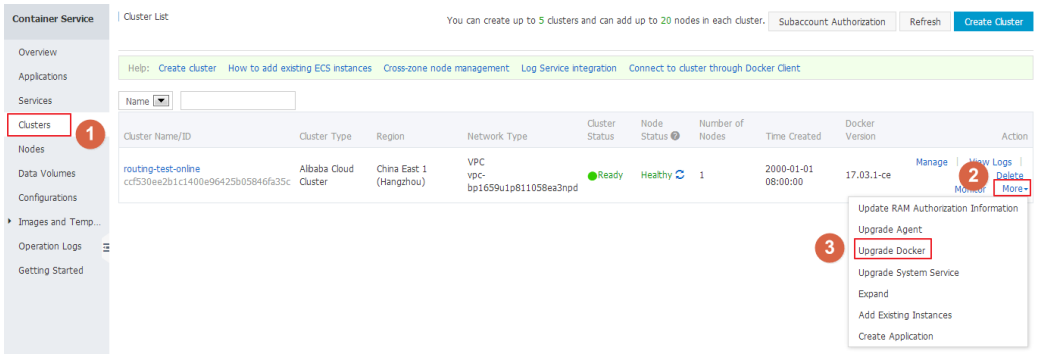
> **Note**
>
> - The cluster Docker daemon upgrade requires that the machine is able to access the Internet to download necessary software packages.
> - The cluster Docker daemon upgrade may fail. To guarantee your data security, we recommend that you back up snapshots before upgrading Docker daemon.
> - During the cluster Docker daemon upgrade, the services deployed on the cluster are interrupted and you cannot perform operations on the cluster and applications. Make appropriate arrangements before the upgrade. The upgrade lasts 3–30 minutes. The cluster status changes to Running after the upgrade.

You can view the Docker version of the cluster on the Cluster List page.



Procedure

1. Log on to the Container Service console.
2. Under Swarm, click **Clusters** in the left-side navigation pane.
3. On the Cluster List page, click **Upgrade** in the Docker Version column, or click **More > Upgrade Docker** at the right of the cluster.



4. On the Upgrade Docker page, click **Upgrade Agent** to upgrade the Agent first if your Agent is not in the latest version.

5. If your Agent is in the latest version, upgrade Docker daemon

   in the following ways:

   - Upgrade Directly

     Click **Upgrade Directly** to enter the Docker Engine upgrade process.

   - Back up Snapshot before Upgrade

     We recommend that you back up the snapshots before upgrading Docker daemon. In this way, you can recover Docker daemon by using the snapshots if an error occurs during the upgrade process.

     Click **Back up Snapshot before Upgrade**, and then the system calls the Elastic Compute Service (ECS) API to take snapshots of the cluster nodes.

     Backing up snapshots may take some time. Wait until the snapshots are backed up, and then the system automatically enters the Docker Engine upgrade process.

     If the snapshots failed to be backed up, you can click **Continue** or **Quit**. Click **Continue** to enter the Docker Engine upgrade process, or click **Quit** to give up the upgrade.

What's next

Return to the **Cluster List** page and you can see that the cluster you upgraded the Docker daemon is in the `Docker-Engine is upgrading` status. This may take a while as container data will be backed up during the upgrade of the Docker Engine.

3.18. Upgrade system services

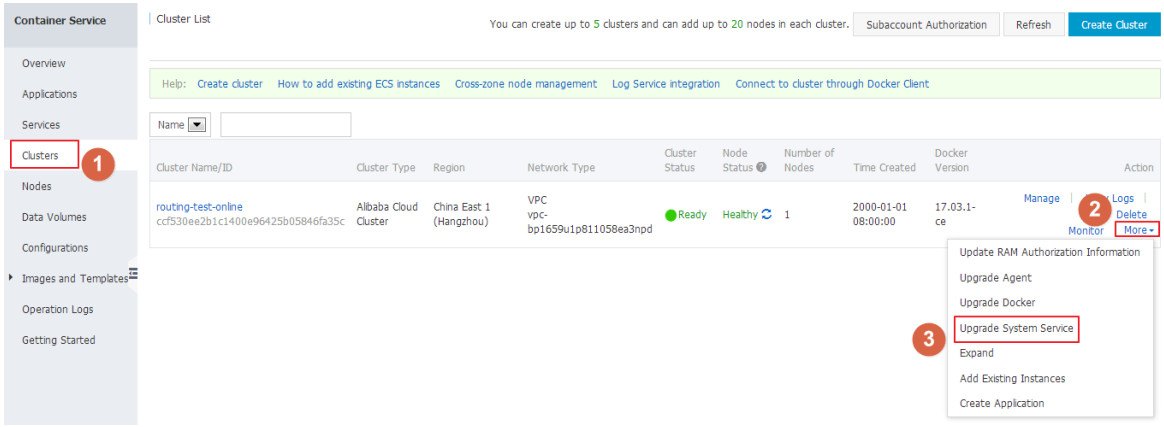
Context

The system services of a cluster, including Log Service `acslogging`, Simple Routing Service `acsrouting`, Monitor Service acsmonitoring, and Volume Service `acsvolumedriver`, are used to deal with general services necessary for applications. This document introduces how to upgrade these system services.

> **Note** During the upgrade of the cluster system services, your applications or services may be temporarily inaccessible or abnormal, so proceed with caution. We recommend that you upgrade the system services when the access traffic is low or at the maintenance time.

Procedure

1. Log on to the Container Service console.
2. Under Swarm, click **Clusters** in the left-side navigation pane.
3. On the Cluster List page, click **More** at the right of the cluster whose system services you want to upgrade and then select **Upgrade System Service** from the drop-down list. As shown in the following figure.



4. The Upgrade System Service dialog box opens. Select the system services you want to upgrade and click **Upgrade**.

   For example, select **Simple Routing Service** (corresponding to acsrouting; note that the upgrade will temporarily affect your access to applications) and **Volume Service** (corresponding to acsvolumedriver; note that the upgrade might temporarily affect the functions of your associated applications).

   Click **Applications** in the left-side navigation pane and select the cluster from the Cluster drop-down list. You can see the system services are being upgraded.

   After the upgrade, the affected services and applications resume normal functioning.

4.Nodes

4.1. Remove a node

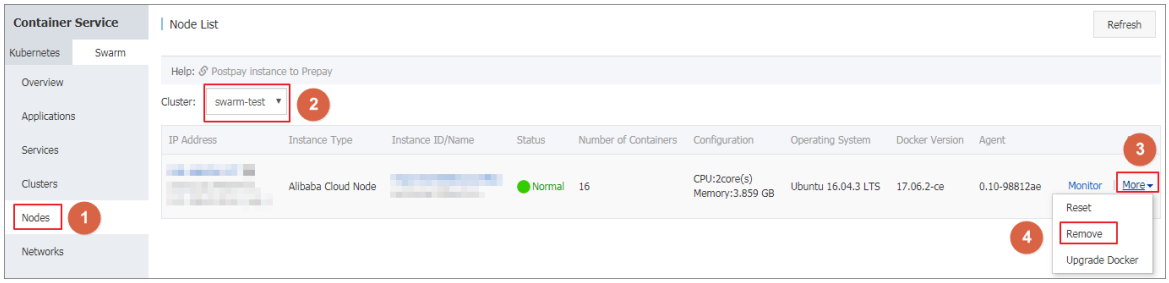
Context

You can remove nodes from a cluster. Deleting a node removes the machine from the cluster. The machine information of the removed node cannot be viewed in the node list.

> **Note**
>
> - Back up the data before removing a node.
> - Deleting the node only removes the Elastic Compute Service (ECS) instance from the cluster. The ECS instance is not released. To release the ECS instance, go to the ECS console and manually release the instance.

Procedure

1. Log on to the Container Service console.
2. Click **Nodes** in the left-side navigation pane.
3. Select the cluster where the node you want to remove resides from the Cluster list.
4. Click More**at the right of the node you want to remove and then > select Remove from the list .**



5. In the displayed confirmation dialog box, click **Confirm**.

4.2. Reset a node

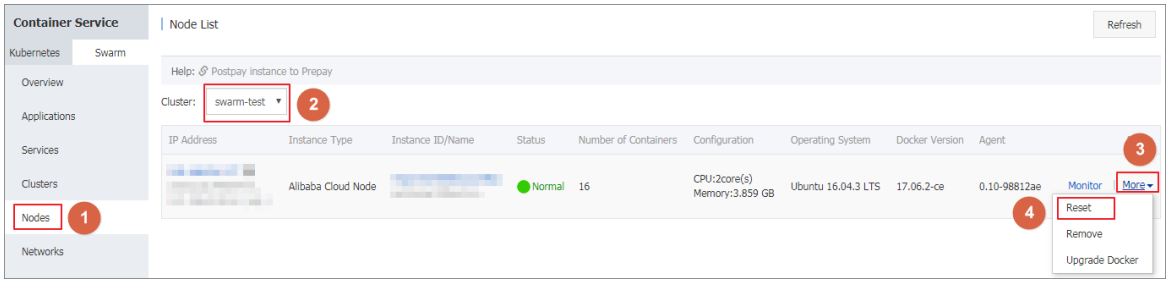
Context

You can reset a node in a cluster. The system disk of the corresponding machine (the reset node) is replaced and the data stored in the system disk is lost. The reset machine is re-added to the cluster.

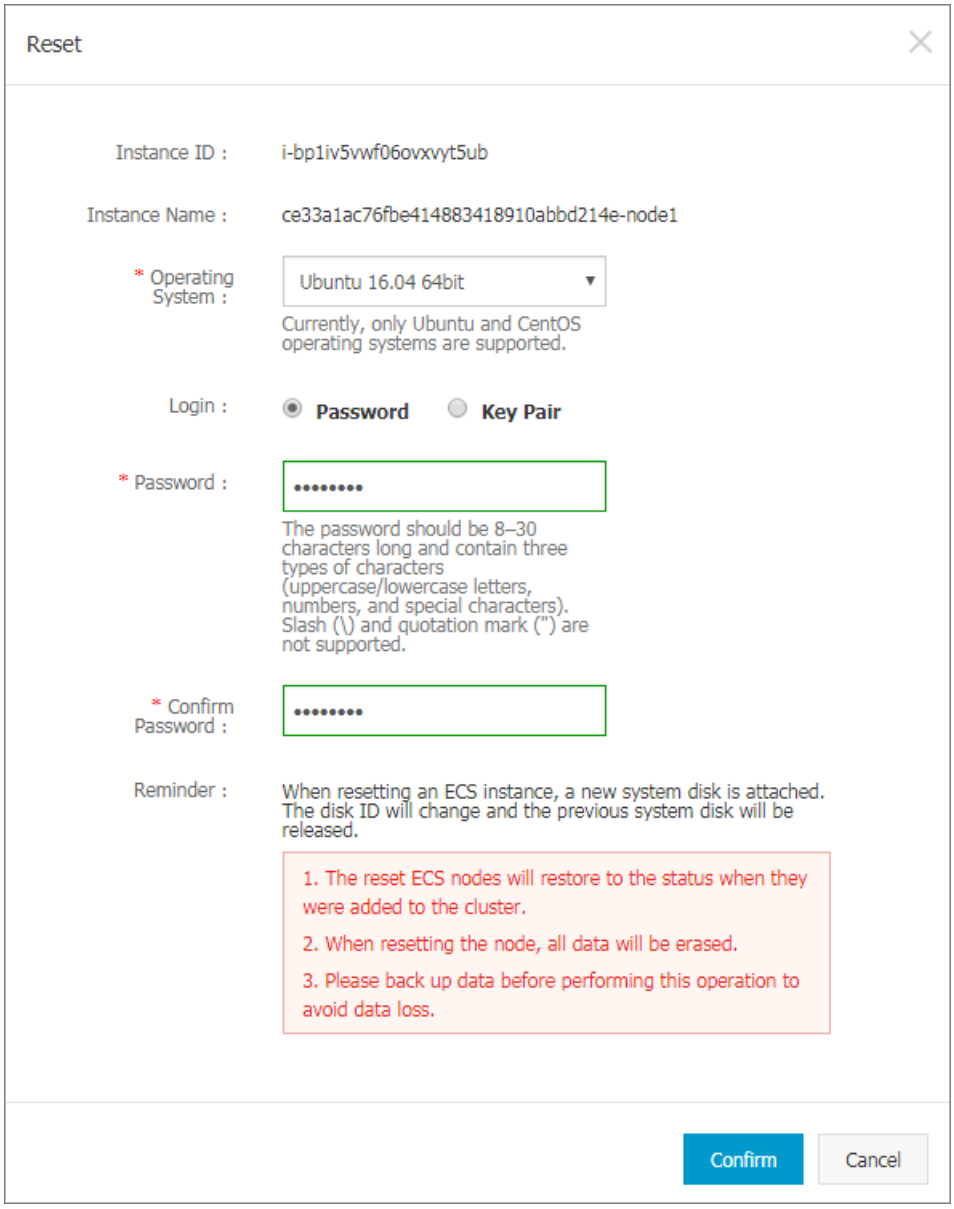
> **Note**
>
> - Resetting an Elastic Compute Service (ECS) instance changes the ECS system disk. The disk ID is changed and the previous system disk is released.
> - The reset ECS instance is restored to the status when it was added to the cluster.
> - All data is cleaned up when resetting the node.
> - Back up your data before resetting the node to avoid data loss.

Procedure

1. Log on to the Container Service console.
2. Click Nodes in the left-side navigation pane.
3. Select the cluster where the node you want to reset resides from the Cluster list.
4. Select the cluster where you want to reset the nodes.**Click More at the right of the node you want to > reset and then select Reset from the list .**



5. In the confirmation window, enter the instance logon password, and click Confirm.In the pop-up confirmation dialog box, fill in the login password for the instance and click OK.



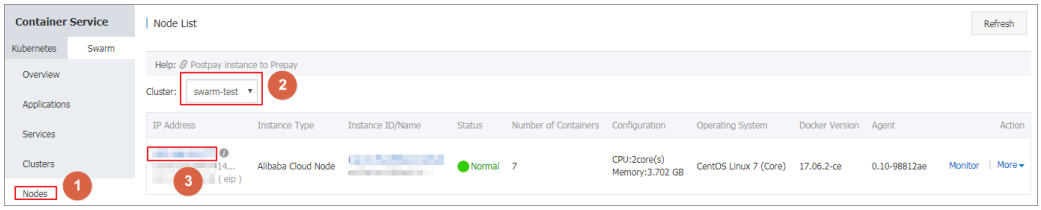
4.3. View containers running on a node

Context

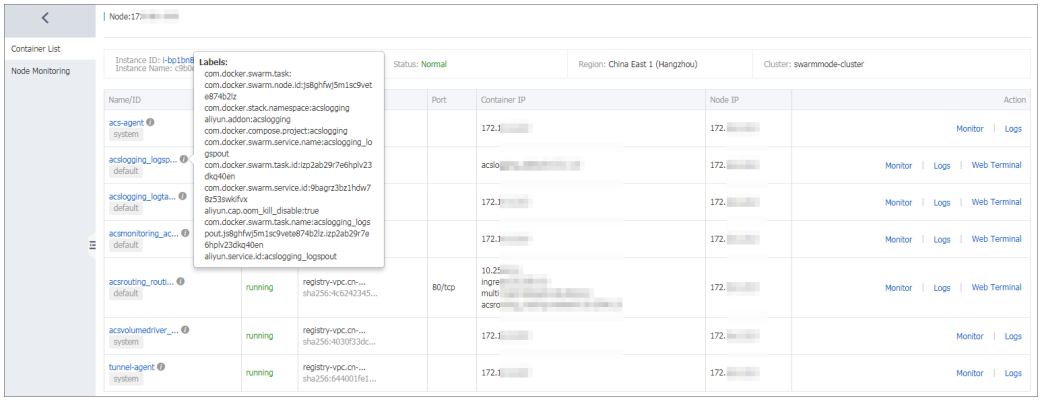
You can view containers running on a node on the Node List page.

Procedure

1. Log on to the Container Service console.
2. Click Swarm > **Nodes** in the left-side navigation pane.
3. On the Node List page, select a cluster from the Cluster drop-down list.
4. Click the node ID.



You can see the list of containers running on the node.



What's next

In the list, you can view the labels, images, the image SHA256 values, logs, and monitoring information of containers and perform operations on containers, including starting and stopping containers, deleting containers, and operating on containers on a remote terminal.

4.4. Update a node certificate

You can update a node certificate of a Swarm cluster to avoid node certificate expiration.

Prerequisites

1. You have created a swarm cluster, see Create a cluster.
2. Updating a node certificate reboots the node Docker Daemon. Make sure that containers on the node are all configured to restart automatically.

   > **Note** You can configure a container restart policy when creating an application. When you create an application by using an image, select the **Always** check box for **Restart**. When you create an application by using a template, configure a container restart policy in the template `restart: always`.

3. If a node certificate expires within 60 days, a prompt is displayed. You must timely update the node certificate.

Context

Each cluster node has a certificate used to access system control services. Each issued certificate has a valid period. When the valid period of a certificate is about to expire, you must manually renew the certificate. Otherwise, the service of the node is affected.

Procedure

1. Log on to the Container Service console.
2. Under the Swarm menu, click **Nodes** in the left-side navigation pane. The certificate expiration information of each cluster node is displayed.

   > **Note** The certificate expiration time is displayed in the status column only if the node certificate expires within 60 days.

3. Select a node in the node list, and click **More > Update Certificate** on the right to reissue the node certificate.

   > **Note** We recommend that you upgrade the cluster agent to the latest version before updating the node certificate.

4. (Optional)If the system prompts you to upgrade the cluster agent after you click **Update Certificate**, the current cluster agent does not support this feature. You need to upgrade the cluster agent to the new version first, see Upgrade Agent. If no prompt is displayed, go to the next step.
5. If no prompt is displayed or the cluster agent is updated, click **Update Certificate**. Confirm updating information and then update the node cluster certificate.

   > **Note**
   > - When the node certificate update is completed, the Docker Daemon node is automatically restarted about 1 minute later.
   > - To guarantee that containers on the node can automatically restart, make sure that an automatic restart policy is configured.

6. After the cluster node certificate is updated, the node certificate information is no longer displayed.

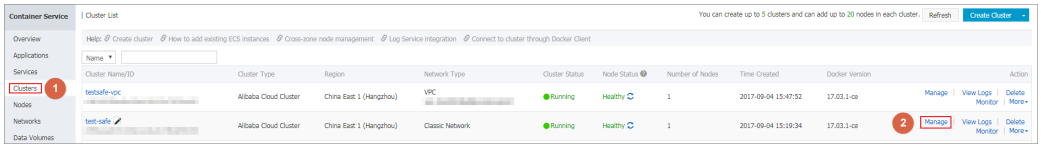
5.Security groups

5.1. Container Service security group rules

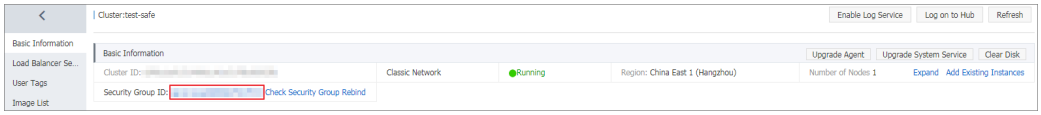
View security group rules

Procedure

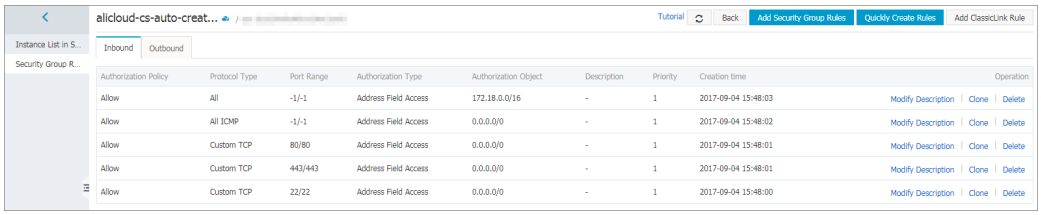
1. Log on to the Container Service console.
2. Click **Clusters** in the left-side navigation pane.
3. On the Cluster List page, click **Manage** at the right of a cluster.



4. Click the security group ID to jump to the details page of this security group on the Elastic Compute Service (ECS) console.



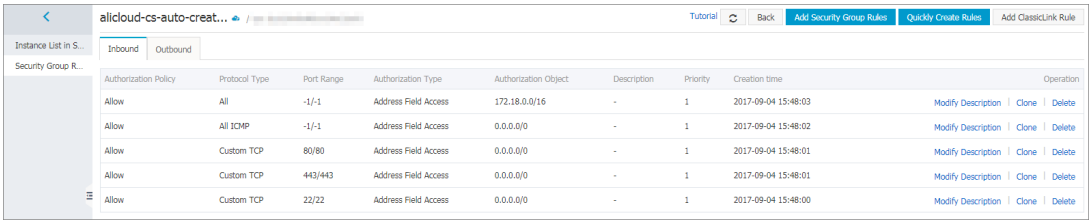
5. Click **Security Group Rules** in the left-side navigation pane. You can view the security group rules.



Security group rules

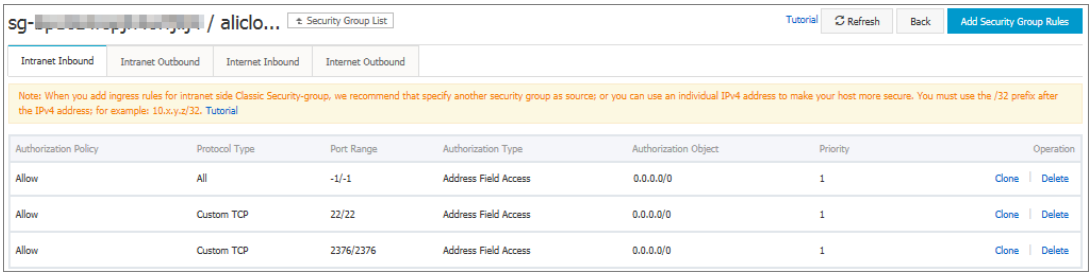
For the Container Service clusters created after February 28, 2017, the security groups created by default have been reinforced. Alibaba Cloud Container Service only sets the inbound security group rules. Container Service only supports creating clusters in the VPC environment since January 1, 2018. The opening rules of VPC clusters are as follows.

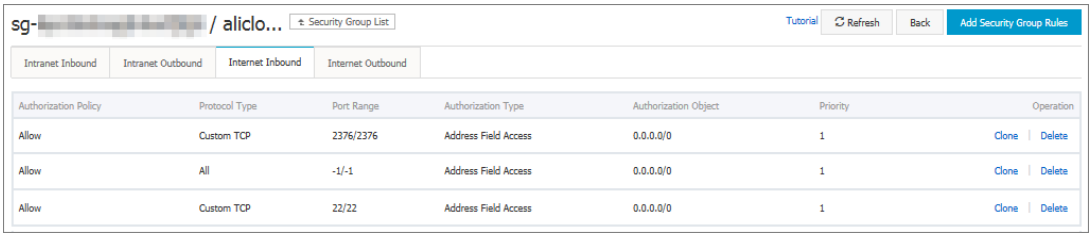
Virtual Private Cloud (VPC) security group:



> **Note**
>
> - Ports 443 and 80 are opened by default for the convenience of your business Web services. You can open or close the ports per your needs.
> - We recommend that you retain the ICMP rules for communication between nodes and the convenience of troubleshooting. Some tools also depend on ICMP.
> - The VPC security group sets the basic address of the container network segment as the Authorization Object. In this example, it is `172.20.0.0/16`. This is related to the initial ClasslessInter-Domain Routing (CIDR) block of Container Service that you set when creating the VPC cluster (for details, see Create a cluster). The Authorization Object ensures the communication between containers.

For the clusters created before February 28, 2017, the security group rules are loose. Take the classic network security group rules as an example.





To tighten the rules, refer to the configurations of the security groups created after February 28, 2017 and make the following changes by using **Add Security Group Rules** and **Delete** in the preceding figure:

- Add a rule in the intranet inbound and Internet inbound, with Allow selected as the Authorization Policy and All ICMP selected as the Protocol Type.
- To directly access ports 80, 443, or other ports of the virtual machine (VM), add the intranet and Internet rules to open these ports.

  > **Note** Make sure you open all the ports you need. Otherwise, some services will become inaccessible. Do not open ports accessed by using Server Load Balancer instances.

- Delete the Internet inbound rules and intranet inbound rules with `-1/-1` as the port range and `0.0.0.0` as the address range.

Security configuration principles

- Each cluster has one security group.

  Every Container Service cluster manages one security group. You can configure rules for this security group.

- Minimal permission principle.

  To ensure the security of your cluster, we recommend that the security group opens the minimal permissions to the external.

- Security groups created by Container Service add some default rules.

  For easier operations on ECS instances, security groups created by Container Service add some default rules, for example, ports 80 and 443 are opened. Delete the rules if you don't need them.

- Try to communicate by using the container intranet and do not expose communications to the host machine.
- When authorizing ECS instances outside the security group to access the security group, authorize a security group, instead of an individual IP address.

  To authorize ECS instances outside the security group to access the current security group, create a new security group, add these ECS instances to the new security group, and then authorize the new security group to access the current security group.

- Open the container network segment at the VPC intranet outbound/inbound.

  Otherwise, the network between containers is disconnected.

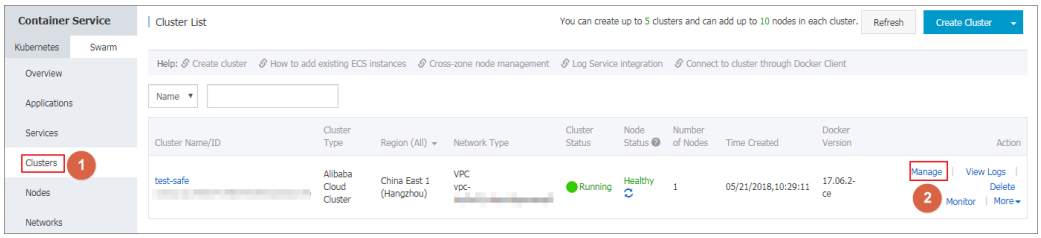
5.2. Check a security group

Context

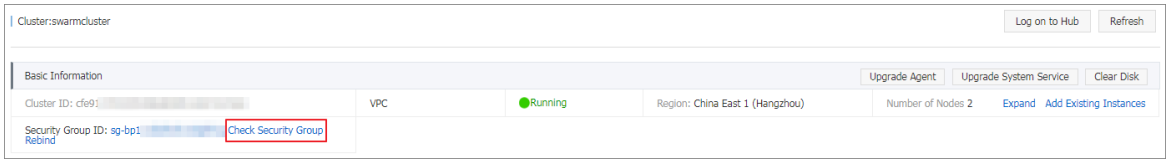
You can check whether the cluster security group rules are safe or not in the Container Service console.

Procedure

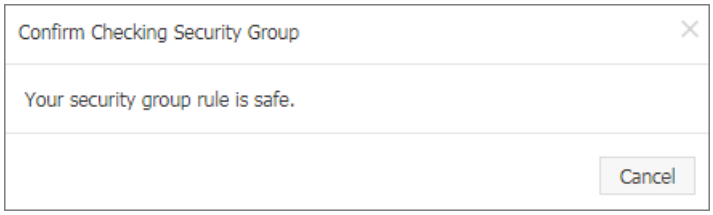
1. Log on to the Container Service console.
2. Click Swarm > **Clusters** in the left-side navigation pane.
3. Click **Manage** at the right of a cluster.



4. Click **Check Security Group**. Then, the system checks the status of the cluster security group.



If the security group rules are normal, the Confirm Checking Security Group dialog box opens. Click **Cancel**.



If your security group rules contain risks, a dialog box opens, listing the security group rules with risks. Click **Repair**. Then, the system deletes security group rules with risks and creates new security group rules.

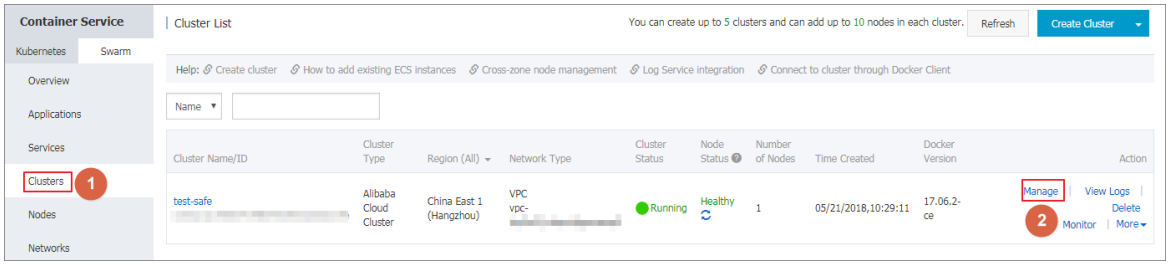
5.3. Rebind a security group

Context

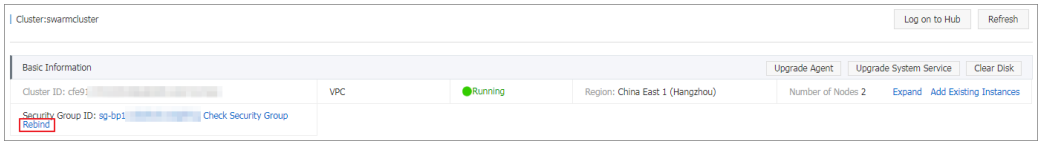
You can rebind a security group to your cluster in the Container Service console. The system re-creates a security group, binds this new security group to your cluster, and adds the nodes in your cluster to this new security group.

Procedure

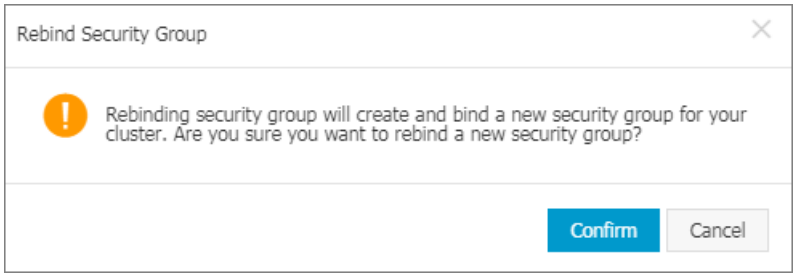
1. Log on to the Container Service console.
2. Click **Clusters** in the left-side navigation pane.
3. On the Cluster List page, click **Manage** at the right of a cluster.



4. Click **Rebind**.



5. The Rebind Security Group dialog box opens. Click **Confirm**.

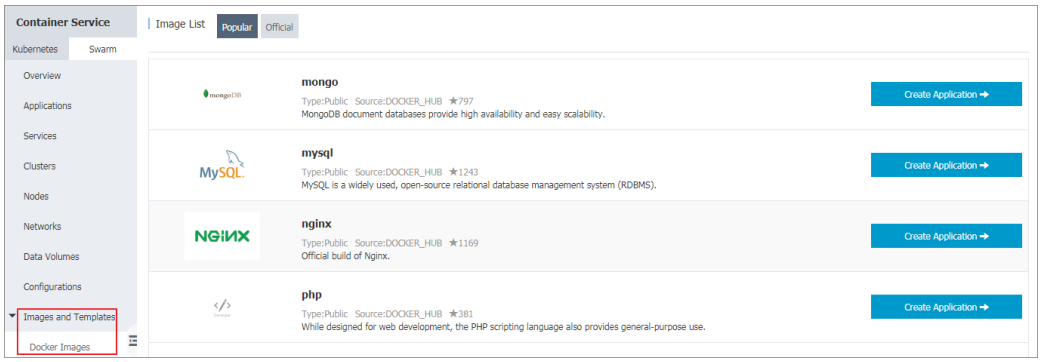


6.Images and templates

6.1. View image list

Procedure

1. Log on to the Container Service console.
2. In the left-side navigation pane, click **Images and Templates > Docker Images** .You can view the image category.
   - **Popular**: Some common images recommended by Container Service.
   - **Official**: Official images provided by Docker Hub.

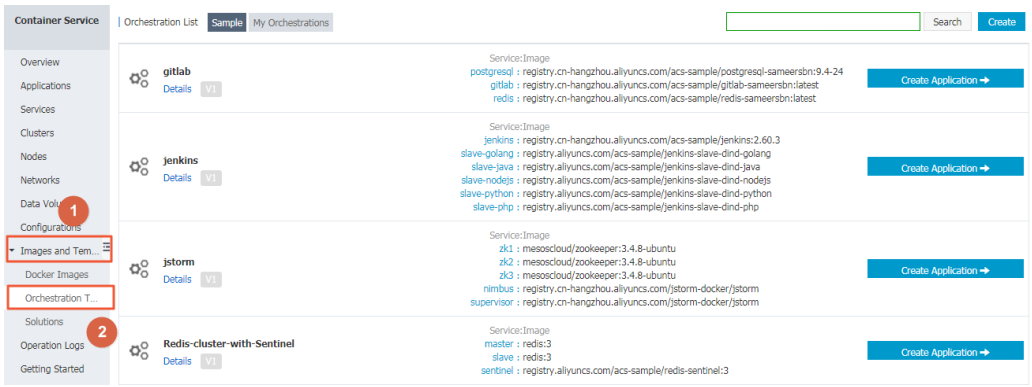


6.2. View orchestration template list

Procedure

1. Log on to the Container Service console.
2. In the left-side navigation pane, click > **Images and Templates > Orchestration Templates** .

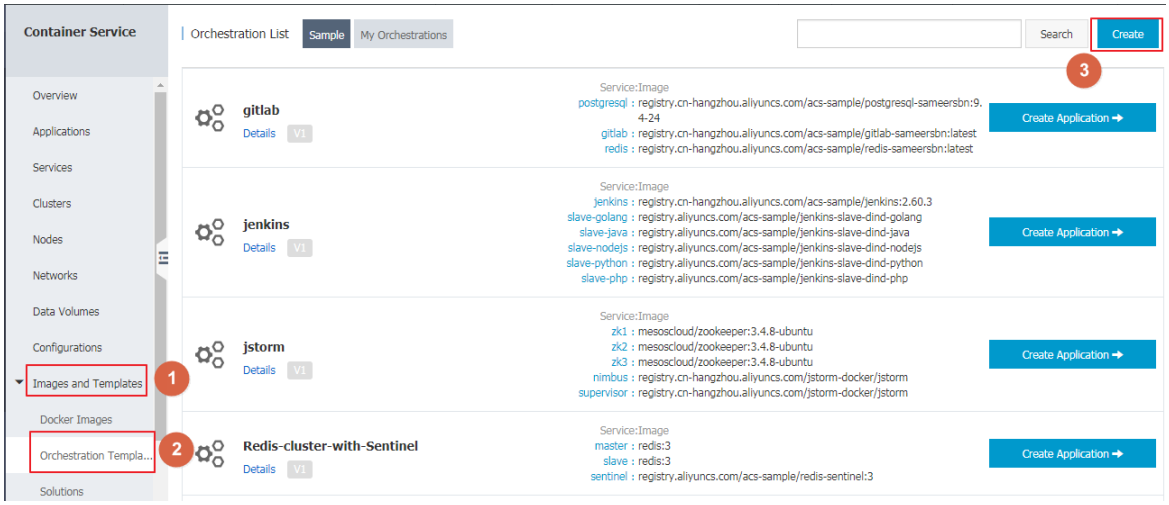
   You can view the template category, or search for specific templates according to the keywords of template name, description, and image information.
   - **Sample**: Common orchestration templates recommended by Container Service.
   - **My Orchestrations**: The orchestration templates you created.



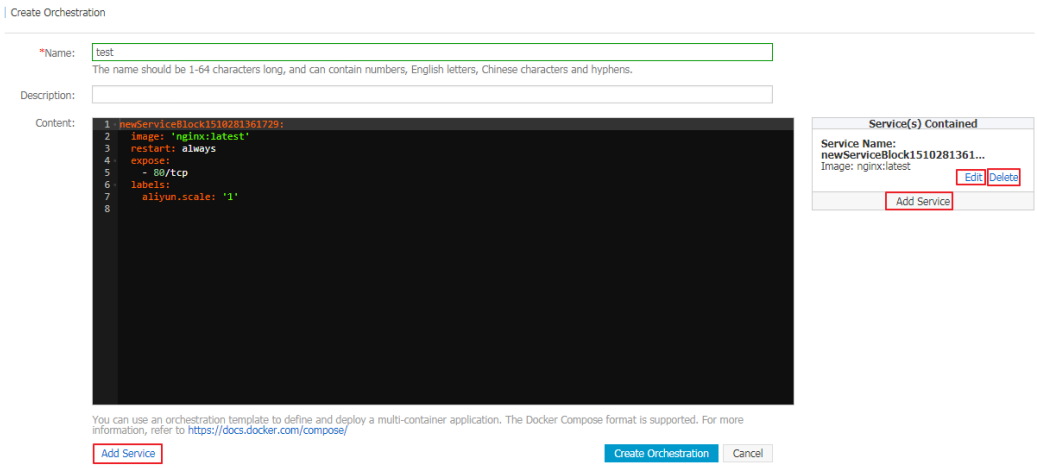
6.3. Create an orchestration template

Procedure

1. Log on to the Container Service console.
2. In the left-side navigation pane, click Orchestration TemplatesImages and Templates > > Orchestration Templates .
3. Click **Create** in the upper right corner.

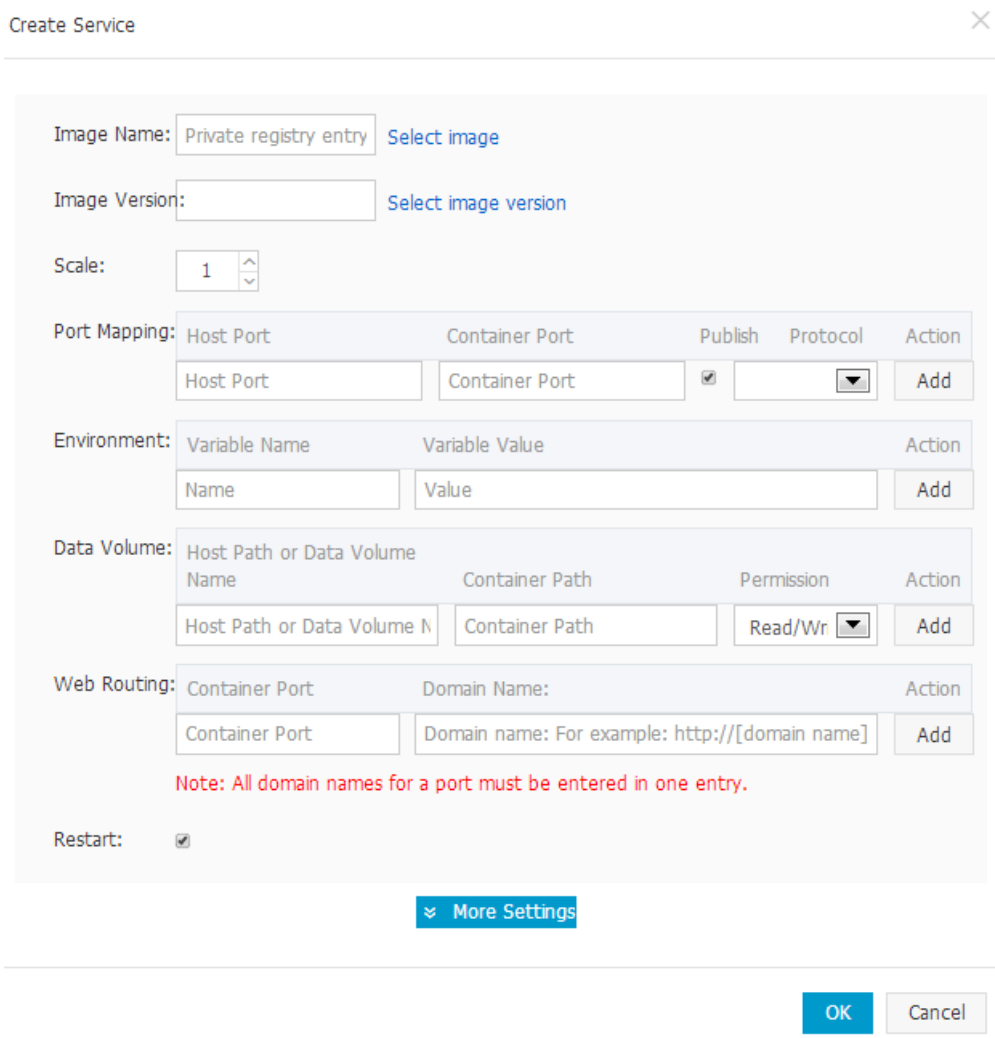


4. On the **Create Orchestration** page, complete the following configurations:
   - **Name**: Enter a name for the template.
   - **Description**: Enter the information about this template.
   - **Content**: The yml file of Docker Compose. For more information, see Compose file details.



The services contained in the orchestration template are displayed on the right side. To modify a service, click **Edit**. The Create Service dialog box opens. Modify the configurations and then click OK. To delete a service, click **Delete**.

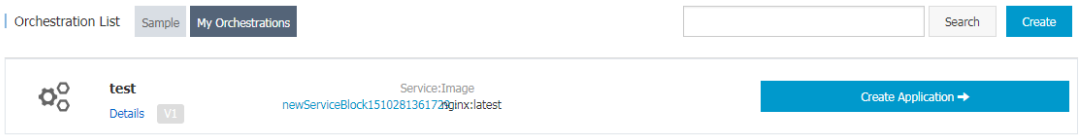
To add another service to the orchestration template, click Add Service. The **Create Service** dialog box opens. Select an image and complete the other configurations. Click OK.



5. Click **Create Orchestration** to create the orchestration template.

What's next

You can view your created orchestration templates under **My Orchestrations** on the **Orchestration List** page.



Click **Details** to view the detailed information of the orchestration template or click **Create Application** to create an application by using this orchestration template.

6.4. Update an orchestration template

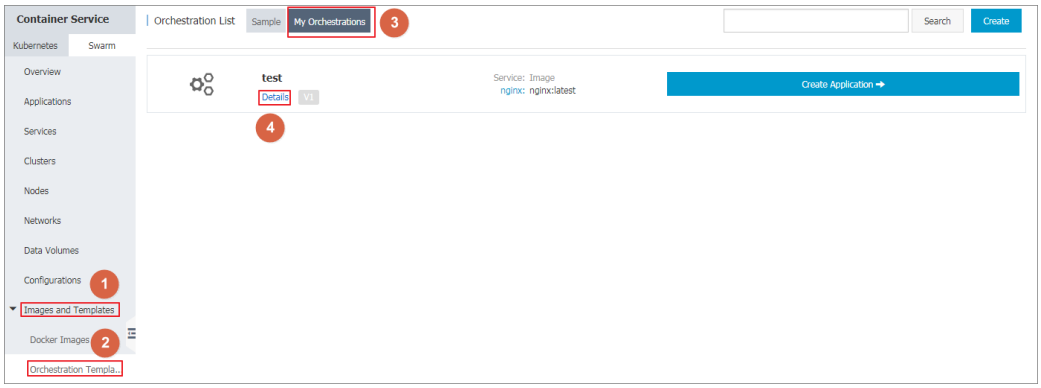
Context

You can only edit **orchestration templates** displayed under **My Orchestrations** on the Orchestration List page. To edit templates displayed under **Sample**, save the sample template as **your own template** and then edit it.

For how to save an orchestration template as a new one, see Save an orchestration template as a new one.

Procedure

1. Log on to the Container Service console.
2. n the left-side navigation pane, click **Images and Templates > > Orchestration Templates**.
3. Click the **My Orchestrations** tab and then click **Details** of the orchestration template you want to update.

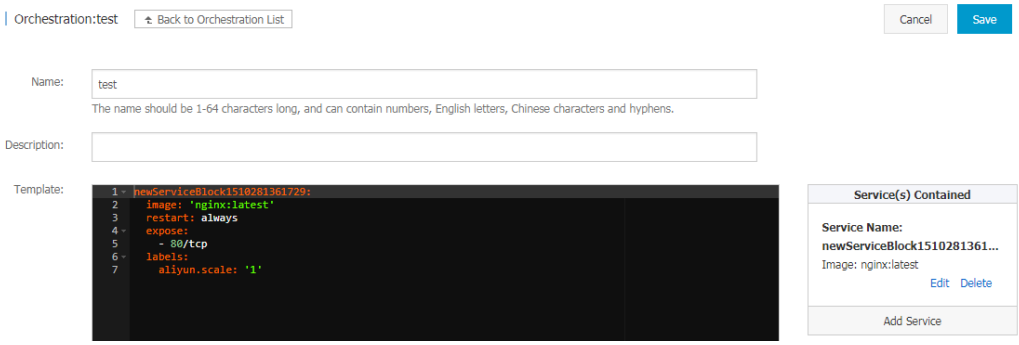


4. Click **Edit** in the upper-right corner.

5. Edit the template content.

   To modify a service, you can modify the content in the template directly or click **Edit** to modify the configurations in the appeared **Create Service** dialog box.

   To add another service to the orchestration template, click Add Service. The Create Service dialog box appears. Select an image and complete the other configurations. Click OK. You can modify the content in the template directly or click **Delete** to delete the service.

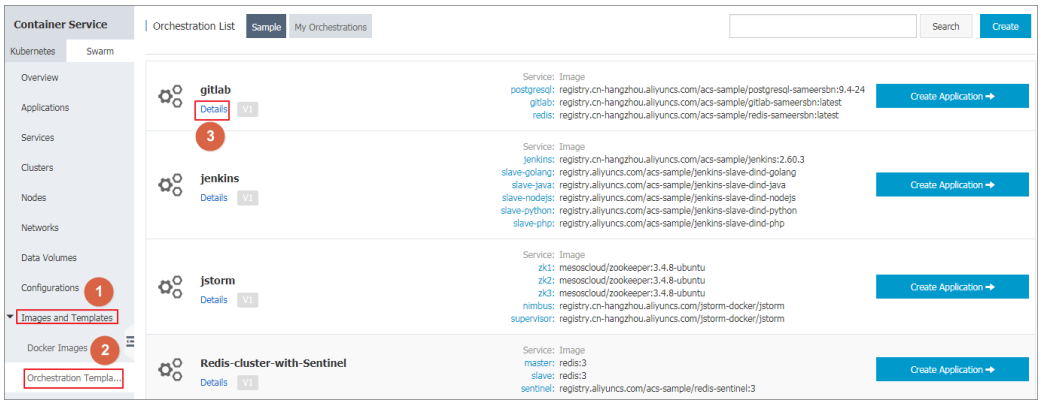


6. Click **Save** in the upper-right corner to save the modifications.

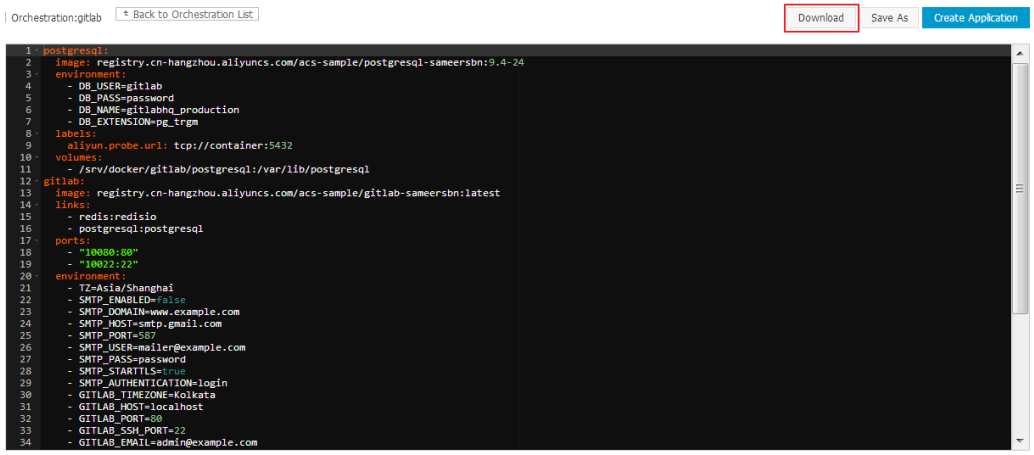
6.5. Download an orchestration template

Procedure

1. Log on to the Container Service console.
2. In the left-side navigation pane, click **Images and Templates** > > **Orchestration Templates** .
3. Click **Details** of the orchestration template you want to download.



4. Click **Download** in the upper right corner to download the template file with the suffix `yml` .



6.6. Delete an orchestration template

Context

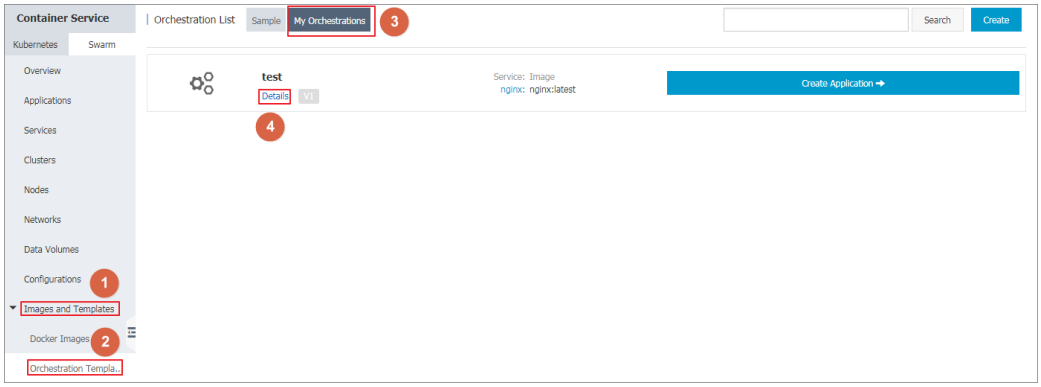
> **Note** You can only delete the orchestration templates in **My Orchestrations** on the **Orchestration List** page.

Procedure

1. Log on to the Container Service console.
2. In the left-side navigation pane, click Images and Templates > **Images and Templates** > >

Orchestration Templates .

3. Click the **My Orchestrations** tab and then click **Details** of the orchestration template you want to delete.



4. Click **Delete** in the upper right corner.

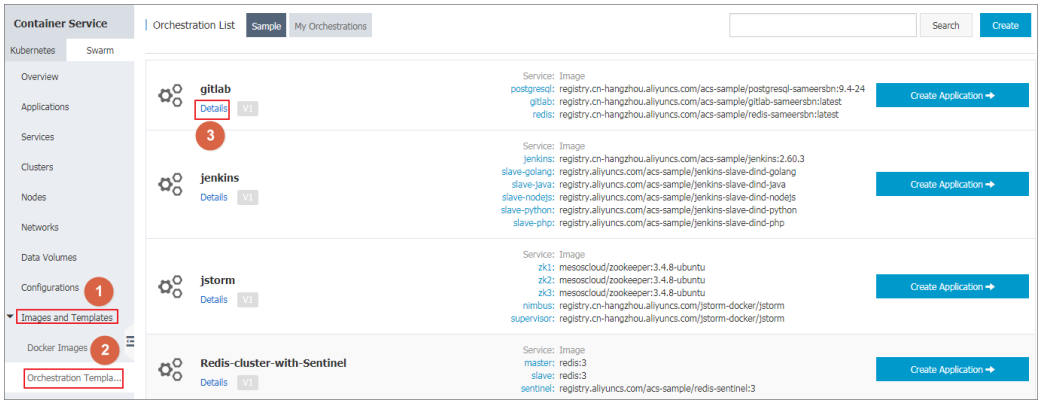
5. Click **OK** in the confirmation dialog box.

6.7. Save an orchestration template as a new one

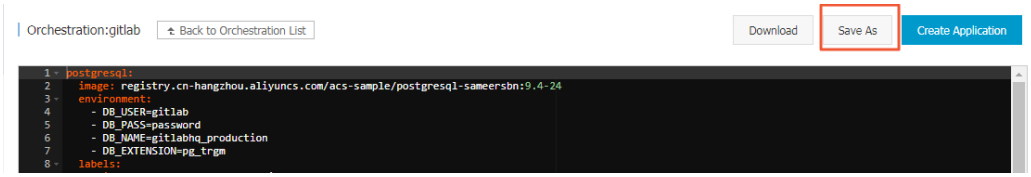
You can save an orchestration template under Sample or My Orchestrations on the Orchestration List page as a new one.

Procedure

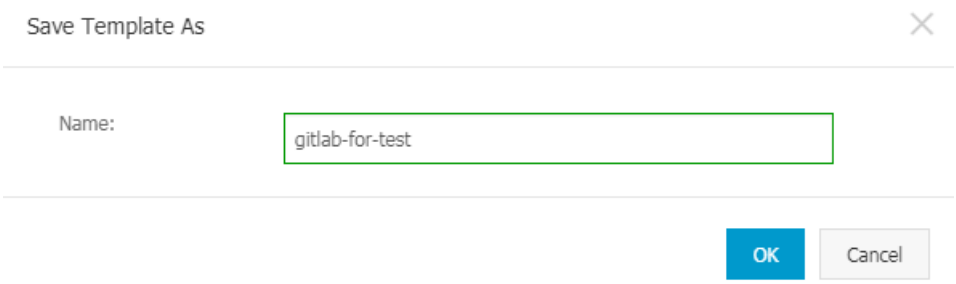
1. Log on to the Container Service console.
2. In the left-side navigation pane, click **Images and Templates > Orchestration Templates** .
3. Click the Sample tab or the My Orchestrations tab, and then click **Details** of the orchestration template you want to save as a new one.



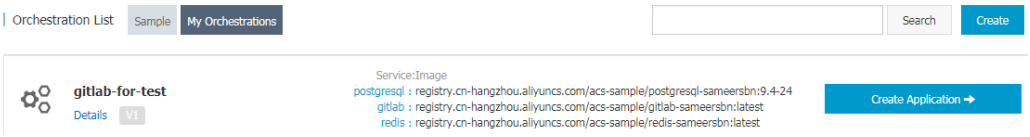
4. Click **Save As** in the upper right corner.



5. The Save Template As dialog box opens. Enter a name for the new template in the Name field and then click **OK**.



The new orchestration template is displayed under **My Orchestration** on the **Orchestration List** page.



7.Service orchestrations

7.1. Overview

Container Service supports describing multi-container applications by using the Docker Compose orchestration template.

The orchestration template allows you to describe an integrated application. The application can be composed of several services. For example, a portal application is composed of an Nginx service, a Web service, and a database service.

A service may have several containers. Make sure all of the containers have the same configurations. For example, the Web service in the preceding application can start two or more containers based on the traffic.

Capability

Container Service supports automatically deploying and managing an application by using the orchestration template.

The labels used by the orchestration template are compatible with most of the labels in Docker Compose V1 and V2. For information about specific compatible labels, see Label description.

The orchestration template also supports the template formats of Compose V1 and V2. For more information, see Docker Compose V1 and Docker Compose V2.

Container Service also provides many extension capabilities based on the community version:

- Unlike the community version of Docker Compose and Swarm, Alibaba Cloud Container Service supports cross-node container link. So you can directly deploy the application described by Docker Compose template to the distributed cluster to provide high availability and scalability.
- Container Service, based on the description in the Compose template of community version, also provides extensions to simplify the deployment, operation, and maintenance of Web and microservice applications. For more information, see Label description.

Example

The following is a WordPress application. It includes the Web service provided by WordPress image and the db service provided by MySQL image.

```
web:
  image: wordpress:4.2
  ports:
    - "80"
  environment:
    - WORDPRESS_AUTH_KEY=changeme
    - WORDPRESS_SECURE_AUTH_KEY=changeme
    - WORDPRESS_LOGGED_IN_KEY=changeme
    - WORDPRESS_NONCE_KEY=changeme
    - WORDPRESS_AUTH_SALT=changeme
    - WORDPRESS_SECURE_AUTH_SALT=changeme
    - WORDPRESS_LOGGED_IN_SALT=changeme
    - WORDPRESS_NONCE_SALT=changeme
  restart: always
  links:
    - db:mysql
  labels:
    aliyun.log_store_wordpress: stdout
    aliyun.probe.url: http://container/license.txt
    aliyun.probe.initial_delay_seconds: "10"
    aliyun.routing.port_80: wordpress;http://www.example.com;https://www.nice.com
    aliyun.scale: "3"
db:
  image: mysql:5.6
  environment:
    MYSQL_ROOT_PASSWORD: password
  restart: always
  labels:
    aliyun.log_store_mysql: stdout
```

7.2. Label description

The labels used by the Container Service orchestration template are compatible with most of the labels implemented in Docker Compose V1 and V2. Many extension capabilities are provided based on the community version.

Labels with extension capabilities

Container Service extends the deployment and lifecycle management capabilities for orchestration templates, and all the extension capabilities are described under `labels` and used as sub-labels.

Label	Description
probe	Sets the health check of a service.
Rolling_updates	Sets the rolling update of a service.
parallelism	Sets how many containers that `rolling_updates` can concurrently update at a time. **Note**: This label must be used with `rolling_updates`.

Label	Description
depends	Sets the dependencies of a service.
scale	Sets the number of containers for a service to scale horizontally.
routing	Sets the access domain name of a service.
routing.session_sticky	Sets whether or not routing keeps session sticky (namely, session persistence) during the routing request. **Note**: This label must be used with `routing`.
lb	Exposes the service port to the Internet or intranet by customizing Alibaba Cloud Server Load Balancer NAT mapping.
Log	Integrates with Alibaba Cloud Log Service to collect container logs and send the logs to Log Service.
Global	Sets the service as a global service.

Labels with function enhancement

Container Service provides the Service deployment constraints (affinity:service) label to set the deployment constraints for a service.

Additionally supported labels

Label	Description
External	Sets a service to directly link to an external address.
dns_options	Sets the DNS options. The semantics of this label is the same as that of --dns-opt parameter in the `docker run` command.
oom_kill_disable	Sets whether or not to prohibit OOM Killer. The semantics of this label is the same as that of --oom-kill-disable parameter in the `docker run` command.

Variable substitution

Container Service supports the parameterized Docker Compose template. The template can include the environment variables as parameters. When the template is deployed, you are prompted to enter the parameter values, and the template variables are substituted during the deployment.

For more information, see Variable substitution.

Container rescheduling

Container Service supports rescheduling Docker containers. When a node is invalid, the container can be automatically scheduled to another available node for operation.

For more information, see Container rescheduling.

High availability scheduling

To make the application have higher availability, Container Service supports scheduling containers of the same service in different zones. When a zone malfunctions, the application can still provide services.

For more information, see High availability scheduling.

Unsupported Docker Compose labels

Currently, Container Service does not support some Docker Compose labels. For more information, see Unsupported Docker Compose labels.

7.3. probe

Set the health check of a service.

- Check the health by using URLs. HTTP and TCP protocols are supported.
- Check the health by using shell scripts.

The health check is initiated from the container host. At regular intervals (two seconds by default), a request is sent to the container or the shell script commands are run on the container.

The health check is successful if the following criteria are met: The HTTP request returns the code 2XX/3XX. The TCP port can establish a link. The shell scripts return the value 0.

Descriptions of the fields used for check:

- `aliyun.probe.url`: The URL requested by HTTP and TCP. You only need to add the word `container`, without entering your domain name or IP address. The URL is used for the health check after being resolved into the corresponding IP address of the container. The service passes the health check when 2XX or 3XX is returned.
  - For example, the container provides the HTTP service by using the port 8080 and provides `/ping` as the URL for the health check. The URL format for the probe is `http://container:8080/ping`. Container Service automatically requests to check the URL returned results by using HTTP GET. The health check is successful if 2XX or 3XX is returned.
  - For example, MySQL container monitors port 3306. The URL format for the probe is `tcp://container:3306`. The service checks whether the container opens the port 3306 or not. If yes, the health check is successful.
- `aliyun.probe.cmd`: The shell command, `/check.sh`, is run during the health check. Container Service regularly runs this command within the container. If the shell scripts return the value 0, the health check is successful.
- `aliyun.probe.timeout_seconds`: The timeout for health check.
- `aliyun.probe.initial_delay_seconds`: The number of seconds delayed to start the health check after the start of the container.

> **Note**
>
> - A service can only contain either `aliyun.probe.url` or `aliyun.probe.cmd`.
> - If both `aliyun.probe.url` and `aliyun.probe.cmd` are not contained in the service, by default, the container is healthy and other `aliyun.probe.xxx` labels are ignored.

Example:

Use URL to check whether or not the container is healthy.

```
os:
  image: my_nginx
  labels:
    aliyun.probe.url: http://container/ping
    aliyun.probe.timeout_seconds: "10"
    aliyun.probe.initial_delay_seconds: "3"
```

Use shell scripts to check whether or not the container is healthy.

```
os:
  image: my_app
  labels:
    aliyun.probe.cmd: health_check.sh
    aliyun.probe.initial_delay_seconds: "3"
```

7.4. Rolling_updates

During a service update, if the service includes more than one container (defined by the scale label), the (N+1)th container is updated after the Nth container is successfully updated. In this way, the service downtime is minimized.

Example:

Deploy the WordPress service.Specify to deploy 2 containers by using the `scale` label. Use the `rolling_updates` label to minimize the service downtime for WordPress.

```
web:
  image: wordpress
  ports:
    - 80
  restart: always
  links:
    - 'db:mysql'
  labels:
    aliyun.logs: /var/log
    aliyun.routing.port_80: http://wordpress
    aliyun.rolling_updates: 'true'
    aliyun.scale: '2'
db:
  image: mariadb
  environment:
    MYSQL_ROOT_PASSWORD: example
  restart: always
  labels:
    aliyun.logs: /var/log/mysql
```

parallelism

The `parallelism` label defines how many containers that `rolling_updates` can concurrently update at a time.

> **Note** This label must be used with the `rolling_update` label.

parallelism value:

- The default value is 1, namely, updating one container at a time.
- When the value is greater than 1, during `rolling_updates`, a certain number (defined by the `parallelism` label) of containers are concurrently updated at a time to realize the batch update.
- When the value is invalid, the default value 1 is used.

> **Note** To ensure that at least one container is providing service, we recommend that the defined `parallelism` value is less than the number of containers in the service.

Example:

Deploy the Nginx service. Specify to deploy 3 containers by using the `scale` label. Use the `rolling_updates` label and `parallelism` label to define that 2 containers are updated in batch at a time.

```
web:
  image: nginx:latest
  restart: always
  environment:
      - "reschedule:on-node-failure"
  ports:
      - 80
  labels:
    aliyun.scale: "4"
    aliyun.rolling_updates: 'true'
    aliyun.rolling_updates.parallelism: "2"
```

7.5. depends

Set the dependencies of a service.

After setting the dependencies of a service, Container Service can control the start sequence of containers, starting the containers one by one.

Example:

> **Note** Separate dependencies by using a comma (,).

```
web:
  image: wordpress:4.2
  ports:
    - 80
  links:
    - db:mysql
  labels:
    aliyun.depends: db,redis
db:
  image: mysql
  environment:
    - MYSQL_ROOT_PASSWORD=password
redis:
  image: redis
```

7.6. scale

Set the number of containers for a service to scale horizontally.

Currently, Docker Compose can only start one container in each service. To expand the number of containers, manually set the number after the container is started.

You can use the `scale` label to scale as the container is started.

Moreover, after a container is deleted, you can redeploy the application by completing the following steps: Log on to the Container Service console. Click **Applications** in the left-side navigation pane. Click **Redeploy** at the right of the application you want to redeploy. Then, Container Service restarts the container or creates a new container to restore the number of containers to the specified quantity.

Example:

```
web:
  image: wordpress:4.2
  ports:
    - 80
  links:
    -db: mysql
  labels:
    aliyun. scale: "3"
db:
  image: mysql
  environment:
    - MYSQL_ROOT_PASSWORD=password
```

7.7. routing

The routing label configures the access domain name of a service.

Format:

```
aliyun.routing.port_$container_port: [http://]$domain|$domain_prefix[:$context_path]
```

Field description:

- `$container_port`: container port. **Note:** This is not the host port.
- `$domain`: domain name. Enter a domain name.
- `$domain_prefix`: domain name prefix. If you enter a domain name prefix, Container Service provides you with a test domain name and the domain name suffix is `.<cluster_id>.<region_id>.alicontainer.com`.
- `$context_path`: requested service path. You can select services according to the requested path.

Domain name selection:

- If the HTTP protocol is used to expose the service, you can use the internal domain name (the top-level domain is `alicontainer.com`) provided by Container Service for testing, or use your own domain name.
- If the HTTPS protocol is used, you can use only your own domain name. For example, `www.example.com`. You must modify the DNS settings to assign the domain name to the Server Load Balancer service provided by the container cluster.

Format requirements of the label statement:

- Container Service allocates a subdomain name to each cluster, and you only need to provide the domain name prefix to bind the internal domain name. The domain name prefix only indicates a domain name level and cannot be separated with periods (.).
- If you do not specify `scheme`, the HTTP protocol is used by default.
- The length of the domain name cannot exceed 128 characters. The length of the context root cannot exceed 128 characters.
- When you bind multiple domain names to the service, use semicolons (;) to separate them.
- A backend service can have multiple ports. These ports are exposed by the container. A port can only

be assigned one label. Therefore, a service with multiple ports must be assigned multiple labels.

Example:

Use the routing label.

Bind the internal domain name `wordpress.<cluster_id>.<region_id>.alicontainer.com` provided by Container Service and your own domain name `http://wp.sample.com/context` to port 80 of the Web service.

```
web:
  image: wordpress:4.2
  links:
    - db:mysql
  labels:
    aliyun.routing.port_80: wordpress;http://wp.sample.com/context
db:
  image: mysql
  environment:
    - MYSQL_ROOT_PASSWORD=password
```

The internal domain name that you finally get is `wordpress.cd3dfe269056e4543acbec5e19b01c074.cn-beijing.alicontainer.com`.

After starting the Web service, you can access the corresponding Web services by using the URL: `http://wordpress.cd3dfe269056e4543acbec5e19b01c074.cn-beijing.alicontainer.com` or `http://wp.sample.com/context`.

To support the HTTPS service, upload the HTTPS certificate by using the Server Load Balancer console on the Alibaba Cloud website, and then bind the corresponding cluster to access the Server Load Balancer terminal.

routing.session_sticky

By using this feature, you can determine whether to maintain session sticky (session persistence) when you set the routing for a routing request. With session persistence, during the session, each request is routed to the same backend container instead of being randomly routed to different containers.

> **Note**
>
> - The setting takes effect only when you have configured `aliyun.routing.port_$containr_port`.
> - Simple routing session persistence is based on the Cookie mechanism. By default, the maximum expiration time of Cookie is 8 hours and the idle expiration time is 30 minutes.
> - Simple routing session persistence is enabled by default.

The setting methods are as follows:

- Enable session persistence

  `aliyun.routing.session_sticky: true`

- Disable session persistence

  `aliyun.routing.session_sticky: false`

Example of a template orchestration file:

```
web:
  image: wordpress:4.2
  links:
    - db:mysql
  labels:
    aliyun.routing.port_80: wordpress;http://wp.sample.com/context
    aliyun.routing.session_sticky: true
db:
  image: mysql
  environment:
    - MYSQL_ROOT_PASSWORD=password
```

7.8. lb

Expose the service port to the Internet or intranet by customizing Alibaba Cloud Server Load Balancer NAT mapping. The Agent must be upgraded to the latest version to support this extension capability label.

The label format is as follows. Variables with `$` are placeholders.

```
aliyun.lb.port_$container_port:$scheme://$[slb_name|slb_id]:$slb_front_port
```

Example:

```
web:
  image: wordpress:4.2
  ports:
    - 7777:80
    - 9999:9999
    - 8080:8080
    - 53:53/udp
  links:
    - db:mysql
  labels:
    aliyun.lb.port_80: http://slb_example_name:8080
    aliyun.lb.port_9999: tcp://slb_example_name:9999
    Aliyun. lb. port_8very: https: // FIG: 80
    Aliyun. lb. port_53: UDP: // FIG: 53
db:
  image: mysql
  environment:
    - MYSQL_ROOT_PASSWORD=password
```

To better use the custom Server Load Balancer `lb` label, you must understand three ports used in a routing request: the Server Load Balancer frontend port, the Server Load Balancer backend port (namely, the Elastic Compute Service (ECS) instance port), and the container port. Take the first `lb` label `aliyun.lb.port_80` as an example. From left to right, The Server Load Balancer backend port is the ECS instance port, which can be obtained from host and container port mapping of the ports label. The container port 80 corresponds to the host port port 80 in the key indicates the port to be exposed by the container, and port 8080 indicates the frontend port to be exposed by Server Load Balancer. The Server Load Balancer backend port is the ECS instance port, which can be obtained from host and container port mapping of the `ports` label. The container port 80 corresponds to the host port 7777. So the backend port forwarded by Server Load Balancer is 7777. The first lb label indicates that a request sent to the Web service: First enters port 8080 of the Server Load Balancer frontend. Then, is forwarded to port 7777 of the backend ECS instance. Enters port 80 of the container according to the port mapping of `ports`. Finally, is submitted to the WordPress process in the container to provide the service. The other three lb labels also apply to the preceding explanation. All the Server Load Balancer instances configured by the lb label do not go through the routing service built in the cluster, and you control the request routing by yourself.

Format requirements of the label statement:

- The Server Load Balancer instance can be specified by using its name or ID.
- The Server Load Balancer instance name is limited to 1-80 characters, including letters, numbers, hyphens (-), forward slashes (/), periods (.), and underscores (_).
- The container port is limited to 1-65535.
- The Server Load Balancer frontend port is limited to 1-65535.

Limits on deploying services with custom Server Load Balancer NAT mapping:

- You must create a Server Load Balancer instance, name it, and create the corresponding listening port. Then, provide the mapping container port `$container_port`, the used protocol $scheme (possible values include `tcp`、`http`、`https`、`udp`, and the Server Load Balancer instance name `$slb_name` or `$slb_id` by using extension labels, and specify the frontend port `$slb_front_port` of the Server Load Balancer instance.
- You must specify the host and container port mapping of the service port to be exposed and then use the standard Dockerfile label `ports` to specify the port mapping. to specify the port mapping. You must specify the host port and this port cannot conflict with the host port mapped by other services. Server Load Balancer uses the host port to bind the backend ECS instance.
- A service can only use one or more Server Load Balancer instances to expose the service port. Services cannot share and use the same Server Load Balancer instance because they are distributed in different ECS instance backends.
- When using the lb label to configure Server Load Balancer routing, the default Server Load Balancer instance cannot be used.
- The host that has the service with Server Load Balancer NAT mapping deployed uses the same host and container port mapping. Therefore, these services only have one instance on each ECS.
- The supported Server Load Balancer protocol `$scheme` includes tcp, http, https, and udp.
- You must create a listening port in the Alibaba Cloud Server Load Balancer console.
- Log on to the Server Load Balancer console to modify the configurations for the Server Load Balancer instance used in Container Service, such as bandwidth limitation.

- The value of the lb label is that you do not need to bind the backend ECS instance of Server Load Balancer by yourself. After configuring the corresponding labels, the backend is bound automatically. instance of Server Load Balancer by yourself. After configuring the corresponding labels, the backend is bound automatically. Therefore, except for binding the Server Load Balancer backend, you must set and modify the Server Load Balancer instances in the Alibaba Cloud Server Load Balancer console.
- Container Service helps you generate a Resource Access Management (RAM) sub-account (you are required to activate RAM). This account has some Server Load Balancer permissions, but does not have the permission to create or delete Server Load Balancer instances. Use this account to help you manage the Server Load Balancer instances used in Container Service, for example, binding some nodes in the cluster as the service backend.
- In the whole lifecycle of the service, the `lb` label always works unless the service is deleted or the service is redeployed after `lb` label is deleted. Meanwhile, the Server Load Balancer instances configured in the `lb` label cannot be mixed.

7.9. Log

Container Service, integrated with Alibaba Cloud Log Service, collects container logs and sends the logs to Alibaba Cloud Log Service.

Example:

```
mysql:
    image: mysql
    ports:
        - 80
    labels:
        aliyun.scale: "1"
    environment:
        - MYSQL_ROOT_PASSWORD=password
wordpress:
    image: registry.aliyuncs.com/jiangjizhong/wordpress
    ports:
        - 80
    labels:
        aliyun.routing.port_80: wordpress-with-log
        aliyun.log_store_dbstdout: stdout #Note here
    links:
        - mysql
```

For more information, see Enable Log Service.

7.10. Global

Set the service as a global service.

Some services need to be deployed to every node, such as monitoring and logging services. When a node is created, these services are deployed to the node.

A service is deployed to each node of the cluster if the service is set as `global`. When a node is created or added in the cluster, a container instance is automatically deployed to the node.

```
monitor:
  image: sample
    labels:
      aliyun.global: true
```

7.11. Service deployment constraints (affinity:service)

Set the deployment constraints for a service.

Container Service supports the container deployment constraints compatible with Docker Swarm. You can control the deployment of a container with the Docker Swarm filters.

But the community version of Docker Compose does not have relevant capabilities to control the direct deployment constraint for a service.

In Container Service, you can add `affinity:service` in `environment` to constrain the `affinity` between services so as to control the service deployment policy. Container Service supports `soft affinity` and `hard affinity` between services.

Example:

In this example, `affinity:service!=db` is the deployment constraint for the `web` service. In this way, the `web` service is always deployed to nodes where the `db` service is not deployed. Therefore, when a node is invalid, the service availability is enhanced. When your cluster has only one node, as hard anti-affinity is specified, the deployment will fail because it cannot meet the specified mandatory constraints.

```
web:
  image: registry.aliyuncs.com/acs-sample/wordpress:4.5
  ports:
    - '80'
  environment:
    - affinity:service! =db
  restart: always
  links:
    - 'db:mysql'
  labels:
    aliyun.logs: /var/log
    aliyun.probe.url: http://container/license.txt
    aliyun.probe.initial_delay_seconds: '10'
    aliyun.routing.port_80: http://wordpress
    aliyun.scale: '2'
db:
  image: registry.aliyuncs.com/acs-sample/mysql:5.7
  environment:
    MYSQL_ROOT_PASSWORD: password
  restart: always
  labels:
    aliyun.logs: /var/log/mysql
```

7.12. External

Set a service to directly link to an external address.

In the extension field, the following fields can be used:

- `host`: Set the domain name of the link.
- `ports`: Set the port of the link.

Example:

Directly start a MySQL container without using the `external` label.

```
web:
  image: wordpress:4.2
  ports:
    - 80
  links:
    - db:mysql
db:
  image: 10.32.161.160:5000/mysql
  environment:
    - MYSQL_ROOT_PASSWORD=password
```

Use the `external` label to describe an RDS service that is not deployed in the cluster and provide the service to the WordPress deployed in the cluster for use.

```
wordpress:
  image: wordpress:4.2
  ports:
    - 80
  links:
    - db:mysql
  environment:
    - WORDPRESS_DB_USER=cloud
    - WORDPRESS_DB_PASSWORD=MYPASSWORD
    - WORDPRESS_DB_NAME=wordpress
db:
  external:
    host: rdsxxxx.mysql.rds.aliyuncs.com
    ports:
      - 3306
```

7.13. dns_options

Set the DNS options. The function of this label is the same as that of --dns-opt in the `docker run` command.

```
wordpress:
  image: wordpress:4.2
  dns_options:
    - "use-vc"
```

7.14. oom_kill_disable

Set whether or not to prohibit OOM Killer. The function of this label is the same as that of --oom-kill-disable in the `docker run` command.

```
wordpress:
  image: wordpress:4.2
  oom-kill-disable: true
```

7.15. Variable substitution

Container Service supports the parameterized Docker Compose template. The template can include the environment variables as parameters. When the template is deployed, you are prompted to enter the parameter values, and the template variables are substituted during the deployment.

For example, you can define the parameter POSTGRES_VERSION.

```
db:
  image: "postgres:${POSTGRES_VERSION}"
```

When the preceding Compose template is deployed, Container Service prompts you to enter the value of the POSTGRES_VERSION parameter, such as 9.3. Container Service substitutes the variable of the Compose template with this parameter value. In this example, a `postgres:9.3` container is deployed.

Container Service is fully compatible with Docker Compose syntax, and you can use either `$VARIABLE` or `${VARIABLE}` syntax in the template.

In the Compose template, you can use `$$` to escape strings containing `$`. In this way, Container Service will not erroneously treat such a string as the parameter.

For more information on the variable substitution supported in the Compose template, see Variable substitution.

7.16. Container rescheduling

Container Service supports rescheduling Docker containers. When a node is invalid, the container can be automatically scheduled to other available node for operation.

By default, container rescheduling is disabled. If needed, you can enable the container rescheduling.

Container Service provides a container rescheduling policy that is compatible with Docker Swarm. You can enable container rescheduling by using environment variable or label.

Environment variable:

```
redis:
  image: redis
  environment:
    - reschedule:on-node-failure
```

Label:

```
web:
  image: nginx
  restart: always
  environment:
    - aaaaa=aaaaa
  labels:
    aliyun.scale: "3"
    com.docker.swarm.reschedule-policies: "[\"on-node-failure\"]"
```

Note If you re-schedule the container, You need to restore the Persistence State required for the docker container, you need to work with docker file volumes that support data migration or sharing.

7.17. High availability scheduling

To make the application have higher availability, Container Service supports scheduling containers of the same service in different zones. When a zone malfunctions, the application can still provide services.

You can specify the zone selection in the orchestration file by using the environment variables in the following formats:

- availability:az==3

  The service must be distributed in at least three zones. The container creation fails if less than three zones are in the current cluster, or the service cannot be distributed in three zones because of limited machine resources.

- availability:az==~3

  Try to distribute the service in three zones. Container can still be created even if the condition cannot be met.

In the following example, the service must be distributed in at least two zones.

```
nnn:
  expose:
    - 443/tcp
    - 80/tcp
  image: 'nginx:latest'
  environment:
    - 'availability:az==2'
  labels:
    aliyun.scale: '8'
  restart: always
  volumes:
    - /var/cache/nginx
```

7.18. Unsupported Docker Compose labels

Label	Description
build	This label is used to build container images by using the Dockerfile and other files in the current directory. Currently Container Service does not provide the function to build images. We recommend that you separate the building and deployment operations. You can use image repository to build the image from the code source, or push the image built locally to the image repository. You can use the `image` label in the orchestration template to refer to the image in the image repository (including private repository).
dockerfile	The same as `build` .
env_file	Currently Container Service does not support specifying environment variables in files. You can add environment variables by using the `environment` label.
mac_address	Currently the setting of Mac address is not supported.
detach	All the images in Container Service are enabled in the detach mode, and the attach mode is not allowed.
stdin_open	The same as `detach` .
tty	The same as `detach` .
extends	Not supported.
networks	The network in Compose file format of version 2 version 2. See Cross-host interconnected container network for network management and service discovery for Container Service. allows the service container to start in the custom network, and the containers in Container Service are all in the same cross-host interconnected container network. Therefore, Container Service does not support using the networks label in Compose file format of version 2. See Container network interconnection for network management and service discovery for Container Service.

8.Applications

8.1. Create an application

Context

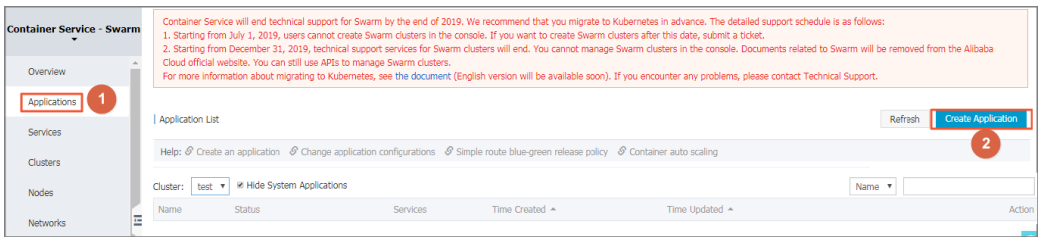
Limits

Swarm clusters only support the compose V1 and compose V2 orchestration templates. The system reports an error if you select to use the compose V3 template.

> **Note** In the orchestration template list, compose V3 templates are marked with `composev3`.

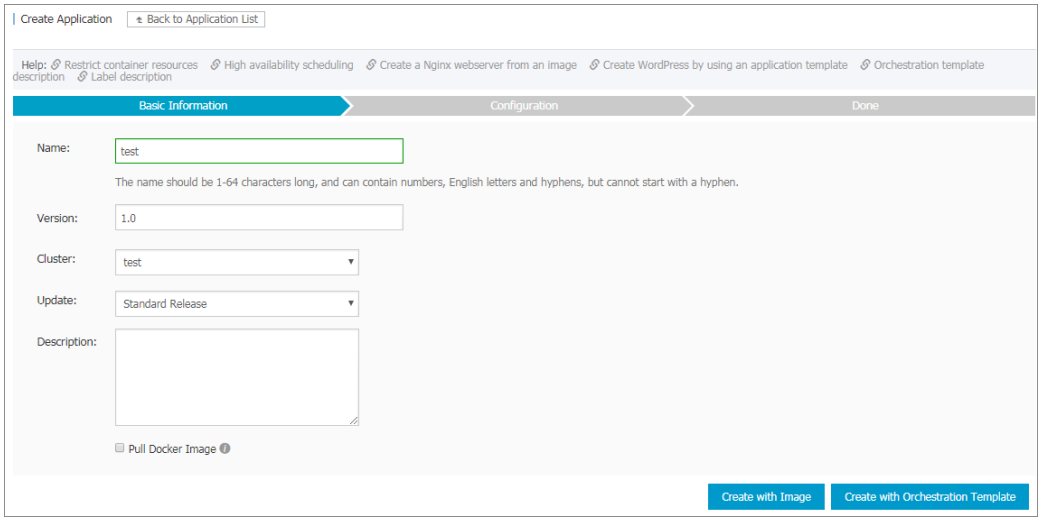
Procedure

1. Log on to the Container Service console.
2. Click **Applications** in the left-side navigation pane.
3. Click **Create Application** in the upper-right corner.



4. Complete the basic information for the application you are about to create.
   - **Name**: Enter the name of the application. It can be 1-64 characters long and contain numbers, English letters, and hyphens (-), but cannot start with a hyphen (-).
   - **Version**: Enter the version of the application. By default, 1.0 is entered.
   - **Cluster**: Select the cluster on which the application is to be deployed.
   - **Update**: The update method of the application. Select **Standard Release** or **Blue-Green Release**. For more information, see Introductions on release strategies.
   - **Description**: Enter the information of the application. This field is optional. The entered description cannot exceed 1024 characters, and is displayed on the **Application List** page.
   - **Pull Docker Image**: With this check box selected, Container Service pulls the latest Docker image from the repository to deploy the application, even when the image tag does not change.

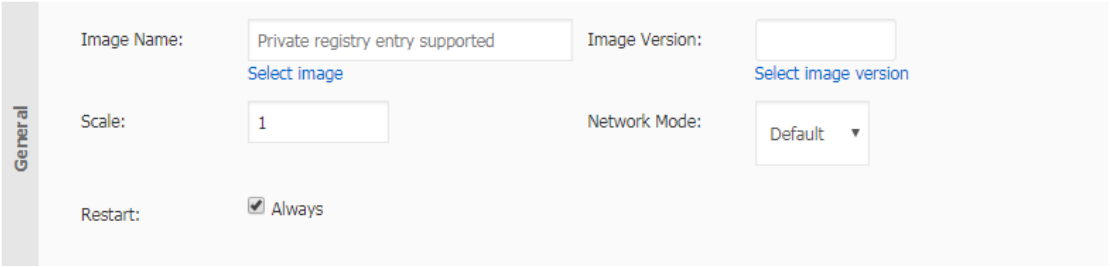
     To improve efficiency, Container Service caches the image. When deploying an application, Container Service uses the cached image instead of pulling the image from the repository if the image tag is the same as that of the local cache. Therefore, if you modify your codes and image but do not modify the image tag for the convenience of upper business, Container Service uses the old image cached locally to deploy the application. With this check box selected, Container Service ignores the cached image and re-pulls the image from the repository when deploying the application to make sure the latest image and codes are always used.



5. Click **Create with Image**.

   Click **Create with Image**. Set the following parameters according to your requirements.

   i. In the General section:



   - Set the **Image Name** and **Image Version**.

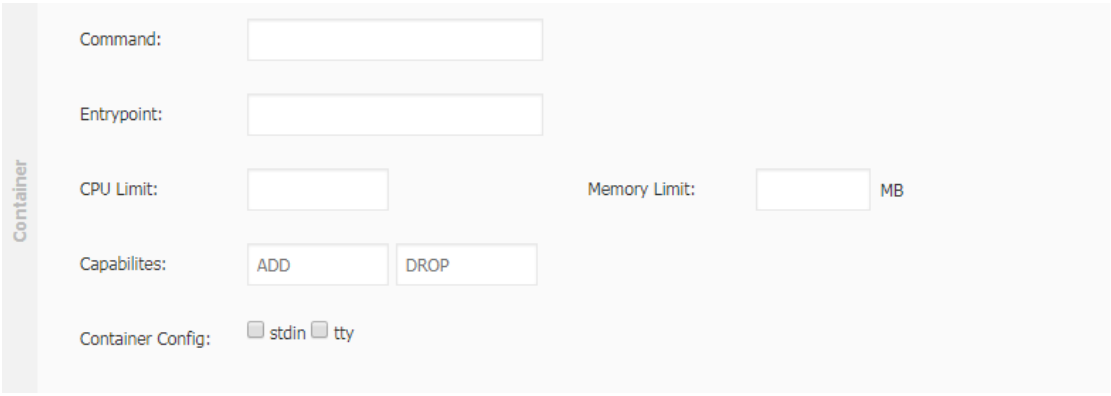
     You can select an image provided by Container Service or enter your image address in the format of `domainname/namespace/imagename:tag`. To select an image, click **Select image**, select the image, and then click **OK**. By default, the Container Service uses the latest image version. To use another version of the image, click **Select image version**, and then click **OK**.

   - Set the number of containers (**Scale**).
   - Select the **Network Mode** of the application. Currently, Container Service supports two network modes: **Default** and **host**. The Default mode is the bridge network mode. The host network mode allows containers to use the network stacks of Elastic Compute Service (ECS) instances. For more information, see Docker container networking.
   - Set the **Restart** field.

     The Always check box is selected by default. With the check box selected, the containers are restarted regardless of the exit status code. Docker daemon restarts the containers unlimitedly. Whatever the container status is, the container tries to be restarted when daemon is started.

     When the check box is not selected, the restart policy becomes no, indicating containers are not restarted automatically on exit.

ii. In the Container section:



- Set the startup command (**Command** and **Entrypoint**) of the container. If configured, the image default configurations are overwritten.

  Command is used to specify the startup command of the container main process. For more information, see Command.

  Entrypoint is used to specify the container startup process and parameter. Used together with command, the cmd contents can be passed to Entrypoint as parameters. For more information, see Entrypoint.

- Set the resource limits (**CPU Limit** and **Memory Limit**) of the container.

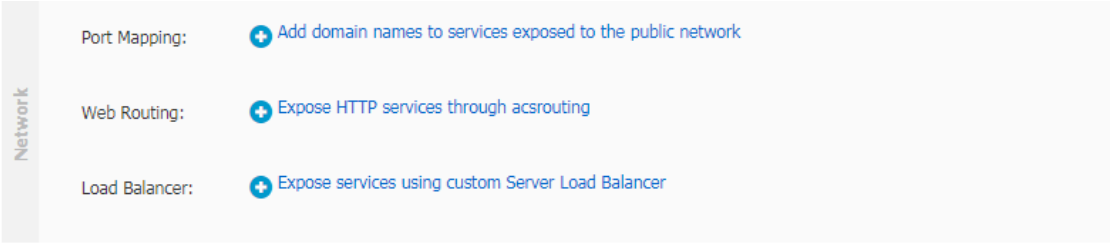
  Set the resource limit for the CPU and memory to be used by the container. For more information, see Restrict container resources.

- Set the **Capabilities**.

  For how to add or drop Linux related privileges for the container, see Capabilities.

- Set the **Container Config**.

iii. In the Network section:



- Set the **Port Mapping**. Specify the port mapping for the host and the container, and select TCP or UDP as the protocol.

  The port mapping is used for the routing between container and host, and is the precondition of Web Routing and Load Balancer. The container provides external services by means of the configured port mapping.

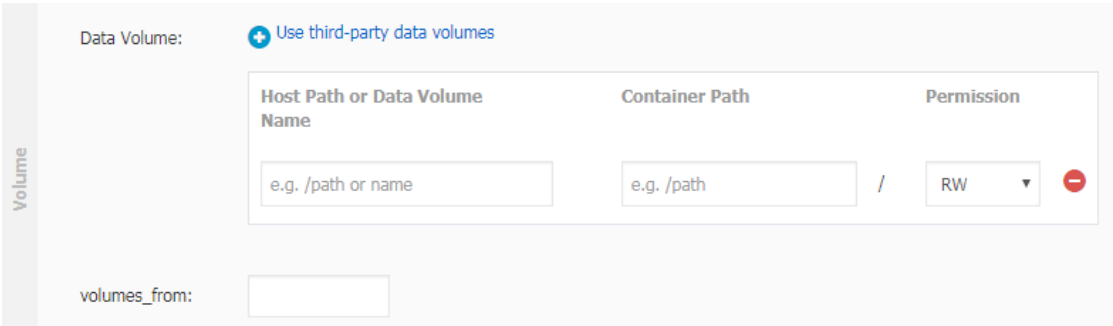
- Set the **Web Routing**. The cluster automatically creates the acsrouting application, including the routing service, and provides the simple routing function. A routing service instance is deployed on each node. In a node, the `acsrouting_routing_index` container implements the routing forward in the cluster to route the HTTP or HTTPS service. For more information, see Simple routing - supports HTTP and HTTPS.

  > **Note** When exposing the HTTP/HTTPS services, you can use the overlay network or Virtual Private Cloud (VPC) to directly access the container port, without configuring the specific host port.

- Set the **Load Balancer**. Configure the port mapping before configuring the mapping of `container_port` `$scheme://$[slb_name|slb_id]:$slb_front_port`. For how to use the Server Load Balancer label, see lb.

  When configuring this parameter, control the routing access path on your own, including the routing mapping of Server Load Balancer front end port > backend host port > container port.

iv. Set the **Data Volume**.



- Create a data volume. Enter the host path or data volume name, the container path, and select RW or RO as the data volume permission. For more information, see volume.
- Configure the volumes_from field. Enter the name and permission parameter of another service or container, such as `service_name:ro`. If no access permission is specified, RW is the default permission. For more information, see volumes_from. After the configuration, the container is authorized to use volumes of another service or container.

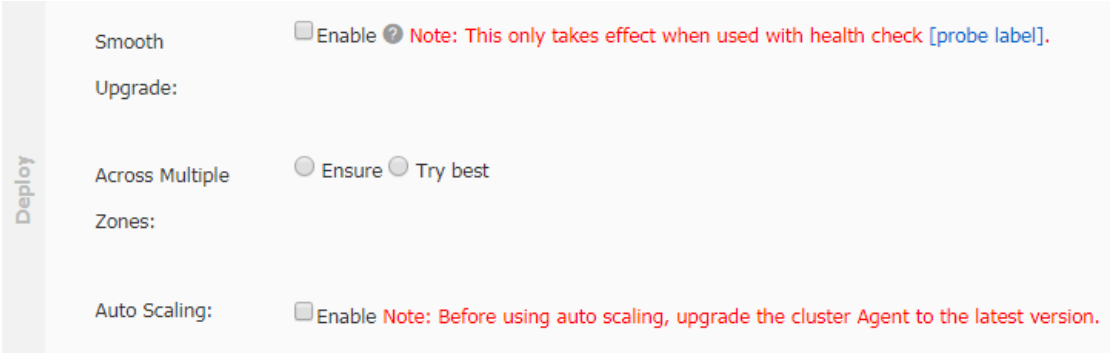
v. Set the **Environment variables**.

Formats such as array, dictionary, and boolean are supported. For more information, see Environment variables.

vi. Set the container **Labels**.

For the extension labels supported by Container Service, see Label description.

vii. In the Deploy section:



- Set whether to enable **Smooth Upgrade** for containers.

  For more information, see Rolling_updates.

- Set the **Across Multiple Zones** settings for containers.

  Select **Ensure** to deploy containers in two zones. The container creation fails if less than two zones are in the current cluster, or the containers cannot be deployed in two zones because of limited machine resources. Select **Try best** to try to deploy containers in two zones. The container can still be created even if this condition is not met.

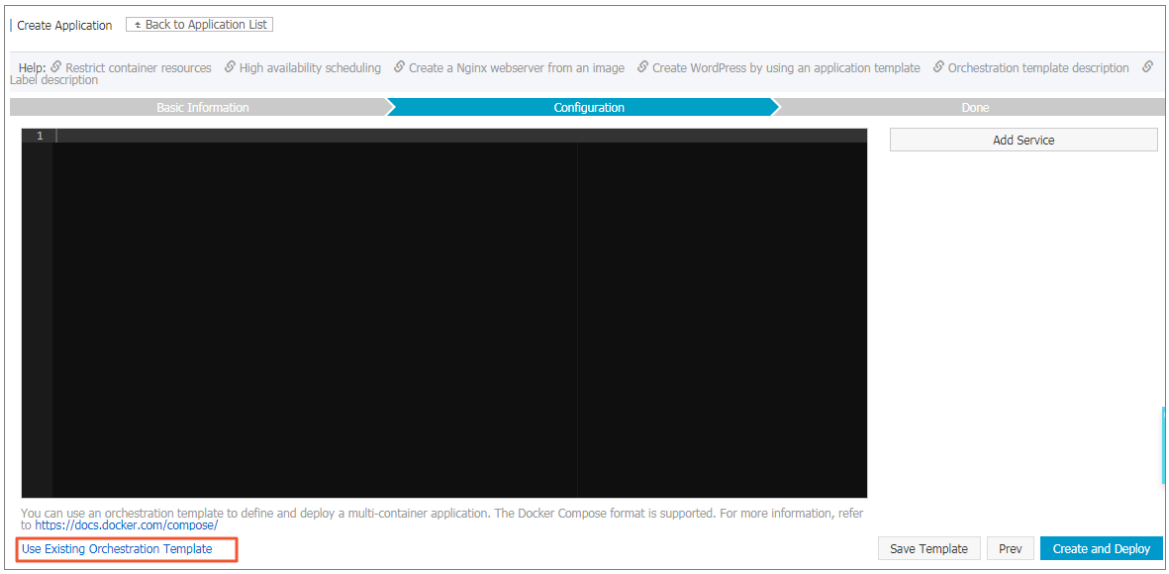
  If this parameter is not configured, Container Service deploys the containers in one zone by default. For more information, see High availability scheduling.

- Set whether or enable the container **Auto Scaling**.

  For more information, see Container auto scaling.

viii. Click **Create** at the right of the page after completing the settings.

6. Click **Create with Orchestration Template**.



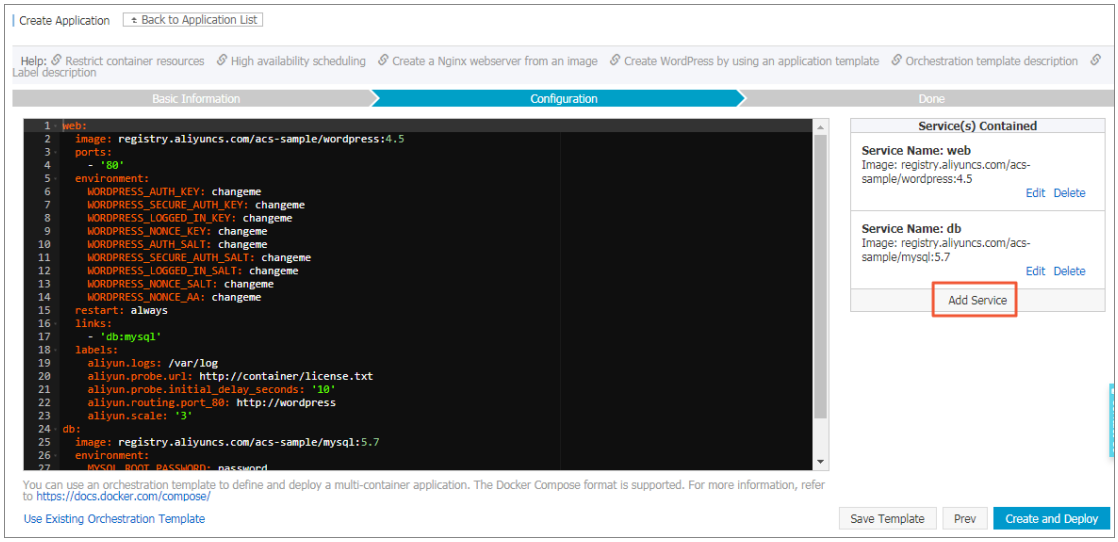
i. Click **Use Existing Orchestration Template** or write a new template by yourself.

The contents of the orchestration template comply with the Docker Compose format.

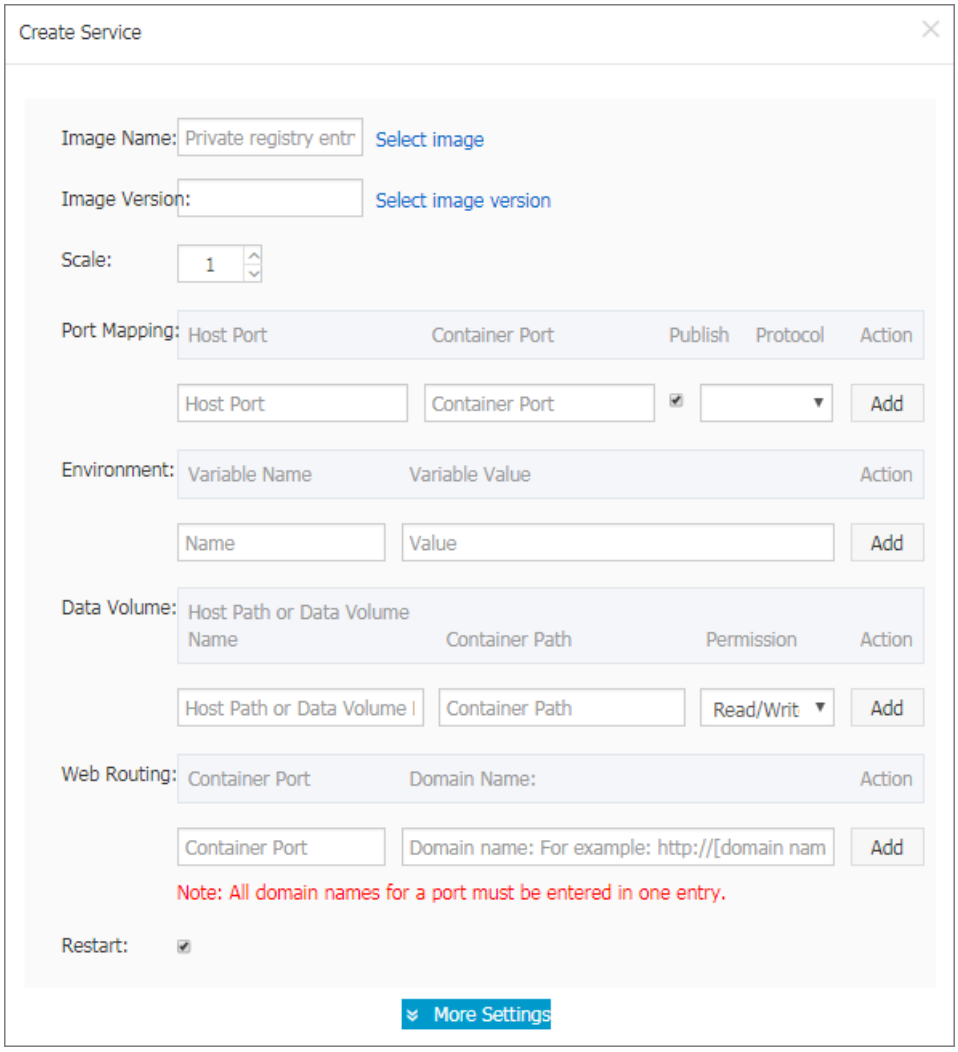
ii. Click **Select** next to the template after clicking Use Existing Orchestration Template.

iii. Edit the orchestration template.

Edit the orchestration template according to your requirements. Make modifications in the template directly, or click **Edit** to modify the service or **Delete** to delete the service.



Click **Add Service** to add another service to this orchestration template. Select the image and configure the parameters. Then, click **OK**.



iv. Click **Create and Deploy** after completing the settings.

8.2. Application parameter configurations

This document aims to help you understand what the parameters on the page mean when you create a swarm application by using an image. Then, you can configure the parameters smoothly. For some parameters, some documents are provided for your reference.

Image Name

- Select an existing image in the image list.
- Enter the image address directly. Take the image address of a Docker Hub user's WordPress container as an example. `registry.cn-hangzhou.aliyuncs.com/acs-sample/wordpress:4.6` is a complete image address composed of domain name, namespace, image name, and label. For more information about the image address, see Basic concepts of images.

Image Version

You can specify the image version, namely, the image tag, after selecting an existing image in the image list. If not specified, the latest version is used by default.

Scale

Configure the number of containers. Multiple containers can enhance application availability efficiently.

Network Mode

Select Default or host.

- **Default**: Namely, the bridge network mode. By connecting to the default bridge docker0, this mode assigns an independent network namespace for each container. You can see that eth0 is created in the container by using this configuration.
- **Host**: The network stack information that allows containers to use host. Containers created in this mode can view all the network devices on the host and have the full access permission to these devices.

For more information about Docker container network, see Docker container networking.

Restart

Specify the restart policy for the container. For more information, see restart.

- With this check box cleared, the system does not restart the container under any circumstances.
- With this check box selected, the system tries to restart the container until the specified container is running normally.

Command

Configure the command that is run by default after the container is started and configure the corresponding parameters. We recommend that you use the Exec format. The command is run if the container is started and docker run does not specify other commands. For more information, see command.

The default command specified by command is ignored if docker run specifies other commands.

Command has three formats:

- Exec format: `CMD ["executable","param1","param2"]`. This is the recommended format of command.
- `CMD ["param1","param2"]`: Use together with entrypoint command that is in the Exec format to provide the additional parameters.
- Shell format: `CMD command param1 param2`.

Entrypoint

The execution command to start containers. Entrypoint command can run the containers in the form of applications or services.

Entrypoint seems similar to CMD, both of which can specify the command to be run and the corresponding parameters. The difference is that entrypoint is not ignored and must be run, even if other commands are specified when running docker run.

Entrypoint has two formats:

- Exec format: `ENTRYPOINT ["executable", "param1", "param2"]`. This is the recommended format

of entrypoint.

- Shell format: `ENTRYPOINT command param1 param2` .

CPU Limit and Memory Limit

CPU 100 indicates one core and the unit of memory is the MB. You can configure the CPU limit and memory limit for a container, which facilitates your resource planning. The corresponding compose labels are `mem_limit` and `cpu_shares` . For more information, see Restrict container resources.

Capabilities

By default, the root permission in Docker containers has strict limits. With the Linux kernel capabilities, related permissions can be granted to the containers. For the parameters used to grant permissions to the containers, see Runtime privilege and Linux capabilities.

The related parameter commands are as follows:

- ADD field: Corresponds to the parameter `-cap-add: Add Linux capabilities` . Enter the Capability Key that containers can add in this field to add the permission to containers.
- DROP field: Corresponds to the parameter `-cap-drop: Drop Linux capabilities` . Enter the Capability Key that containers already have by default in this field to delete this permission from containers.

Container Config

Select the stdin check box to enable standard input for containers. Select the tty check box to assign an virtual terminal to send signals to the containers. These two options are usually used together, which indicates to bind the terminal (tty) to the container standard input (stdin). For example, an interactive program obtains standard input from you and then displays the obtained standard input in the terminal.

Port Mapping

Specify the port mapping between host and container, and select TCP or UDP as the protocol. Port mapping is used for the routing between container and host and implements the access to the container from outside.

Port mapping is the prerequisite for the configurations of simple routing and Server Load Balancer routing. The container provides external services by using the configured port mapping.

Web Routing

After a cluster is created in Container Service, the acsrouting application, including the routing service (namely, routing), is created automatically to provide the simple routing function. Each node has a service deployed. In a node, the acsrouting_routing_index container implements the routing forward in the cluster to route the HTTP services or HTTPS services. For more information, see Simple routing - supports HTTP and HTTPS.

> **Note** When exposing HTTP/HTTPS services, the specific host port can be unconfigured. The container port can be accessed directly by using the overlay network or Virtual Private Cloud (VPC).

Load Balancer

Control the routing access path on your own when configuring this parameter, including the routing mapping of Server Load Balancer frontend port > backend host port > container port.

Configure the port mapping in advance. Then, configure the mapping of `container_port` and `$scheme://$[slb_name|slb_id]:$slb_front_port`. For more information about the Server Load Balancer labels, see lb.

Data Volume

We recommend that you use data volumes to store the persistent data generated by containers, which is more secure, and easier to manage, back up, and migrate. For more information, see Use volumes.

- Select to create a data volume. Enter the host path or data volume name, the container path, and select RW or RO as the data volume permission.
- Enter the name and permission parameters of another service or container in the volumes_from field. For example, service_name:ro. If the access permission is not specified, the default permission is RW. For more information, see volume compose . After the configuration, containers can be granted to use the data volumes of another service or container.

Environment

Environment variables support the input form of key-value pairs and the formats such as array, dictionary, and boolean. For more information, see environment-variables.

You can configure the relevant environment variables for Docker containers. Environment variables can be used as flags and represent some parameters of environment deployment. You can also use environment variables to pass configurations and build automated deployment scripts.

Labels

Label applies metadata to Docker objects and can be used to build images, record license information, and describe the relationship among containers, data volumes, and network to implement powerful features.

Labels support the input form of key-value pairs and are stored in the format of strings. You can specify multiple labels for containers. The Docker native labels and Label description are supported.

Smooth Upgrade

Select whether or not to **enable** the smooth upgrade. Enabling smooth upgrade is equivalent to adding the label `rolling_update=true`. Use together with the label probe to make sure the containers can be updated successfully. For more information, seeprobe and Rolling_updates.

Across Multiple Zones

Select **Ensure** or **Try best**.

Select **Ensure** to deploy containers in two different zones. With this option selected, the container creation fails if the current cluster does not have two zones or the containers cannot be distributed in two zones because of limited machine resources.

You can also select Try best. Then, Container Service tries to deploy the containers in two different zones. Container can still be created even if the condition cannot be met.

If this parameter is not configured, Container Service deploys the containers in one zone by default. For more information, see High availability scheduling.

Auto Scaling

To meet the demands of applications under different loads, Container Service supports auto scaling for the service, which automatically adjusts the number of containers according to the container resource usage in the service.

For more information, see Container auto scaling.

8.3. Restrict container resources

One advantage of Docker containers is that they allow you to restrict resources, such as CPU, memory, and I/O performance. In swarm clusters, you can restrict the resources for applications.

You can restrict container resources in the Container Service console by setting interface parameters or configuring settings in orchestration templates.

Interface parameters

You can **restrict resources** Change application configurations when Create an application or changing the service configurations.

In swarm clusters, a single CPU core is equivalent to 100 CPUs. The unit of memory is MB.

Orchestration templates

In orchestration templates, you can use the `mem_limit` and `cpu_shares` labels to set CPU limit and memory limit.

CPU limit

A single CPU core is equivalent to 100 CPUs. If your machine is configured with 4 cores, the total number of available CPU resources is 400. In orchestration templates, you can use the `cpu_shares` label to specify CPU limit. `cpu_shares: 50` indicates 0.5 core.

Memory limit

You can use the `mem_limit` label to restrict memory usage. The unit is the byte and the minimum memory is 4 MB. If you set the memory limit and a container applies for a memory that exceeds the limit, the container is stopped because of OOM.

The following orchestration template demonstrates how to restrict CPU and memory.

```
n1:
  expose:
    - 443/tcp
    - 80/tcp
  image: 'nginx:latest'
  cpu_shares: 50 #0.5 core
  mem_limit: 536870912 #512MB
  labels:
    aliyun.scale: '1'
  restart: always
  volumes:
    - /var/cache/nginx
```

Resource scheduling

To ensure that containers can obtain sufficient specified resources, such as 0.5 CPU core and 512 MB of memory in the preceding example, Container Service reserves resources for containers. For example, a 4-core machine can schedule up to eight `cpu_shares=50` containers. If you create containers without specifying the `cpu_shares` and `mem_limit` labels, Container Service does not reserve resources for such containers by default.

Other resource limits

For other resource limits, see Docker Compose instructions.

8.4. High availability scheduling

To make the application have higher availability, Container Service supports scheduling containers of the same service in different zones. When a zone malfunctions, the application can still provide services.

You can specify the zone selection in the orchestration file by using the environment variables in the following formats:

- `availability:az==3`: The service must be distributed in at least three zones. The container creation fails if less than three zones are in the current cluster, or the service cannot be distributed in three zones because of limited machine resources.
- `availability:az==~3`: Try to distribute the service in three zones. Container can still be created even if the condition cannot be met.

> **Note** The deployment constraint only works for newly created containers. It does not work when containers created in the past change the configurations.

In the following example, the service must be distributed in at least two zones.

```
nnn:
  expose:
    - 443/tcp
    - 80/tcp
  image: 'nginx:latest'
  environment:
    - 'availability:az==2'
  labels:
    aliyun.scale: '8'Aliyun. Scale: '8'
  restart: always
  volumes:
    - /var/cache/nginx-/Var/Cache/nginx
```

8.5. Specified node scheduling

To deploy a service to a specified node, you can use the `constraint` keyword.

> **Note** The deployment constraint only works for newly created containers. It does not work when existing containers change the configurations.

In the following example, the service is deployed to node1.

```
web:
    image: 'nginx:latest'
    restart: always
    environment:
        - 'constraint:aliyun.node_index==1'
    ports:
        - 80
    labels:
        aliyun.scale: 2
```

Container Service supports the following expressions:

Expression	Description:
constraint:aliyun.node_index==1	Deploy the service to node1.
constraint:aliyun.node_index! =1	Do not deploy the service to node1.
constraint:aliyun.node_index==(1|2|3)	Deploy the service to node1, node2, or node3.
constraint:aliyun.node_index! =(1|2|3)	Deploy the service to a machine other than node1, node2, and node3.
affinity:image==~redis	Try to deploy the service to a machine with a Redis image. The image full name is supported. For example,Supports filling the full name of the mirror, such `registry.cn-hangzhou.aliyuncs.com/xxx/xxx` 。
affinity:service! =~redis	Try not to deploy the service to a machine with a Redis service. For more information, see Service deployment constraints (affinity:service).

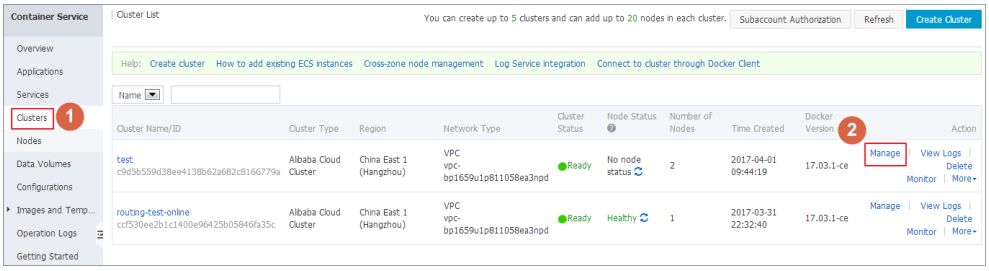
8.6. Schedule an application to specified nodes

To deploy an application to specified nodes, we recommend that you use user tags and the `constraint` keyword to make the deployment configurations.

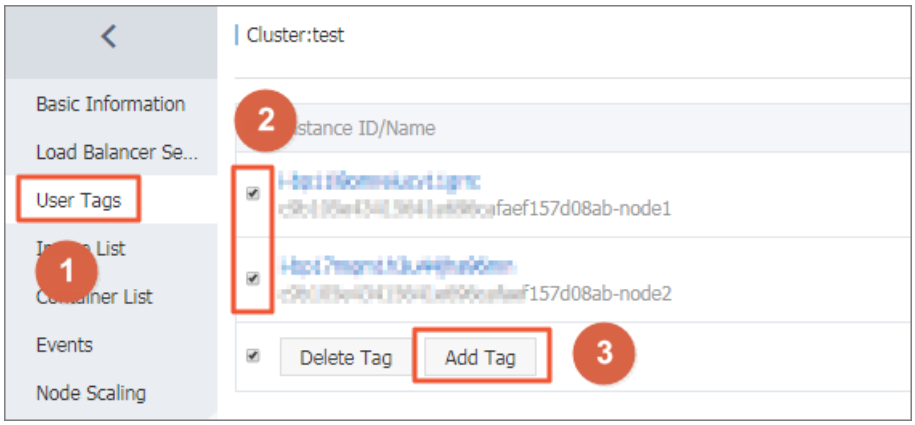
> **Note**
>
> - The deployment constraint only works for newly created containers. It does not work when existing containers change the configurations.
> - After you use a user tag to deploy an application, deleting the user tag does not affect the deployed application, but will affect the next deployment of the application. Proceed with caution when deleting user tags.

Procedure

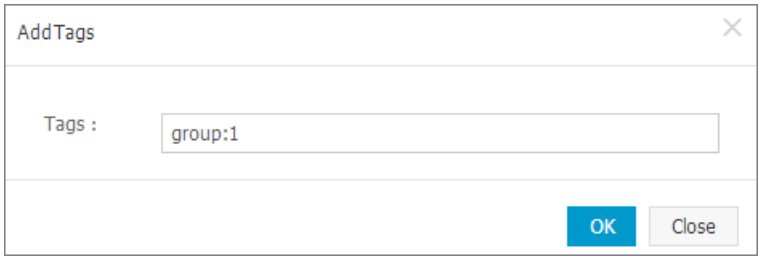
1. Add user tags for nodes.
   i. Log on to the Container Service console.
   ii. Click Swarm > **Clusters** in the left-side navigation pane.
   iii. Click **Manage** at the right of the cluster.



   iv. Click **User Tags** in the left-side navigation pane.
   v. Select the nodes that you want to deploy the application and then click **Add Tag**.



   vi. Enter your tag key and tag value, and then click **OK** to add user tags for the selected nodes.



2. Create an application by clicking **Create with Orchestration Template**. Configure the `constrai`

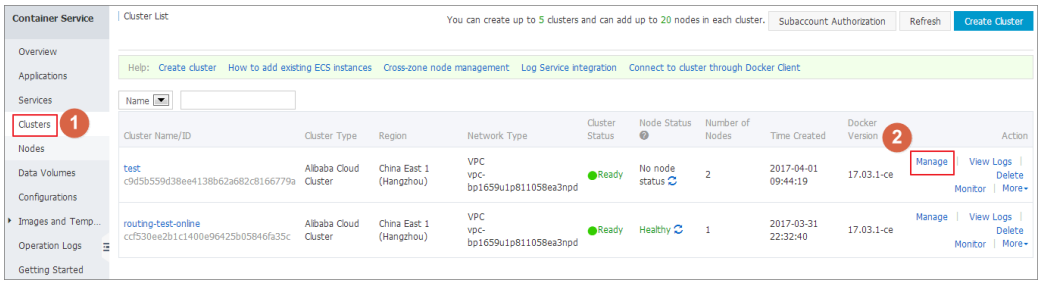
`nt` keyword in the template.

For information about how to create an application, see Create an application.

```
environment:
  - constraint:group==1 #Indicates to deploy the application on all the nodes with the
"group:1" tag
```

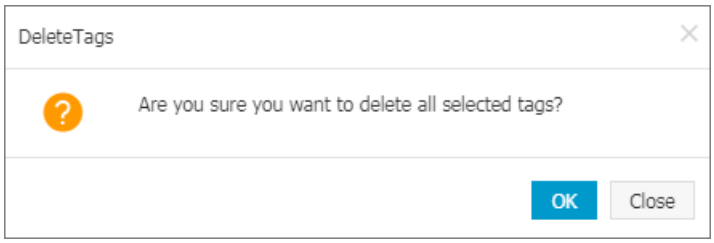
Delete a user tag

1. Log on to the Container Service console.
2. Click Swarm > **Clusters** in the left-side navigation pane.
3. Click **Manage** at the right of the cluster.



4. Click **User Tags** in the left-side navigation pane.
5. Select the nodes that you want to delete the user tags and then click **Delete Tag**.

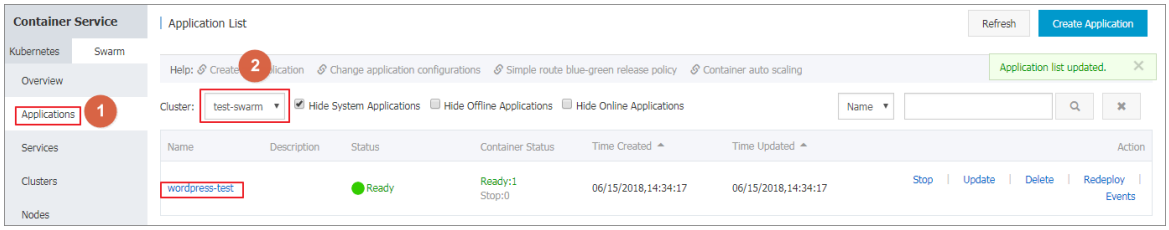
6. The confirmation dialog box appears. Click **OK**.



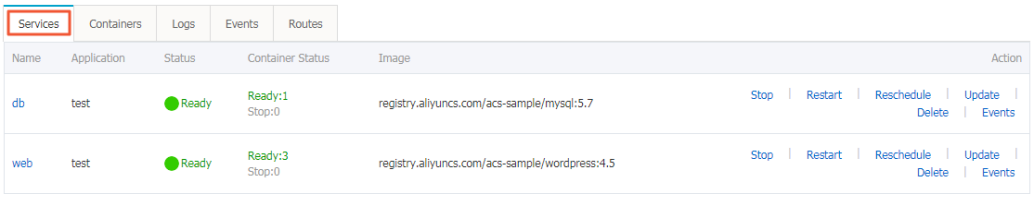
8.7. View application details

Procedure

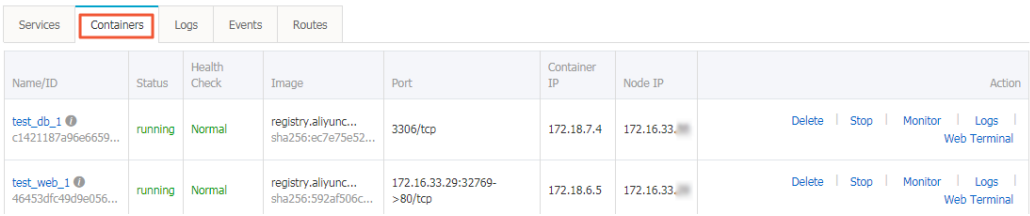
1. Log on to the Container Service console.
2. Click **Applications** in the left-side navigation pane.
3. Select the cluster in which the application you want to view resides from the Cluster list.
4. Click the application name.



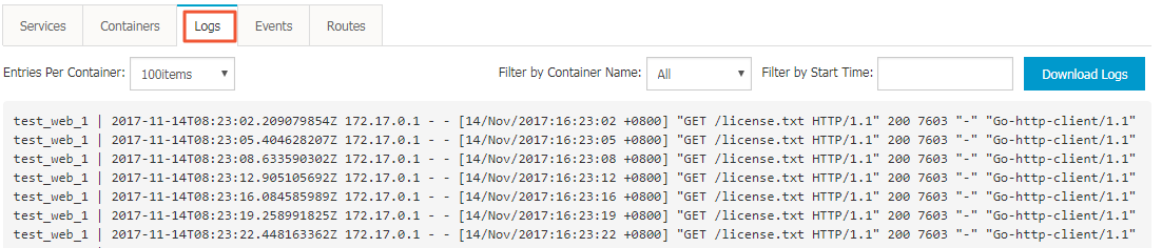
5. Click the **Services** tab to view the services of the application.



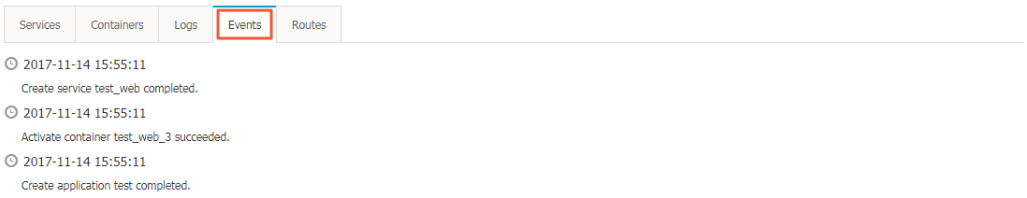
6. Click the **Containers** tab to view the containers of the application.



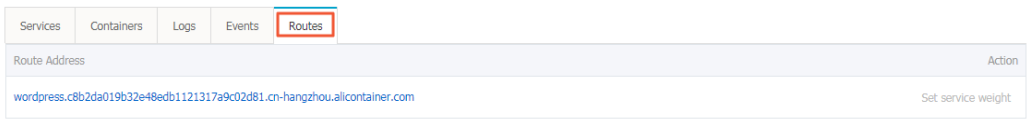
7. Click the **Logs** tab to view the logs of the application.



8. Click the **Events** tab to view the events of the application.



9. Click the **Routes** tab to view the route address of the application.



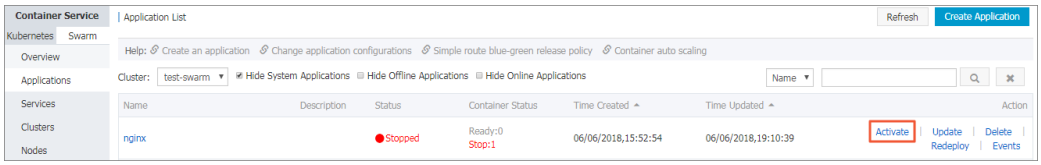
8.8. Stop or activate an application

Context

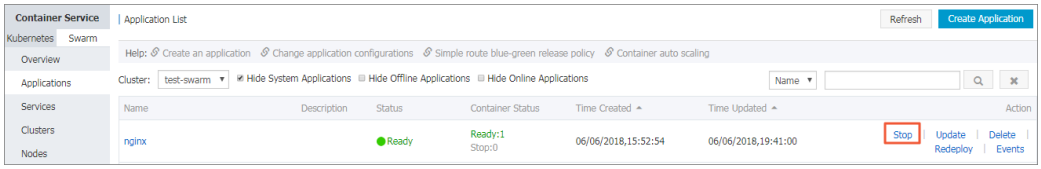
Procedure

1. Log on to the Container Service console.
2. Click Swarm > **Applications** in the left-side navigation pane.
3. Select the cluster in which the application you want to stop or activate resides from the Cluster drop-down list.
4. Activate or stop the application according to the status.

   Click **Activate** at the right of the application that you want to activate.



   Click **Stop** at the right of the application that you want to stop.

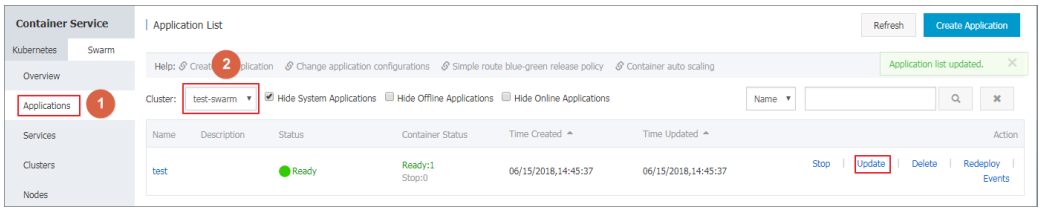


5. In the displayed dialog box, click **OK**.

8.9. Change application configurations

Procedure

1. Log on to the Container Service console.
2. Click **Applications** in the left-side navigation pane.
3. Select the cluster where the application you want to change configurations resides from the Cluster list.
4. Click **Update** at the right of the application you want to change configurations.



5. The Change Configuration dialog box opens. Modify the configurations.

   Note You must update the **Version**. Otherwise, the **OK** button is not available.

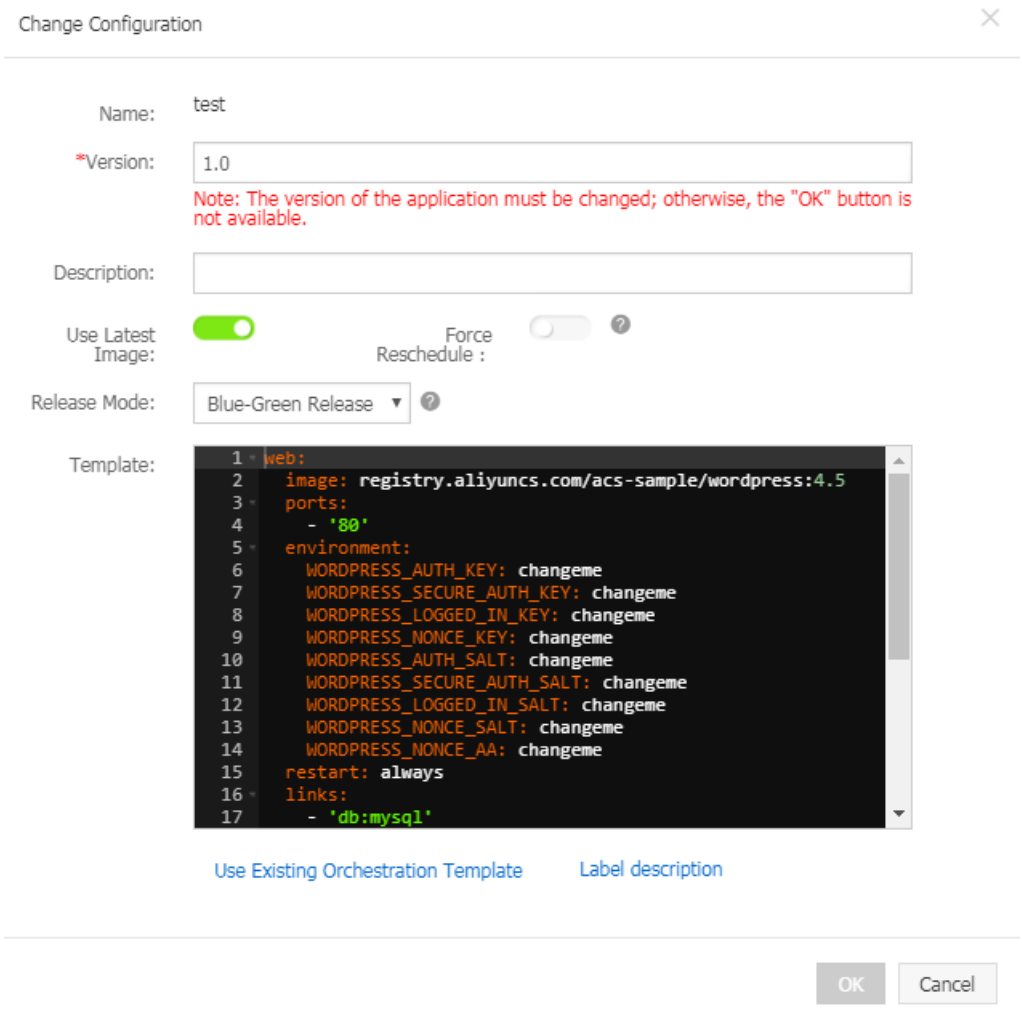
   - \: By default, to avoid losing the container data in the local data volume on the current machine, Container Service restarts the container or recreates a container on the current machine when you change the application configurations. To schedule the container to another machine, turn

on the Force Reschedule switch. Then, Container Service schedules the container to another machine according to your scheduling settings in the Template.

> **Note** After scheduling the container to another machine by turning on the Force Reschedule switch, the container data in the local data volume on the former machine are lost. So proceed with caution. So proceed with caution.

- **Use Existing Orchestration Template**: Click **Use Existing Orchestration Template** to select a template to change the application configurations. A confirmation dialog box opens. Click Confirm.

> **Note** The new template will overwrite the current template. Click OK in the Change Configuration dialog box.



What's next

If the application is not updated after you change the configurations, you can redeploy the application to apply the configuration modifications. For more information, see Redeploy an application.

8.10. Redeploy an application

You can redeploy an application after deploying it, if necessary. Redeploying an application re-pulls the image used by the application. Therefore, if you update the application image after deploying the application, redeployment will use the updated image to deploy the application.

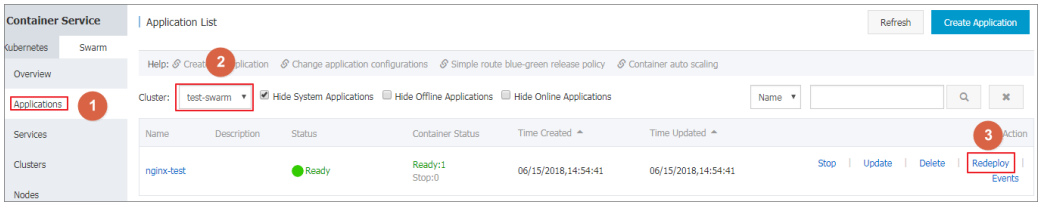
> **Note** Redeployment does not update the data volume, which means the old data volume of the host is still used. Therefore, if you mount a data volume to the host and change the configurations of the data volume in the new image, the new configurations will not take effect after the redeployment.

You must redeploy an application in the following situations:

- You update the image after deploying the application and want to deploy the application according to the updated image.
- You stop or delete some containers and want to activate or recreate those containers. During the redeployment, Container Service activates the stopped containers and recreates the deleted containers.

Procedure

1. Log on to the Container Service console.
2. Click Swarm > **Applications** in the left-side navigation pane.
3. Select the cluster where the application you want to redeploy resides from the Cluster drop-down list.
4. Click **Redeploy** at the right of the application you want to redeploy.

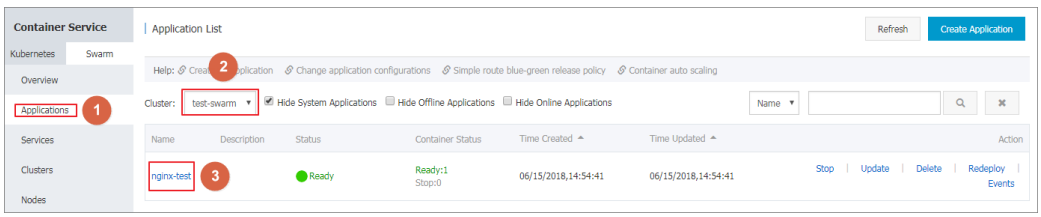


5. A confirmation dialog box appears. Click **OK**.

Check whether the redeployment succeeds

To confirm whether the redeployment is successful or not, view the image `sha256` to check whether the container image after the redeployment is the latest one.

1. Log on to the Container Service console.
2. Click Swarm > **Applications** in the left-side navigation pane.
3. Select the cluster where the redeployed application resides from the Cluster drop-down list.
4. Click the application name.



5. Click the **Containers** tab to view the image `sha256`.

The redeployment is successful if the container image is the latest one.

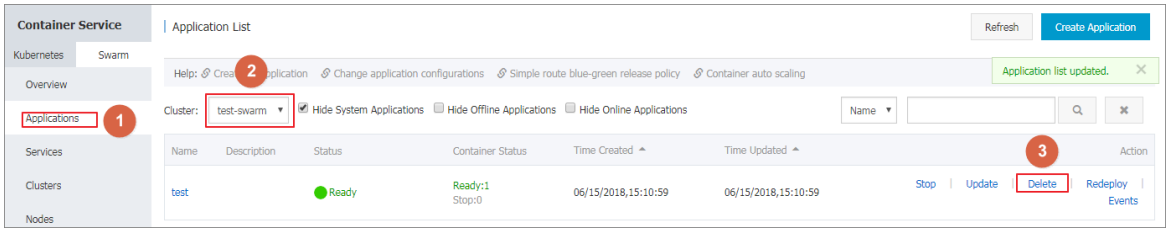
8.11. Delete an application

Context

You can delete applications that will not be used.

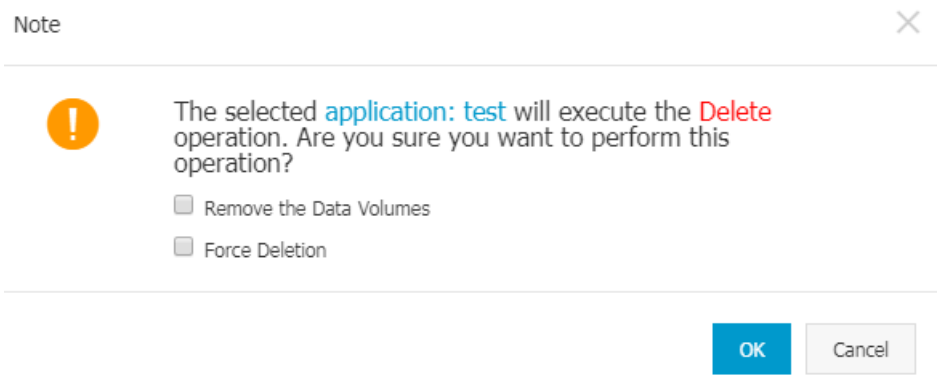
Procedure

1. Log on to the Container Service console.
2. Click **Applications** in the left-side navigation pane.
3. Select the cluster where the application you want to delete resides from the Cluster list.
4. Click **Delete** at the right of the application you want to delete.



5. A confirmation dialog box opens. Click **OK**.

   Select the **Remove the Data Volumes（volume）** check box to delete all the data volumes related to this application. The named data volumes cannot be deleted.



8.12. Run offline tasks

Container Service abstracts the basic model of offline computing and provides the offline computing function based on Docker containers.

The core functions include:

- Job orchestration
- Job scheduling and lifecycle management

- Integration of storage and other functions

Basic concepts

The following table compares the concepts of offline applications with those of online applications.

Concept	Offline application	Online application
Container	Task execution unit	Service execution unit
Operation history	Execution history of tasks that encountered an error and were re-executed	None
Service (Task)	A special function that can be divided into several containers for execution	A group of containers with the same functions
Application (Job)	A combination of several tasks	A combination of several services

In a word, an offline job contains several tasks. Each task can be executed by several containers. Each container can have multiple operation histories. By contrast, an online application contains several services and each service can be provided by several containers simultaneously.

Docker Compose-based job orchestration

Similar to online applications, Docker Compose can be used to describe and orchestrate jobs. Docker Compose supports the vast majority of Docker functions, such as:

- CPU, memory, and other resource limits
- Data volumes
- Environment variables and labels
- Network models and port exposure

In addition, Alibaba Cloud Container Service has expanded the following functions:

- Container quantity: The number of containers that each task is divided into.
- Number of retries: The number of retries made by each container.
- Remove containers: Whether or not to delete a container after it has completed its run. You can select the following policies: `remove-finished` (deletes the container after it completes its run), `remove-failed` (deletes the container that fails the run), `remove-all` (deletes all of the containers), and `remove-none` (does not delete any container).
- DAG model task dependencies: Tasks in a job can have dependencies between each other. Tasks that others depend on are executed first.

The following is an example of offline job Docker Compose.

```
version: "2"
labels:
  aliyun.project_type: "batch"
services:
  s1:
    image: registry.aliyuncs.com/jimmycmh/testret:latest
    restart: no
    cpu_shares: 10
    mem_limit: 100000000
    labels:
      aliyun.scale: "10"
      aliyun.retry_count: "20"
      aliyun.remove_containers: "remove-all"
  s2:
    image: registry.aliyuncs.com/jimmycmh/testret:latest
    cpu_shares: 50
    mem_limit: 100000000
    labels:
      aliyun.scale: "4"
      aliyun.retry_count: "20"
      aliyun.remove_containers: "remove-finished"
      aliyun.depends: "s1"
```

Note:

- Only Docker Compose 2.0 is supported.
- Add the label `aliyun.project_type: "batch"` at the job level. If this label is not added or its value is not `batch`, the application is considered as an online application.
- Any value of `restart` will be changed to `no`.
- Use the `aliyun.depends` label to specify dependencies. A task can depend on several other tasks. Separate the tasks by using commas (,).
- The default value of `aliyun.retry_count` is 3.
- The default value of `aliyun.remove_containers` is `remove-finished`.

Job lifecycle management

The container status is determined by the container running and exit status. The task status is determined by the statuses of all the containers in the task. The job status is determined by the statuses of all the tasks in the job.

Container status

- Running: The container is running.
- Finished: The container exits and `ExitCode==0`.
- Failed: The container exits and `ExitCode!=0`.

Task status

- Running: A container is running.
- Finished: All containers are finished.
- Failed: The number of failures of a container exceeds the set value.

Job status

- Running: A task is running.
- Finished: All tasks are finished.
- Failed: A task failed.

The preceding statuses can all be retrieved by means of API to facilitate automated Operation and Maintenance(O&M).

Shared storage

Data is shared and exchanged between containers and tasks. Shared storage can be used to resolve this issue. For example, when running an MR job on Hadoop, HDFS is used for data exchange. In Container Service, two types of shared storage can be used. Their features and application scenarios are compared as follows:

Storage	Advantages	Disadvantages	Scope
OSSFS data volumes	Cross-host sharing.	Low read/write and ls performance. Modifying a file causes the file to be overwritten.	Shared configuration files. Attachment upload.
NAS data volumes	Cross-host sharing; On-demand capacity expansion; high performance, high reliability; high Mount speed	Slightly higher cost	Heavy Io applications that need to share data, such as file servers, etc; heavy Io applications that require rapid migration, such as databases, etc.
A third-party storage integrated by you, such as Portworx	Virtualize the cloud disks in the cluster into a large shared disk. High performance. Snapshots, multiple copies.	Certain O&M capabilities are required.	I/O-intensive applications that need data sharing, such as file servers. I/O-intensive applications that need fast migration, such as databases.

For more information about how to use the data volumes, see the following documents:

- Create an OSSFS data volume
- Creating NAS data volumes
- Use OSSFS data volumes to share WordPress attachments

Integrate with Log Service and CloudMonitor

Log and monitoring are important tools used to analyze offline jobs. Alibaba Cloud Container Service integrates with the CloudMonitor function. Adding a label in the orchestration template can collect CPU, memory, and other data of containers to CloudMonitor. For more information, see the following documents:

- Enable Log Service
- Container monitoring service

Procedure

1. Log on to the Container Service console and create a cluster.

   For more information, see Create a cluster.
2. Click **Applications** in the left-side navigation pane and then click **Create Application** in the upper-right corner.
3. Complete the basic information for the application and then click **Create with Orchestration Template**.
4. Use the preceding orchestration template and then click **Create and Deploy**.
5. On the **Application List** page, click the application name to view the application running status.

8.13. Timing tasks

Having a timing task is a common requirement. Generally, select one or more machines and realize the timing tasks by using crontab. However, for large-scale or a large number of timing tasks, this method has many limits such as:

- Low reliability. If one machine goes down, all the timing tasks on this machine cannot be executed.
- No scheduling function. Loads among machines might not be balanced.
- No retry mechanism. Tasks might fail to be run.
- Cannot run large-scale distributed tasks.

On the basis of the offline tasks, Container Service provides the function of timing tasks, with which you can solve the preceding problems by some simple descriptions. For more information on offline tasks, see Run offline tasks.

> **Note** You can only use this function if you have upgraded your Agent since October 25th, 2016 or the cluster is newly created.

Timing task description based on Docker Compose

The same as offline tasks, timing tasks are also based on Docker Compose. You can realize the timing function by adding the `aliyun.schedule` label in the orchestration template. The following example shows.

> **Note** When you create or update a timing task, the system will not pull the latest image. This is because using the latest image will make the same task use different images at different time, which might cause your troubleshooting complicated. We recommend that you update your timing task image by updating the image tag.

```
version: "2"
labels:
  aliyun.project_type: "batch"
  aliyun.schedule: "0-59/30 * * * * *"
services:
  s1:
    image: registry.aliyuncs.com/jimmycmh/busybox:latest
    labels:
      aliyun.scale: "5"
      aliyun.retry_count: "3"
      aliyun.remove_containers: "remove-all"
    command: date
```

Note:

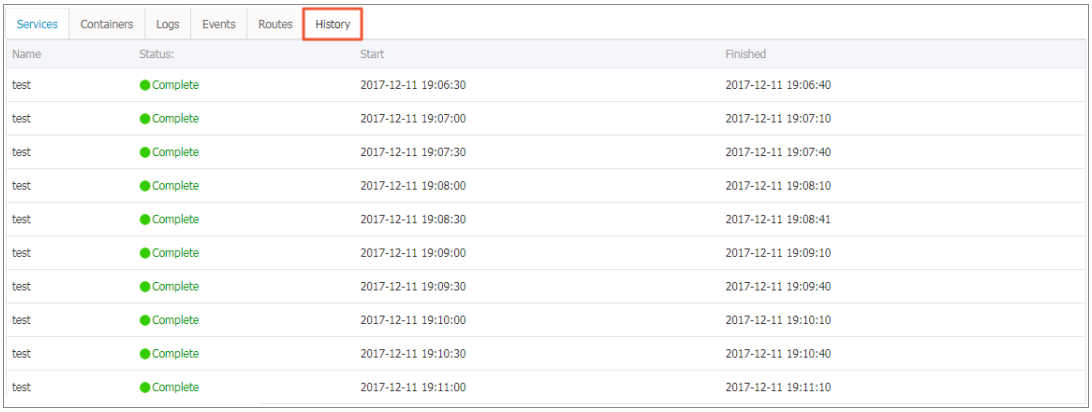
- `aliyun.schedule: "0-59/30 * * * * *"` indicates running this task every 30 seconds. The format of schedule is the same as that of crontab (but note that the format of schedule is second minute hour day month week, and that of crontab on Linux is minute hour day month week) and uses Beijing time.
- The timing function is only applicable to offline tasks. Therefore, the system automatically adds the `aliyun.project_type: "batch"` label after you add the `aliyun.schedule` label. So the `aliyun.project_type: "batch"` label in the preceding example can be omitted.
- In addition, all the functions in offline tasks can still be used in timing tasks (for example, scale, retry_count, remove_containers). For the specific meanings of these labels, see Run offline tasks.

Execution process

After a timing task is created, the application is in the status of Waiting. When the specified time of the task is reached, the task is started. The subsequent status changes are the same as the offline applications. The application status repeats this process when the next execution time is reached.

For the same timing task, only a single instance can be executed at a time. If the task execution time is longer than the task execution period (for example, the execution time of the preceding task is longer than 30s), the next execution enters the execution queue. When the length of the execution queue is greater than 3, this execution will be discarded.

You can click the History tab on the application details page to view the execution history and results. Only the last 10 items of execution history are kept in the list.



High availability

The timing task controller adopts the master-slave mode. The control function is switched to the slave controller when the master controller malfunctions.

If the task execution time is in the master-slave switch period, the task will be executed until the switch is completed. If the task needs to be run for several times in the master-slave switch period, the task will be executed only once after the switch is completed. Therefore, to ensure that the task is not lost, do not design tasks with a repetition period less than one minute.

8.14. Default system application list

Application name	Chinese name	Services contained	Brief introduction
acsrouting	acsrouting	routing	Provides the request routing service with layer-7 protocol, which consists of Server Load Balancer and an HAProxy container. After the domain name is correctly configured, the request can be sent to the specified container.
acslogging	acslogging	logtail, logspout	Integrates with Alibaba Cloud Log Service to upload the logs printed by applications in containers to Alibaba Cloud Log Service for storage, facilitating you to query and analyze the logs. For how to use acsmonitoring, see Enable Log Service.
acsmonitoring	Monitoring service	acs-monitoring-agent	Integrates with Alibaba Cloud CloudMonitor and currently popular third-party open-sourced monitoring frameworks, facilitating you to query the monitoring information and configure the monitoring alarms. For how to use acsmonitoring, see Container monitoring service.

Application name	Chinese name	Services contained	Brief introduction
acsvolumedriver	Data volume	volumedriver	Integrates with Alibaba Cloud CloudMonitor and currently popular third-party open-sourced monitoring frameworks, facilitating you to query the monitoring information and configure the monitoring alarms. For how to use acsmonitoring, see Container monitoring service.

9.Configurations

9.1. Create a configuration

Context

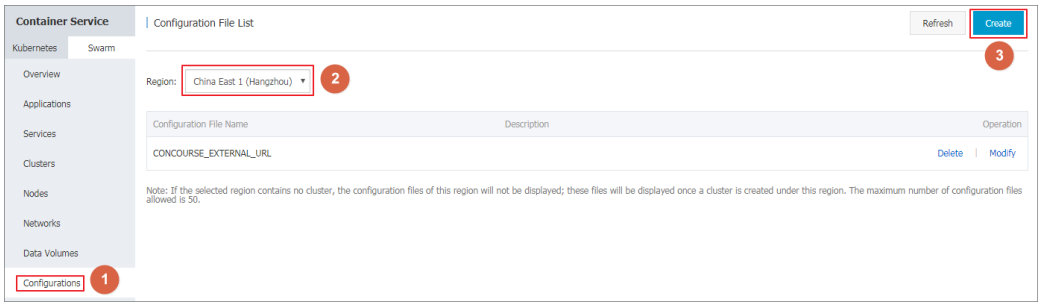
Container Service supports creating configurations and transmitting the configurations by configuring the parameters, which facilitates you to manage multiple container environment variables.

Limits

- You can only select a region of an existing cluster when creating a configuration.
- The configuration creation fails and the system reports an error if clusters in the selected region do not have any nodes.
- You cannot view the configurations in a region on the Configuration File List page if clusters in the region are all deleted or do not have any nodes. You can view the configurations in the region on the Configuration File List page when a cluster in the region contains nodes.

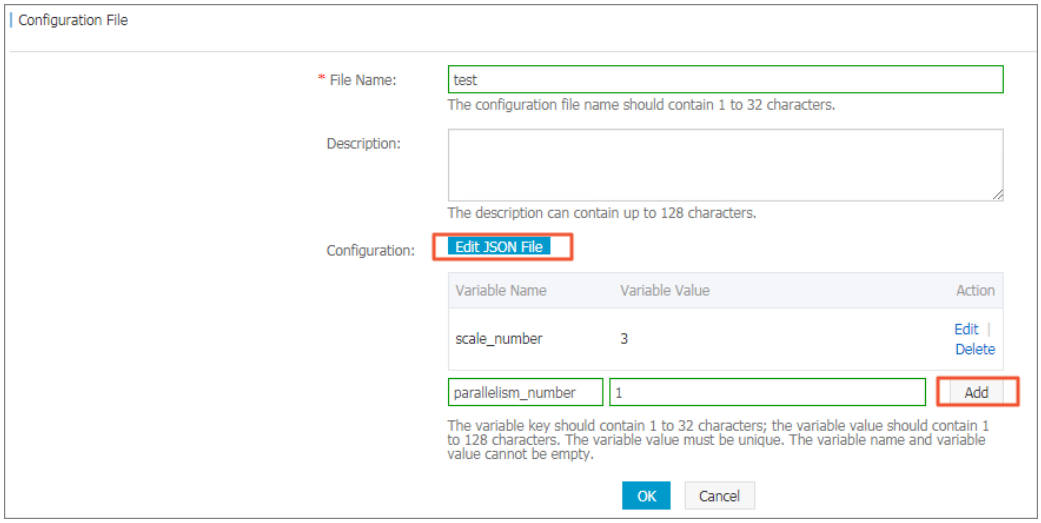
Procedure

1. Log on to the Container Service console.
2. Click Swarm > **Configurations** in the left-side navigation pane.
3. Select the region where you want to create a configuration from the Region drop-down list and then click **Create** in the upper-right corner.



4. Enter the configuration file information and click OK.
   - **File Name**: It can contain 1-32 characters.
   - **Description**: It can contain up to 128 characters.
   - **Configuration**: You can add up to 50 configurations in a region. Enter the **Variable Name** and **Variable Value**, and then click **Add** on the right.

     You can also click **Edit JSON File**. The JSON Format dialog box appears. Enter the configurations in the dialog box and click **OK**.



In this example, the variables `scale_number` and `parallelism_number` are set, which are used to pass the parameters of the Alibaba Cloud extension labels `scale` and `rolling_updates` respectively.

9.2. Modify configurations

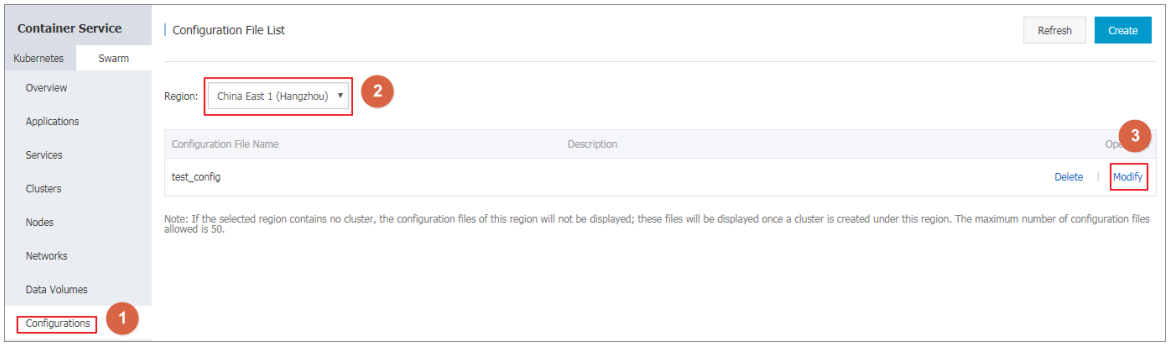
Context

You can modify the configurations of a configuration file.

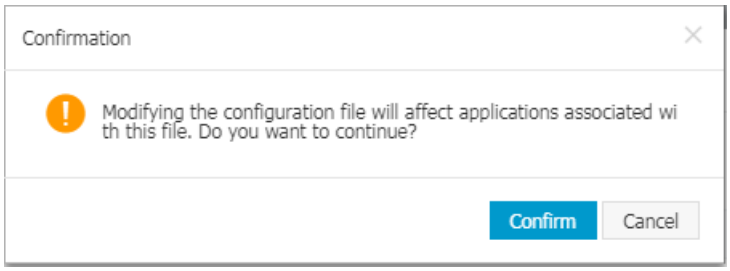
> **Note** Modifying the configuration file affects applications that use this file.

Procedure

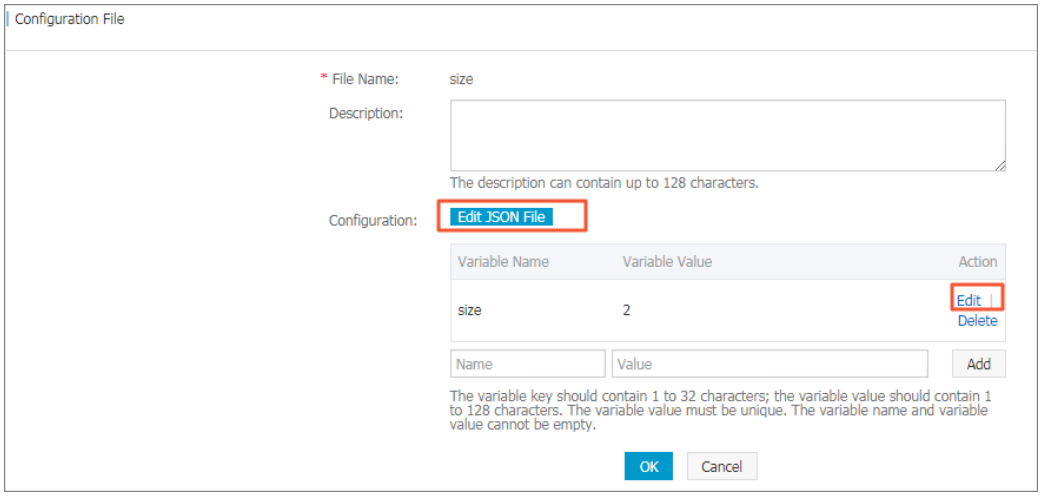
1. Log on to the Container Service console.
2. Click Swarm > **Configurations** in the left-side navigation pane.
3. On the Configuration File List page, select a region from the Region drop-down list.
4. Click **Modify** at the right of the configuration that you want to modify.



5. Click **Confirm** in the displayed dialog box.



6. Modify the configurations.
   - Click **Edit** at the right of the configuration you want to modify. Update the configuration and then click **Save**.
   - You can also click **Edit JSON File**. Click **OK** after making the modifications.



7. After modifying the configurations, click **OK**.

9.3. Implement multiple environments by using configurations

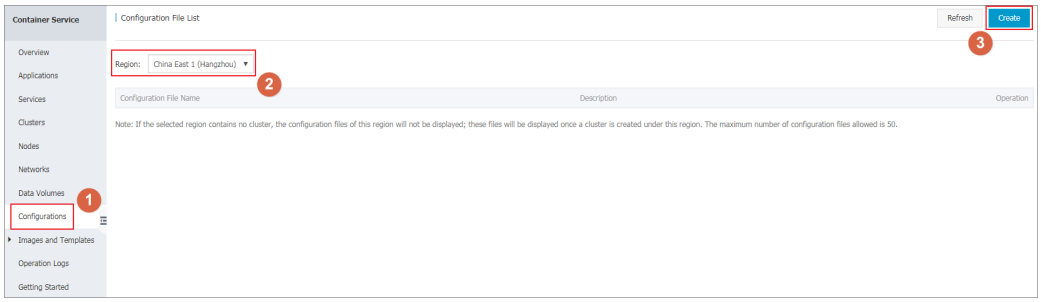
An application consists of codes and configurations. After an application is containerized, the configurations are usually transmitted by using container environment variables to deploy multiple applications using the same image and different configurations.

Limits

- When associating a configuration file with an application, make sure the configuration file is in the same region as the application.
- Currently, associating a configuration file when creating an application is only available when you create the application by using an orchestration template.

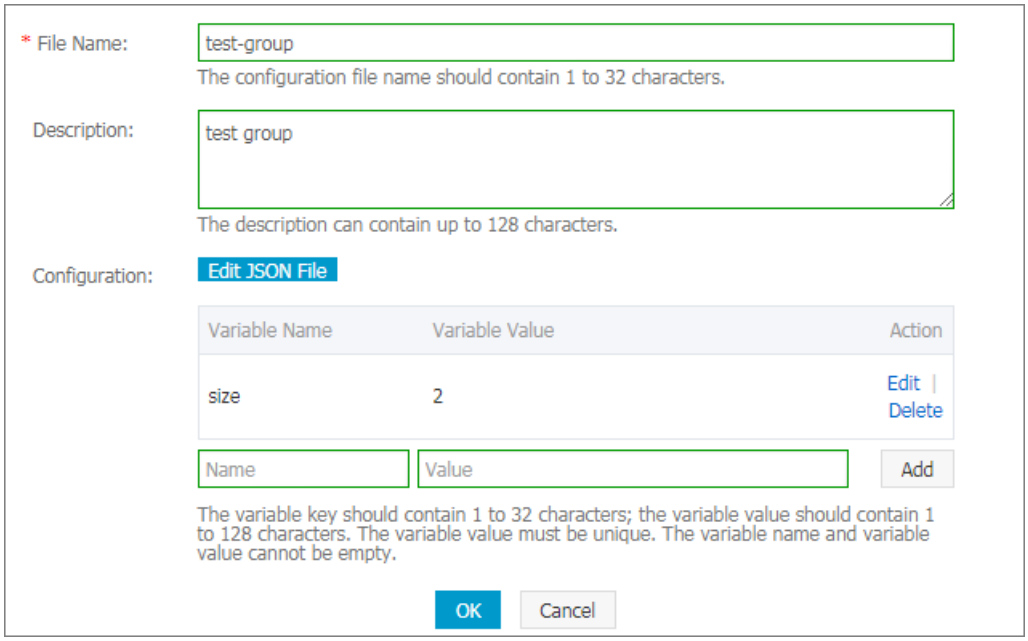
Create an application

1. Log on to the Container Service console.
2. Under Swarm, click **Configurations** in the left-side navigation pane. Select the region in which you want to create a configuration from the Region list and click **Create**.



3. Complete the settings and then click **OK**.
   - **File Name**: It can contain 1-32 characters.
   - **Description**: It can contain up to 128 characters.
   - **Configuration**: You can add up to 50 configurations in a region.

   In this example, the `size` variable is set.

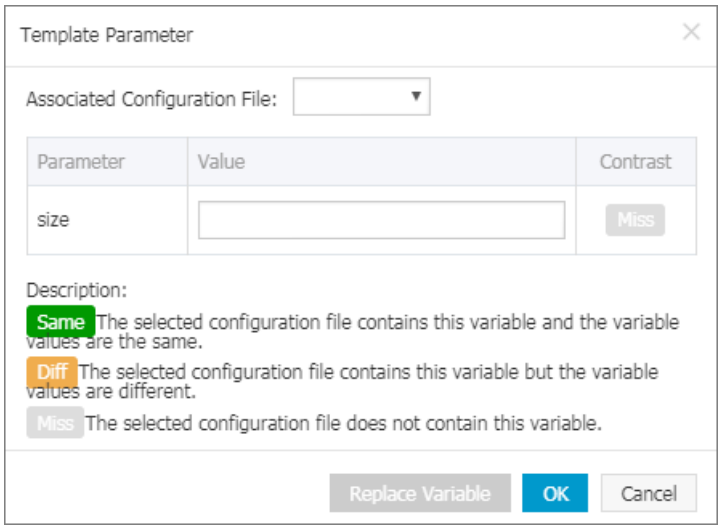


4. Under Swarm, click **Applications** in the left-side navigation pane. Select the cluster in the same region as the created configuration from the Cluster list and click **Create Application**.
5. Enter the basic information of the application and click **Create with Orchestration Template**.
6. Enter the following orchestration template and then click **Create and Deploy**.

   Wherein, `size` is a dynamic variable and will be overwritten by the value in the configuration.

```
busybox:
 image: 'busybox'
 command: 'top -b'
 labels:
     aliyun.scale: $size
```

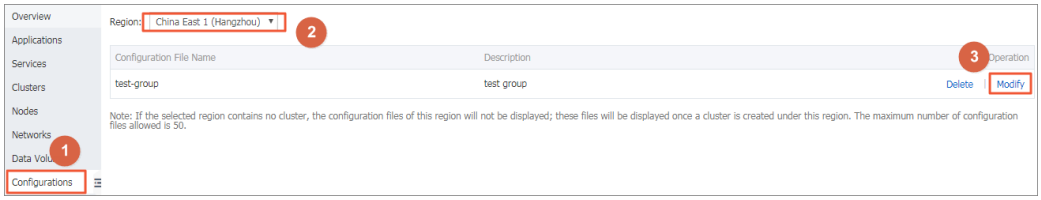
7. The dialog box appears. Select the configuration file to be associated with from the **Associated Configuration File** drop-down list. Click Replace Variable and click **OK**.



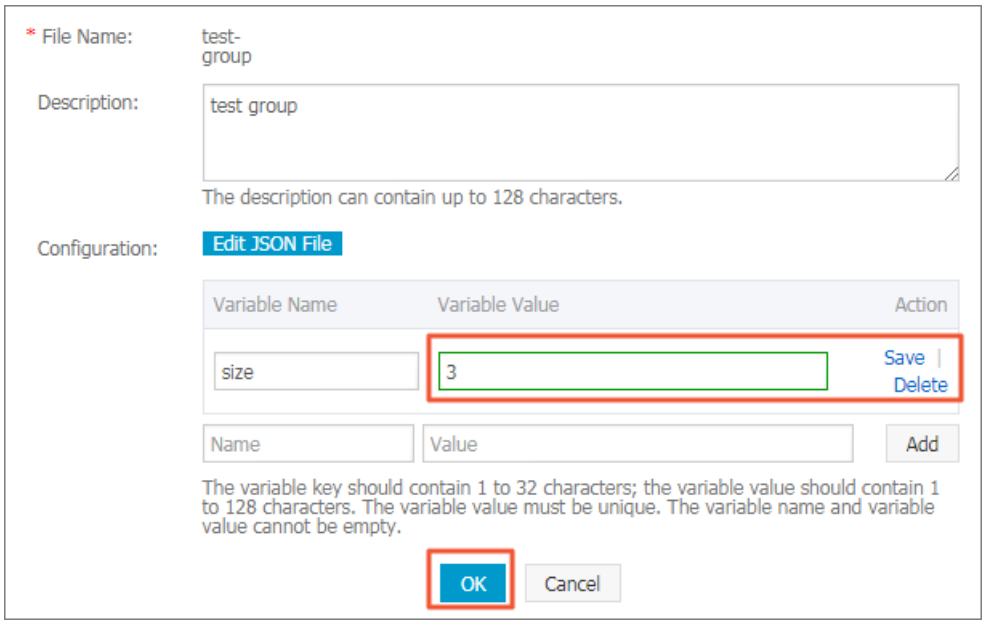
Update an application

If you associated a configuration file when creating an application, you can update the application by modifying the configuration file and redeploying the application.

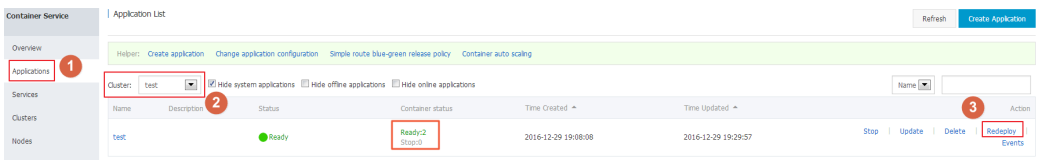
1. Log on to the Container Service console.
2. Under Swarm, click **Configurations** in the left-side navigation pane. Select the region in which the configuration you want to modify resides from the Region list, and click **Modify** at the right of the configuration.



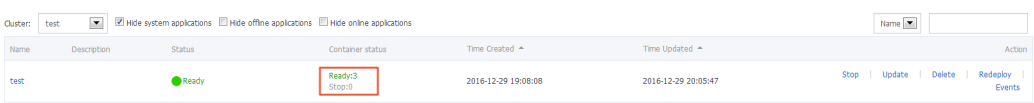
3. Click **Confirm** in the displayed dialog box.
4. Click **Edit** (changes to **Save** after you click it) at the right of the variable you want to modify. Modify the variable value. Click **Save** and then click **OK**.



5. Under Swarm, click **Applications** in the left-side navigation pane. Select the cluster in the same region as the created configuration, and then click **Redeploy** at the right of the application.



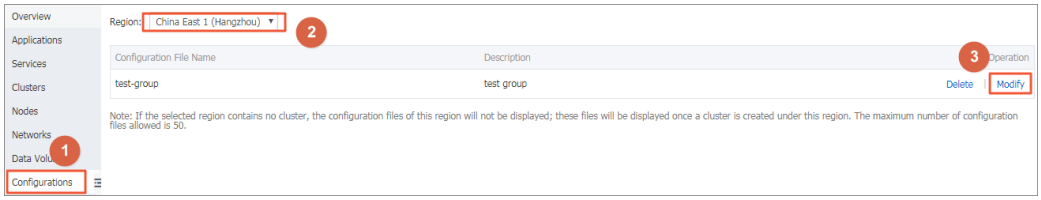
After the application is updated, the number of containers changes to three.



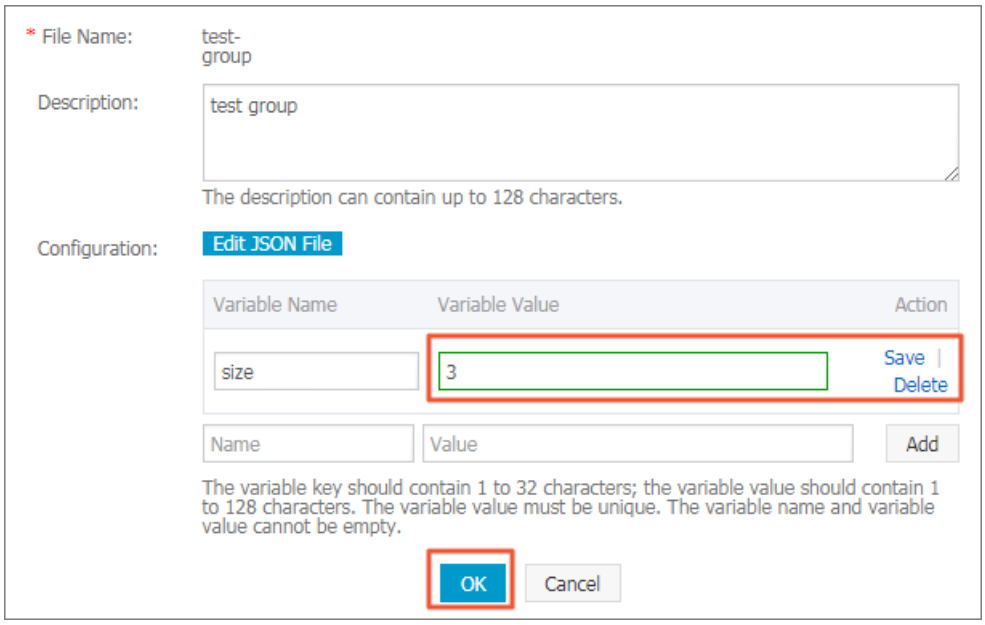
Trigger an update

If you associated a configuration file when creating an application, you can redeploy the application by using the redeployment trigger.

1. Log on to the Container Service console.
2. Under Swarm, click **Configurations** in the left-side navigation pane. Select the region in which the configuration you want to modify resides from the Region list, and click **Modify** at the right of the configuration.

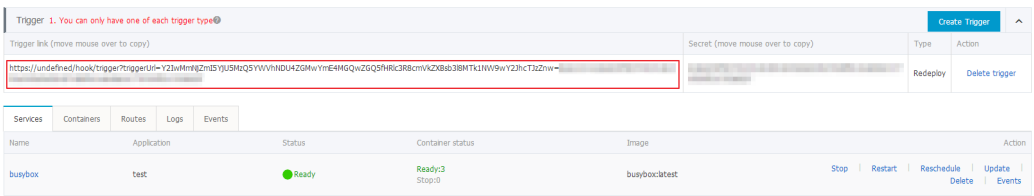


3. Click **Confirm** in the displayed dialog box.
4. Click **Edit** (changes to **Save** after you click it) at the right of the variable you want to modify. Modify the variable value. Click **Save** and then click **OK**.



5. Create a redeployment trigger.

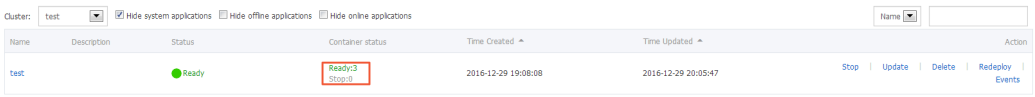
   For how to create a trigger, see Triggers.



6. Initiate the redeployment trigger.

```
curl "https://cs.console.aliyun.com/hook/trigger?triggerUrl=Y2ViZDhkZTIwZGMyMjRmOTM4NDIzMTgwMzI3NmIwM2IxfHRlc3QtZ3JvdXB8c2NhbGluZ3wxOXZwYzNmOXFiNTcwfA==&secret=466242376775654951546d6451656a7a66e7f5b61db6885f8d15aa64826672c2"
```

   After the application is updated, the number of containers changes to three.



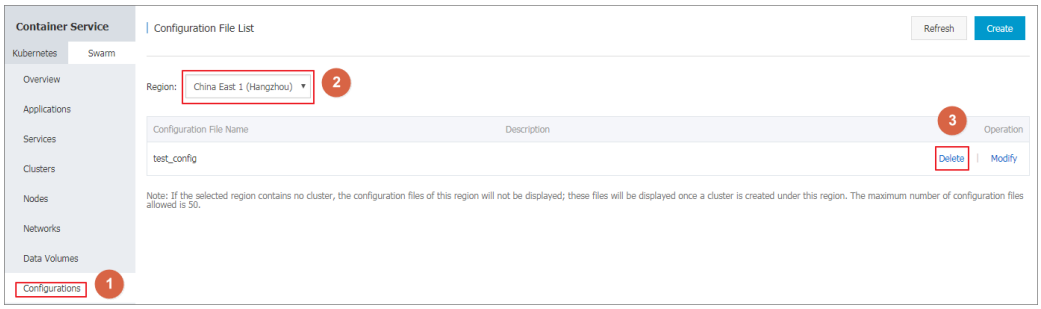
9.4. Delete a configuration

Context

You can delete a configuration that is no longer in use.

Procedure

1. Log on to the Container Service console.
2. Click **Configurations** in the left-side navigation pane.
3. On the Configuration File List page, select a region from the Region list.
4. Click **Delete** at the right of the configuration that you want to delete.



10.Services

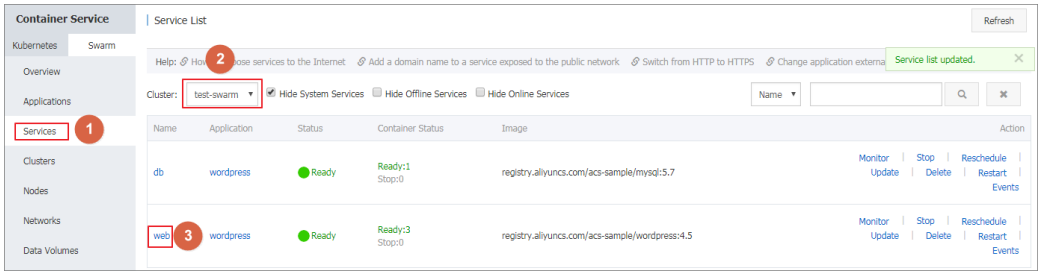
10.1. Instructions

An application is composed of one or more services. You can update the application by changing the application configurations or changing the configurations of a service.

10.2. View service details

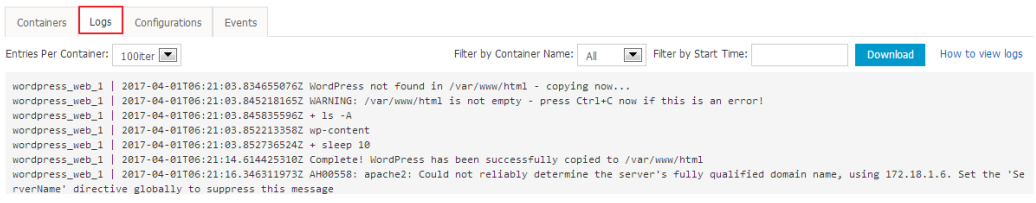
Procedure

1. Log on to the Container Service console.
2. Click Swarm > **Services** in the left-side navigation pane.
3. Select the cluster in which the service you want to view resides from the Cluster drop-down list.
4. Click the service name. As shown in the following figure.

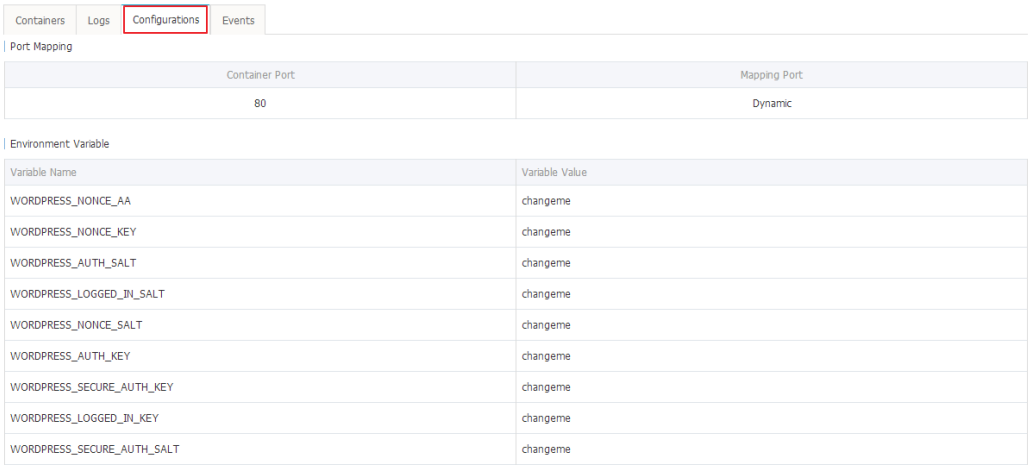


5. The service details page appears and you can view all the containers of this service.

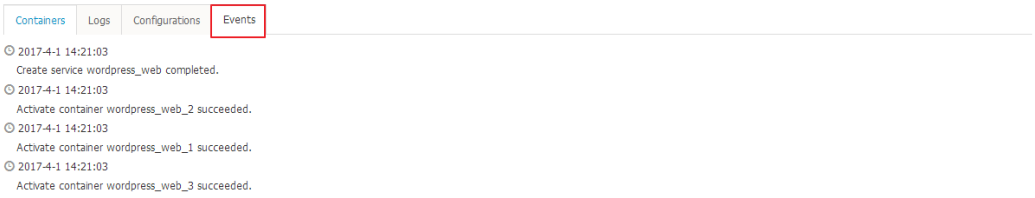
6. Click the **Logs** tab to view the service-level logs.



7. Click the **Configurations** tab to view the configurations of the service.



8. Click the **Events** tab to view the events of the service.

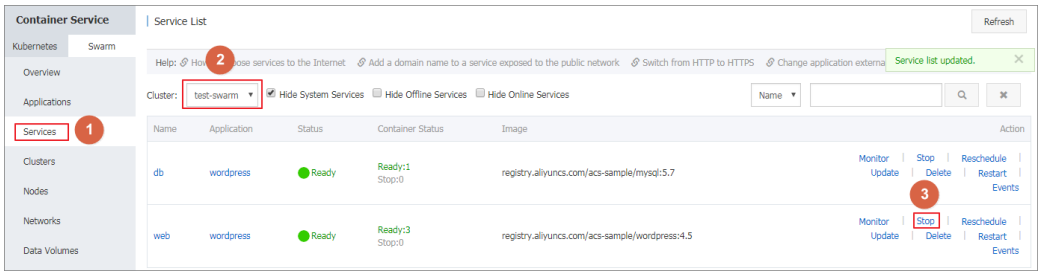


10.3. Activate or stop a service

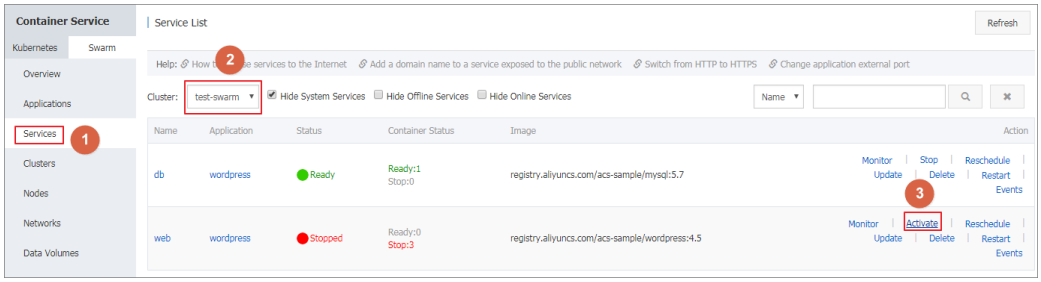
Procedure

1. Log on to the Container Service console.
2. Click Swarm > **Services** in the left-side navigation pane.
3. Select the cluster in which the service you want to activate or stop resides from the Cluster drop-down list.
4. Activate or stop the service according to the status.

   Click **Stop** at the right of the service that you want to stop.



   Click **Activate** at the right of the service that you want to activate.

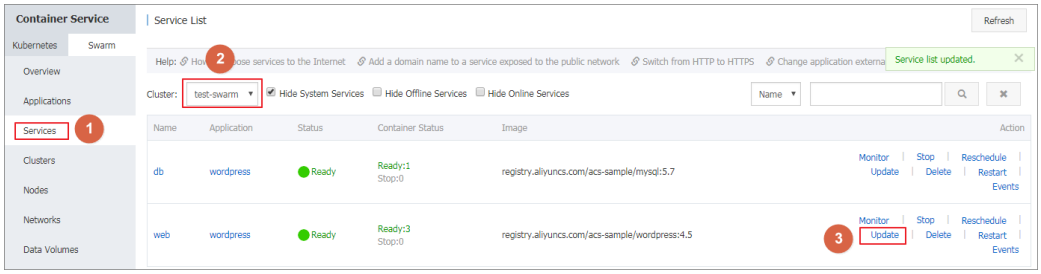


5. In the displayed dialog box, click **OK**.

10.4. Change service configurations

Procedure

1. Log on to the Container Service console.
2. Click Swarm > **Services** in the left-side navigation pane.
3. Select the cluster in which the service you want to change the configurations resides from the Cluster drop-down list.
4. Click **Update** at the right of the service that you want to change the configurations. As shown in the following figure.



5. The Update Service page appears. Change the configurations
6. and then click **Update**.

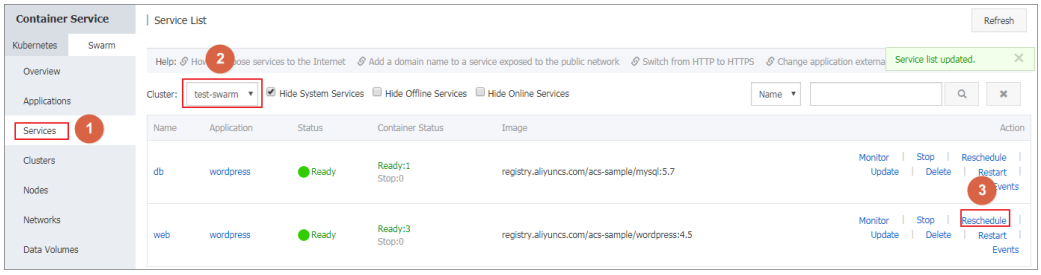
10.5. Reschedule a service

Context

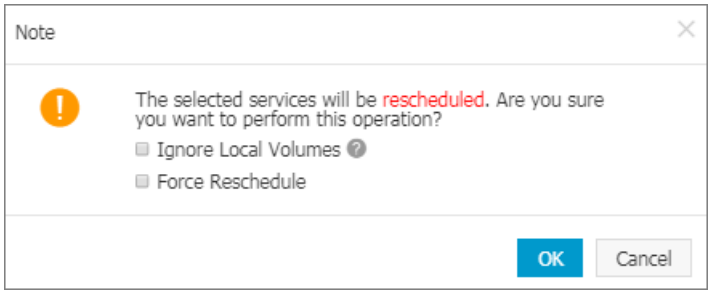
To rebalance the cluster load, you can rebalance the number of containers running on each node by moving the containers from nodes with high loads to the newly added nodes or nodes with low loads.

Procedure

1. Log on to the Container Service console.
2. Click **Services** in the left-side navigation pane.
3. Select the cluster in which the service you want to reschedule resides from the Cluster list.
4. Click **Reschedule** at the right of the service you want to reschedule. As shown in the following figure.



5. In the displayed dialog box, select or clear the **Ignore Local Volumes** and **Force Reschedule** check boxes. Then, click **OK**.



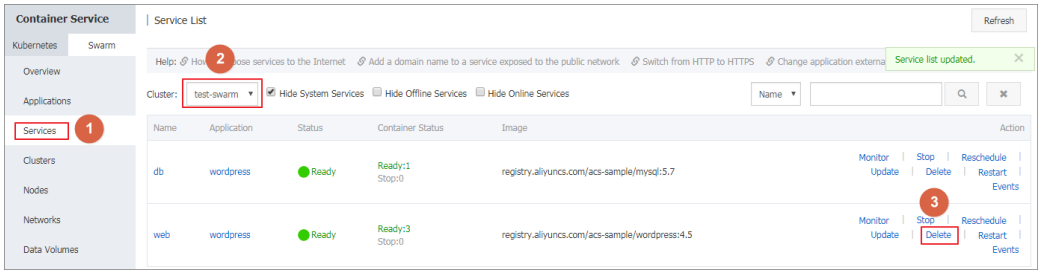
- **Ignore Local Volumes**: For containers with local volumes, rescheduling might move these containers to other machines and cause the data loss. To ignore the local volumes, select this check box. With this check box deselected, the containers with local volumes will not be rescheduled. To ignore the local volumes, select this check box. With this check box deselected, the containers with local volumes will not be rescheduled.
- **Force Reschedule**: Currently, to ensure the stability of online services, rescheduling only occurs when the memory usage and CPU usage of the machine are over 60% To ignore this limit, select this check box. Then, Container Service ignores the usage limit and forces rescheduling the service. and 40% respectively by default. To ignore this limit, select this check box. Then, Container Service ignores the usage limit and forces rescheduling the service.

> **Note** The memory and CPU usage values are subject to the container configurations. Therefore, the values are not always the actual usage of the machine.

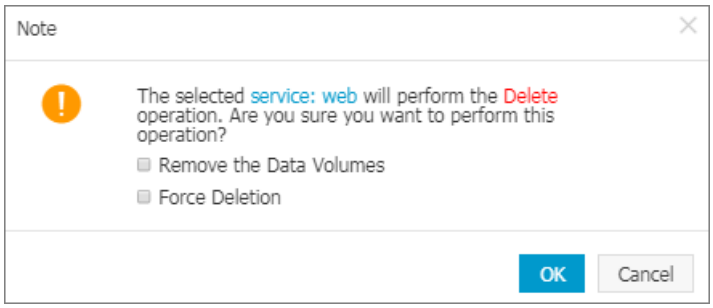
10.6. Delete a service

Procedure

1. Log on to the Container Service console.
2. Click **Services** in the left-side navigation pane.
3. Select the cluster in which the service you want to delete resides from the Cluster list.
4. Click **Delete** at the right of the service that you want to delete.



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



11.Networks

11.1. Container network interconnection

Container Service creates the global network for containers, and the containers in the cluster can access other containers by using the `eth0` network interface of the container.

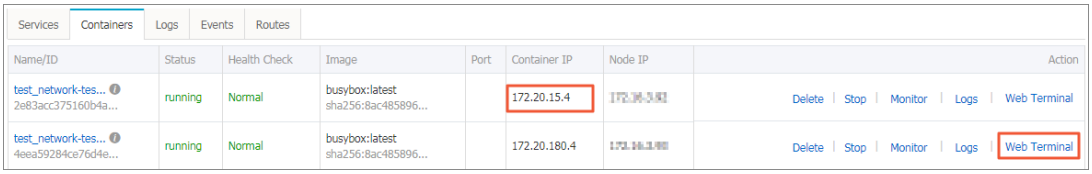
Orchestration example

For example, create containers on two machines respectively and record the IP addresses, names, and hostnames of the containers. Log on to the container web terminal and test the network communication across nodes for these two containers by using the ping command.

```
network-test1:
        image: busybox
        hostname: server1
        command: sh -c 'ifconfig eth0; sleep 100000'
        tty: true
        environment:
            - 'constraint:aliyun.node_index==1'
network-test2:
        image: busybox
        hostname: server2
        command: sh -c 'ifconfig eth0; sleep 100000'
        tty: true
        environment:
            - 'constraint:aliyun.node_index==2'
```

Test methods

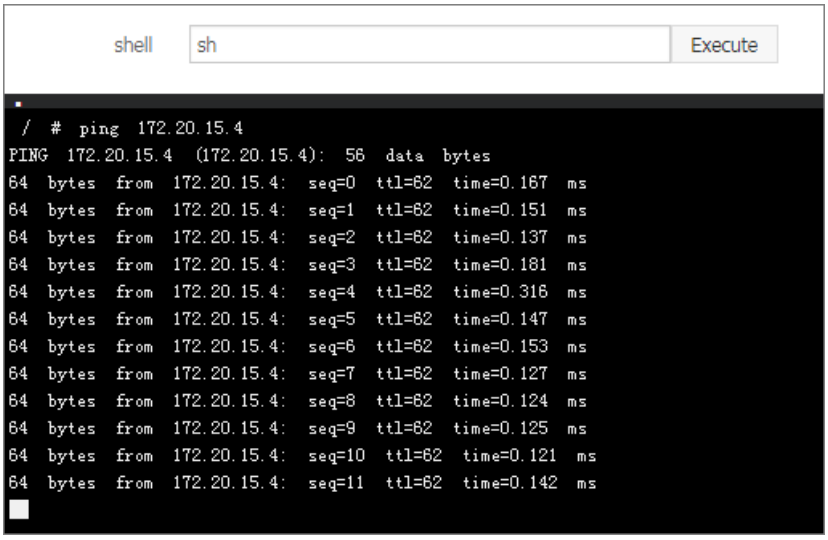
The containers of these two services are distributed in different nodes.



Test network communication by using container IP address

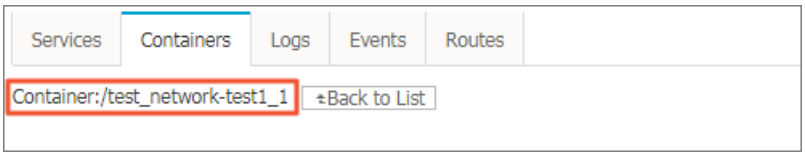
By using the Container Service console or the `ifconfig eth0` log output by container test_network-test1_1, you can see the IP address of container test_network-test1_1 is `172.18.0.4` Then, you can connect to the web terminal to access the IP address of container test_network-test1_1 in container test_network-test2_1 to test whether or not the network communication works.

Enter sh in the shell field and then click Execute. Then, enter the command ping `ping 172.18.0.4` . As shown in the following figure.

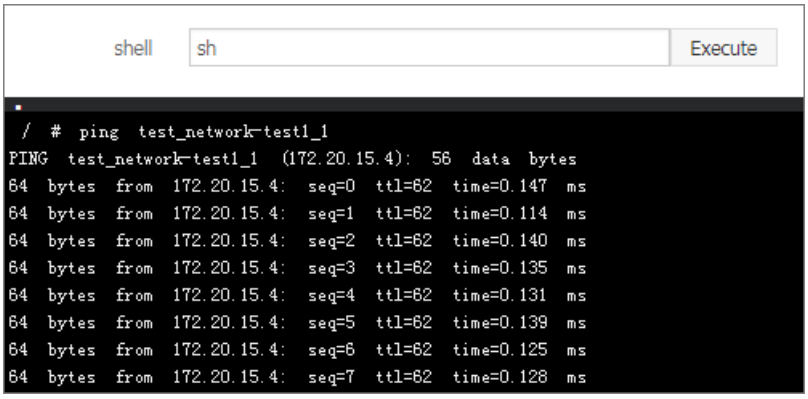


Test network communication by using container name

You can view the container name on the details page of the corresponding container. The default container name is `{project-name}_{service-name}_{container-index}`. In this example, the container name is as follows.

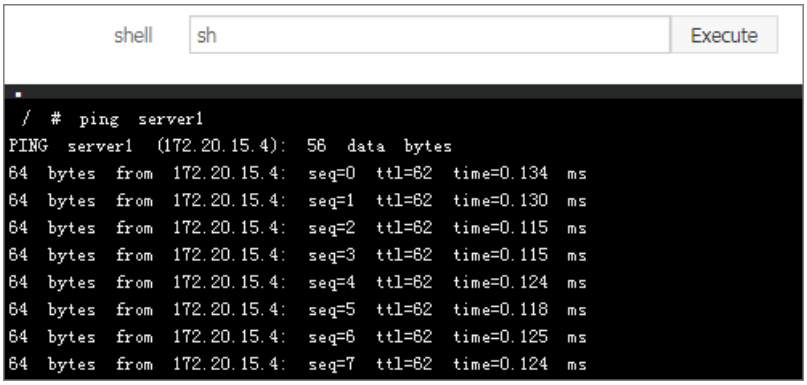


Enter `sh` in the shell field and then click Execute. Then, enter the command `ping test_network_test1_1` to test the network communication by using the container name. As shown in the following figure.



Test network communication by using hostname

In this example, the hostname is specified in the orchestration template. Therefore, you can also test the network communication by using the hostname. As shown in the following figure.



11.2. Use VPC in Container Service

This document introduces how to use Virtual Private Cloud (VPC) in Container Service and the corresponding notes.

VPC CIDR block

- To create a VPC container cluster in Container Service, plan the network according to the actual conditions first. Specify the ClasslessInter-Domain Routing (CIDR) to segment the corresponding subnetworks when creating a VPC.
- Each VPC can only specify one CIDR block. The CIDR block range is as follows. 172.16.0.0/12 is the default CIDR block of VPC.
  - 10.0.0.0/8
  - 172.16.0.0/12
  - 192.168.0.0/16

Container CIDR block

Specify the corresponding container CIDR block when creating a VPC cluster in Container Service. Currently, Container Service supports the following container CIDR blocks:

- 192.168.1.0/24
- 172.[16-31].1.0/24

Network planning

To ensure the network intercommunication between containers, add each container CIDR block to the route table. Therefore, to avoid the conflict of CIDR blocks, perform the corresponding network planning for your application service according to the CIDR blocks of VPC and container.

CIDR block planning

Both VPC and container can have 172 as the CIDR block. Therefore, when selecting 172 as the CIDR block of VPC and container, you must pay attention. For example, if the VPC CIDR block is 172.16.0.0/12 and the VSwitch CIDR block is 172.18.1.0/24. Then, the IP address of the Elastic Compute Service (ECS) instance on the VSwitch is 172.18.1.1-172.18.1.252 according to the CIDR block definition of the VSwitch.

if the VPC CIDR block is 172.16.0.0/12 and the VSwitch CIDR block is 172.18.1.0/24. Then, the IP address of the Elastic Compute Service (ECS) instance on the VSwitch Then, the IP address of the Elastic Compute Service (ECS) instance on the VSwitch is 172.18.1.1-172.18.1.252 according to the CIDR block definition of the VSwitch. If the container CIDR block is also 172.18.1.0/24, the IP address of the container and that of the ECS instance are the same. In this way, an exception occurs in the network communication between containers. In this way, an exception occurs in the network communication between containers. So you must pay attention to the network planning when using a VPC.

Route table planning

Currently, up to 48 route entries can be included in a route table of a VPC.

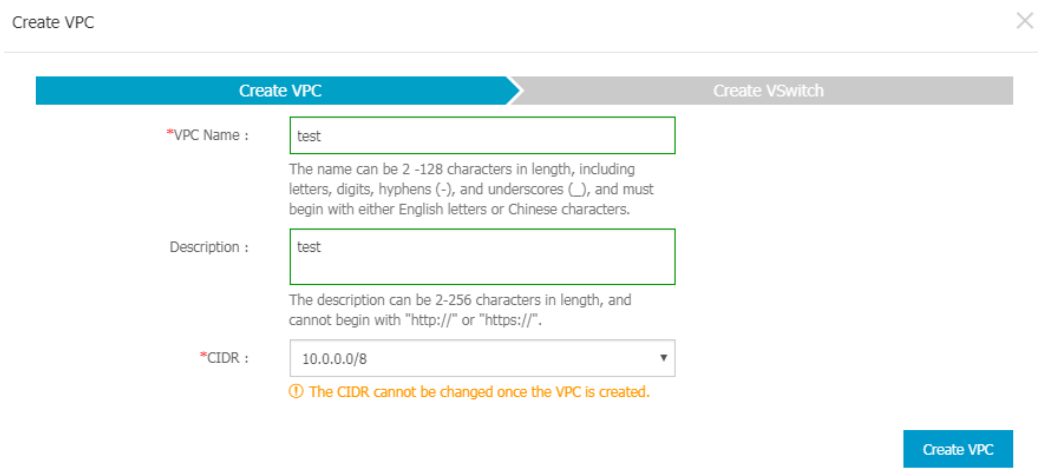
Example

Use the following complete example to demonstrate the whole creation process of a VPC cluster in Container Service.

Step 1. Create a VPC

1. Log on to the VPC console.
2. Click **VPC** in the left-side navigation pane.
3. Select the region. In this example, select China East 1 (Hangzhou).
4. Click Create VPC. The Create VPC dialog box opens. Complete the configurations and then click Create VPC.

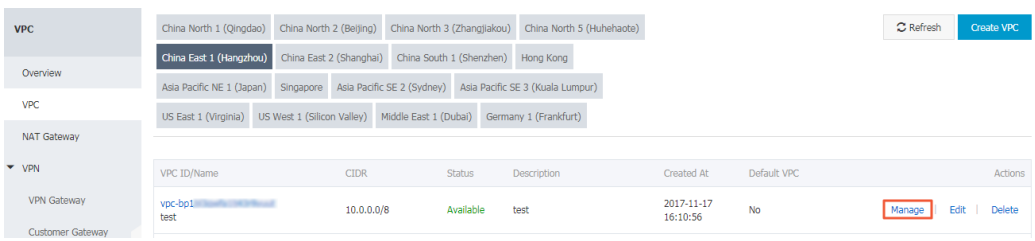
   In this example, to avoid conflicting with the container CIDR block, select 10.0.0.0/8 as the CIDR. as the CIDR.



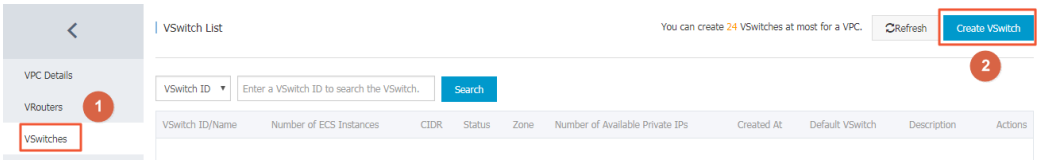
Step 2. Create a VSwitch

After creating a VPC, create the corresponding VSwitch under this VPC.

1. On the VPC list,

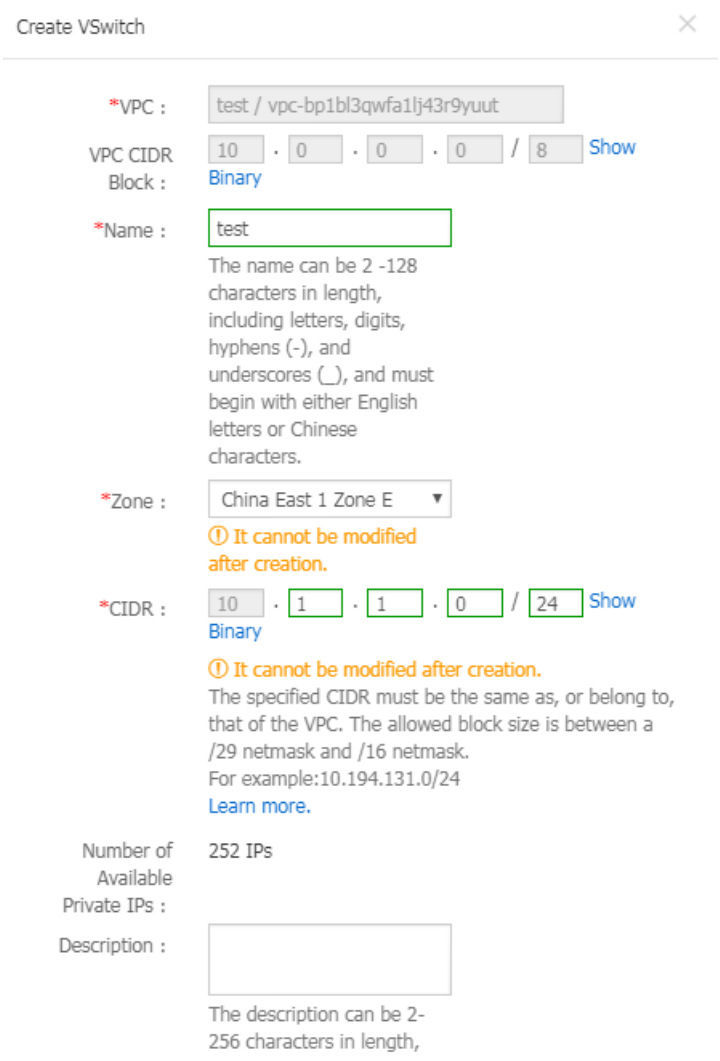


2. click **Manage** at the right of the created VPC.
3. Click **VSwitches** in the left-side navigation pane, and then click **Create VSwitch** in the upper-right corner.



4. The Create VSwitch dialog box opens. Complete the configurations and then click **OK**.

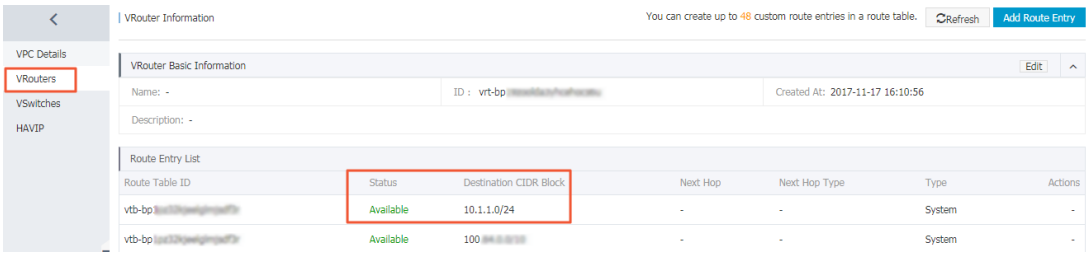
   In this example, select a China East 1 zone and enter 10.1.1.0/24 as the CIDR. If you select this VSwitch when the IP address of the ECS instance will be 10.1.1.1-10.1.1.252. The number of available private IPs is 252 in total, which means you can purchase 252 ECS instances under the VSwitch of this CIDR block. creating an ECS instance, when the IP address of the ECS instance will be 10.1.1.1-10.1.1.252. The number of available private IPs is 252 in total, which means you can purchase 252 ECS instances under the VSwitch of this CIDR block.



Step 3. View route table

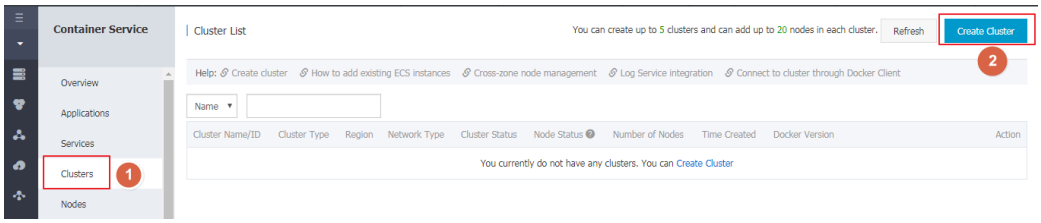
After creating the VPC and VSwitch, you can view the route table.

Click **VRouters** in the left-side navigation pane. The network of VSwitch is the default system route table.



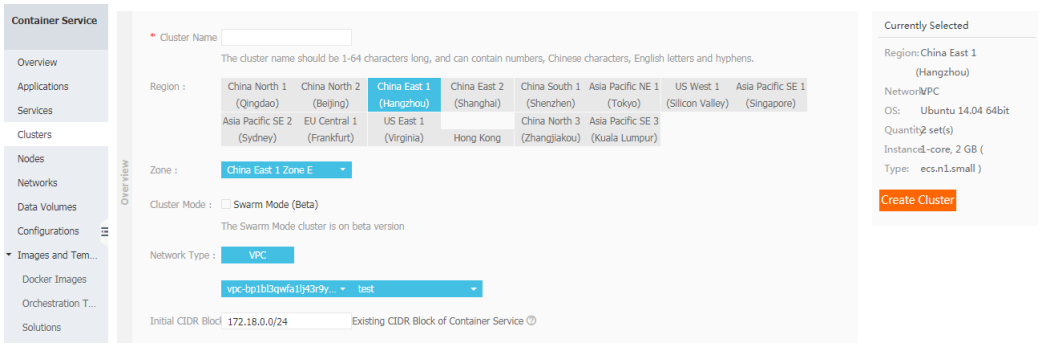
Step 4. Create a container cluster

1. Log on to the Container Service console.
2. Click **Clusters** in the left-side navigation pane,
3. and then click **Create Cluster** in the upper-right corner.



4. Complete the configurations and then click **Create Cluster**.

   Select China East 1 (Hangzhou) as the **Region**, and VPC as the **Network Type**. Select the created VPC and VSwitch from the corresponding list.



In this example, the **Initial CIDR Block of Container Service** is 172.18.0.0/24. Then, the container CIDR block on the nodes of this cluster is 0/24. The container IP address on each node is 172.18.x.[1-255]. 172.18.[1-254]. 0/24. The container IP address on each node is 172.18.x.[1-255].

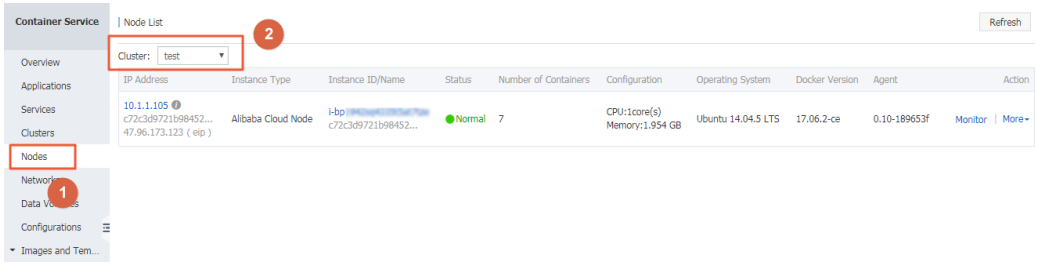
Step 5. Verify node IP address

After creating the container cluster, you can verify the preceding theory of network planning by verifying the cluster node IP address, route table, and checking the application container IP address.

You can verify the VSwitch CIDR block by checking the ECS instance IP address in the container cluster node list.

1. Log on to the Container Service console.
2. Click **Nodes** in the left-side navigation pane.
3. Select the cluster where the node that you want to verify resides from the Cluster list.

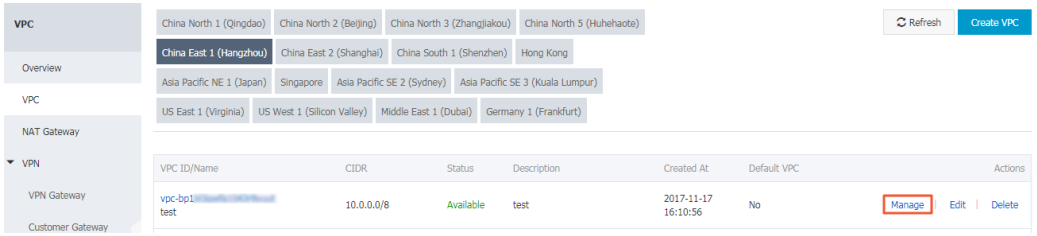
From the following figure, we can see that the IP address of the newly purchased ECS instance belongs to the VSwitch CIDR block 10.1.1.0/24.



Step 6. Verify route table

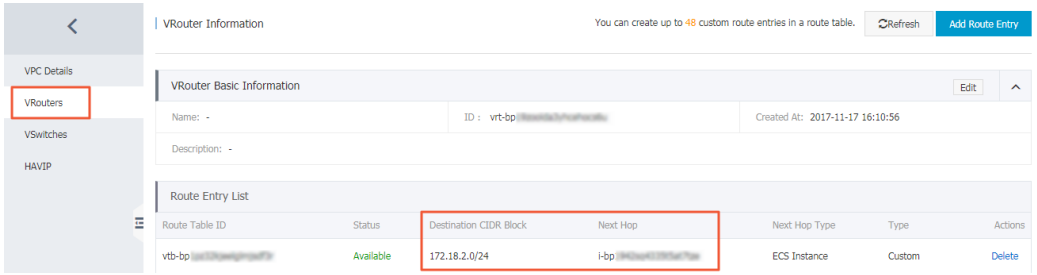
Verify the route table after verifying the node IP address.

1. Log on to the VPC console.
2. Click **VPC** in the left-side navigation pane.
3. Click
4. **Manage** at the right of the created VPC.



5. Click **VRouters** in the left-side navigation pane.

   A route entry with 172.18.x.0/24 0/24 as the CIDR block is added to the route table. The next hop is the corresponding ECS instance ID.



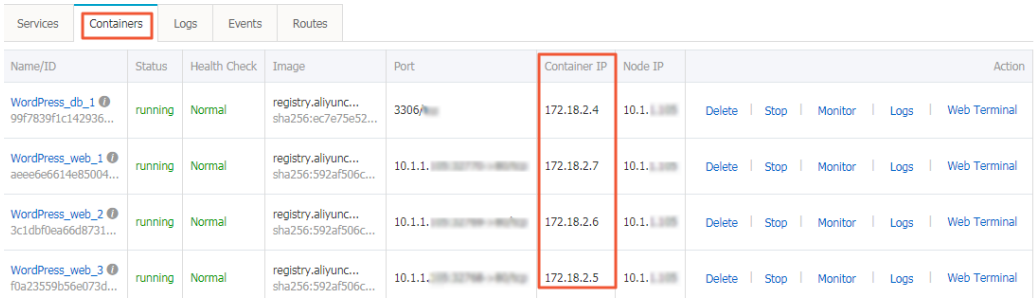
Step 7. Verify container IP address

Finally, verify whether the container IP address is correct or not.

In this example, deploy a WordPress application by using an orchestration template in Container Service console, and then verify the container IP address by checking the container list on a node.

For how to create the WordPress application, see Create an application by using an orchestration template.

1. Log on to the Container Service console.
2. Click **Applications** in the left-side navigation pane.
3. Select the cluster where the created application resides from the Cluster list.
4. Click the application name. Click the **Containers** tab.



The preceding verification shows that a VPC container cluster is successfully created.

12.Data volumes

12.1. Overview

The characteristic of Docker determines the containers are non-persistent. Deleting a container also deletes its data. Data volumes provided by Docker can realize persistent storage by attaching to the host directories, but the data volumes in the host have the following limits in the cluster environment:

- Data cannot be migrated when containers are migrated between machines.
- Different machines cannot share data volumes.

To solve these issues, Alibaba Cloud Container Service provides third-party data volumes. By packaging various cloud storage resources as data volumes, these data volumes can be attached to containers directly and automatically reattached when containers are restarted or migrated. Currently, cloud disks and OSSFS are supported.

12.2. Create an OSSFS data volume

OSSFS is a FUSE-based file system provided by Alibaba Cloud (click https://github.com/aliyun/ossfs to view the project homepage). OSSFS data volumes can package Object Storage Service (OSS) buckets as data volumes.

The performance and functions of OSSFS differ from those of local file systems because data must be synchronized to the cloud by the means of network. Do not run databases, I/O-intensive applications, logs and other applications that require constantly writing files to OSSFS. OSSFS is suitable for sharing configuration files across containers, uploading attachments, and other scenarios without rewrite operations.

OSSFS differs from local file systems in the following ways:

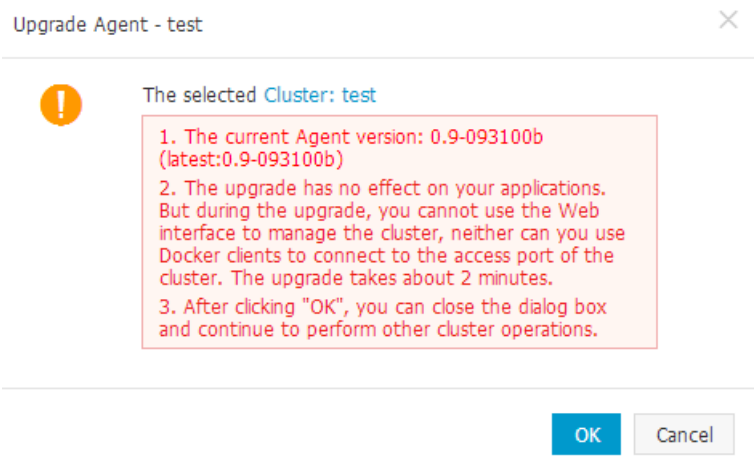
- Random write or append write leads to the entire file being overwritten.
- Metadata operations, such as list directory, provide poor performance because the system must remotely access the OSS server.
- The file/folder rename operation is not atomic.
- Coordinate the actions of each client on your own when multiple clients are mounted to the same OSS bucket. For example, avoid multiple clients from writing the same file.
- Hard link is not supported.

Prerequisites

You can only use the data volume function when your cluster meets the following conditions:

- The cluster Agent is of version 0.6 or later.

  You can check your Agent version on the Cluster List page. Select the target cluster, and click **More > Upgrade Agent** on the right.



If your Agent version is earlier than 0.6, upgrade the Agent first. For how to upgrade Agent, see Upgrade Agent.

- Deploy the acsvolumedriver application in the cluster. We recommend that you upgrade the acsvolumedriver application to the latest version.

  You can deploy and upgrade the acsvolumedriver application by upgrading the system services. For more information, see Upgrade system services.

  > **Note** When acsvolumedriver is upgraded or restarted, containers that use OSSFS data volumes are restarted, and your services are also restarted.

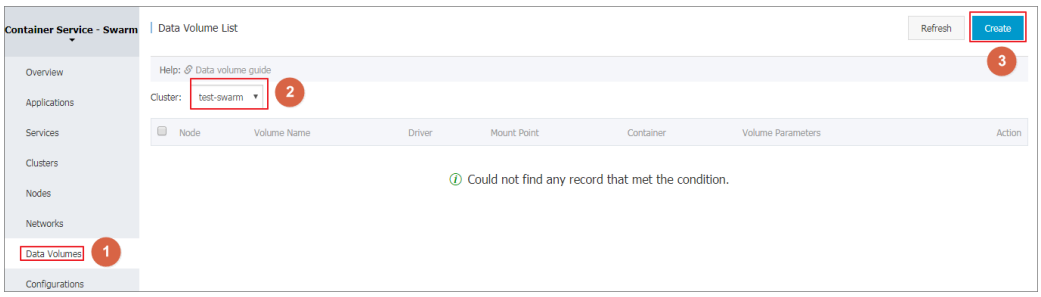
Step 1. Create an OSS bucket

1. Log on to the OSS console.
2. and create a bucket.

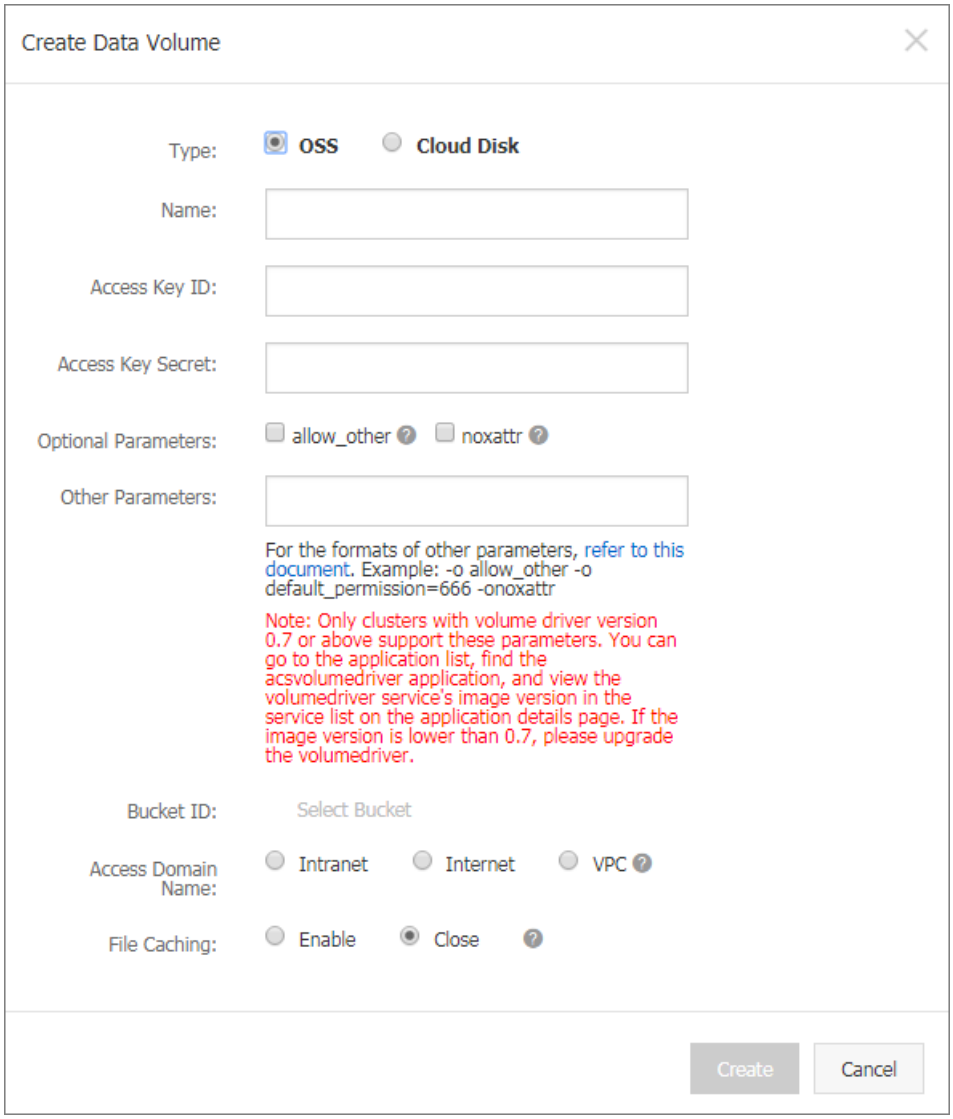
   In this example, create a bucket in the region of China East 1 (Hangzhou).

Step 2. Create an OSSFS data volume

1. Log on to the Container Service console.
2. Click **Data Volumes** in the left-side navigation pane.
3. Select the cluster in which you want to create a data volume from the Cluster list. Click **Create** in the upper-right corner.



4. The **Create Data Volume** dialog box appears. Select **OSS** as the Type, configure the data volume parameters, and then click **Create**. Container Service creates a data volume with the same name on each cluster node.



- **Name**: The name of the data volume that must be unique within the cluster.
- **Access Key ID/Access Key Secret**: The AccessKey required to access OSS. You can obtain them from the AccessKey console.
- **Bucket ID**: The name of the OSS bucket to be used. Click **Select Bucket** in the dialog box, and click **Select**.
- **Access domain name**: If the bucket and ECS instances are in different regions, select **external domain name**. If they are located in the same region, you must select the corresponding cluster network type. For VPC network, select **VPC domain name**, and for classic network, select **intranet domain name** respectively.
- **File Caching**: Select Disable if you want to synchronize the modifications of the same file on multiple machines (for example, modify the file on machine A and read the modified contents on machine B).

> **Note** Turning off the File Caching causes ls folder to become slow, especially when a lot of files exist in the same folder. Therefore, when there is no such requirement, enable the File Caching and increase the speed of the ls command.

You can view the created OSSFS data volumes on the Data Volume List page.

Subsequent operations

After the data volumes are created, you can use the data volumes created in your app. For more information about how to use data volumes in an application, see Use third-party data volumes.

12.3. Creating NAS data volumes

Alibaba Cloud NAS is a file storage service for Alibaba Cloud ECs instance, providing a standard file access protocol, you do not need to make any changes to an existing application, ready to use with unlimited capacity and performance expansion, single namespace, multi-share, high reliability and high availability features such as distributed file systems.

Restrictions on Use

At present, Ali cloud NAS is open to North China 1, North China 2, North China 3, North China 5, East China 1, east China 2, South China 1, Asia Pacific Southeast 1, only clusters located in these areas can create NAS data volumes.

Prerequisites

You can only use the data volume function when your cluster meets the following conditions:

- The cluster Agent is of version 0.6 or later versions.

  You can view your Agent version on the Cluster List page. Select the desired cluster and click more upgrade agents on the right.

  If your version of the agent is too low, upgrade your agent first. For more information about how to upgrade the agent, see.

- Deploy the acsvolumedriver application in the cluster. We recommend that you upgrade the acsvolumedriver application to the latest version.

  You can deploy and upgrade the acsvolumedriver application by upgrading the system services. See for details.

Procedure

This example shows an example of a VPC Container service cluster located in the East China 1 region.

Step 1 create a NAS File System

1. Log in to the file store management console.
2. Create a NAS file system.

   > **Note** The NAS file system you created needs to be located in the same area as your cluster.

   - Geography: select the same region as the container cluster. This example selects East China 1.
   - Storage Type: this example selects the capacity type.
   - Protocol type: Select NFS.

- Availability area: Select East China 1 Availability Zone B. Different available areas of the same region can be interfaced.
- Click **OK**.

3. After you click OK. This example creates a NAS file system located in the East China 1 region.

Step 2 Add a mount point for the container service cluster

1. Log in to the file store management console.
2. Click the list of file systems in the left-hand navigation bar to select the file systems that are created in step 1, click to the right to add a mount point.
3. Configure in the pop-up add mount point dialog box.

4. This example adds a VPC mount point.

> **Note** VPC network select where your container cluster is located Otherwise, an error occurs when you create a data volume.

Step 3 Add the cluster ECs instance network IP to the NAS File System whitelike list

In order for the ECS instance in the cluster to have access to the NAS file system, the internal network IP of the ECS instance needs to be added to the White List of NAS file systems.

- For clusters created after February 2017, when you create a NAS data volume, the internal network IP of the ECS instance in the cluster is automatically added to the White List of NAS file systems, you do not need to do anything.

  After creating the NAS data volumes, When you expand the cluster (via, or, the container service automatically creates a NAS data volume for the newly added or expanded ECs instance and automatically the newly added nets IP of the ECS instance is added to the White List of NAS file systems.

- For clusters created before February 2017, you can add the internal network IP of the ECS instance in the cluster to the NAS File System in two ways.
  - Add White List manually.

    Log in to the file storage management console, create a permission group and add a permission group rule to add the inner network IP of the cluster ECs instance to the White List. See NAS use documentation for details and use permission groups for access control.

    Cluster Expansion after you add a whiteable list and create a NAS data volume through this method (, or, the container service automatically creates NAS data volumes for newly added or expanded ECs instances, but before using these volumes, you must manually and only add the internal network IP of the newly added ECs instance to the NAS file system. A white list.

- Authorization in Ram. The white list is automatically added after authorization, and cluster capacity will be expanded later (via, or) the newly added nets IP of the ECS instance is automatically added to the White List of NAS file systems.
  a. Log in to the RAM Management Console.
  b. Click User administration in the left-hand navigation bar.
  c. Locate the user group named porter_[cluster_id] and click.

  d. Click user authorization policies in the left-hand navigation bar, and click Edit authorization policies in the upper-right corner.

  e. In the search box, enter Nas for the search, select, add to the list of selected Authorization Policy names and click OK.

  f. Click OK to complete the authorization based on the page boot.

Step 4 Create NAS data volumes

1. Log on to the Container Service console.
2. Click **Data Volumes** in the left-side navigation pane.
3. Select the cluster where you want to create the data volume and click Create in the upper-right corner of the page.

4. In the pop-up dialog box, set the data volume parameters and click Create. The Container Service creates NAS data volumes with the same name on all nodes in the cluster.

   You can log in to the file store management console and click the ID of the NAS file system to be mounted by the cluster, view the details of the file system.

   See the file system for details to complete the configuration of the data volumes.

   - Data Volume name: the name of the data volume. The data volume name must be unique within the cluster.
   - File System ID: ID of the NAS file system.
   - Accesskey ID and accesskey secret: The accesskey for your account.

     > **Note** The container service began supporting STS token functionality in December 5, 2017. If your cluster was created after this date, when you create a NAS data volume in this cluster, you enter accesskey.

   - **Mount Point Domain Name**: Enter the mount address of the mount point in the NAS file system for the cluster.
   - Sub-Directory: the sub-directory under the NAS path, which starts, once set, the data volume is mounted to the specified sub-directory.
     - If this sub-directory does not exist in the NAS root directory, the data volume is mounted after the sub-directory is created by default.

- If this field is left empty, the data volume is mounted to the NAS root directory by default.

- **Privilege**: Configure the access permission of the mount directory, such as 755, 644, and 777.
  - You can only configure the privilege when the data volume is mounted to the NAS sub-directory, that is, you cannot configure the privilege if the data volume is mounted to the NAS root directory.
  - If this field is left empty, use the permissions of the NAS files by default.

> **Note** Upgrade the volume driver to the latest version when using the sub-directory, permissions option.

Subsequent operations

After the data volumes are created, you can use the data volumes created in your app. For more information about how to use data volumes in an application, see.

12.4. Create cloud disk data volumes

Cloud disk is a block storage system officially provided by Alibaba Cloud, and an elastic block storage product of distributed storage architecture that Alibaba Cloud provides to Elastic Compute Service (ECS). Cloud disk provides random storage of data block level, features in low latency, persistence, and high reliability, and adopts the distributed mechanism of three copies.

Cloud disk can be used for relational database applications or development and test applications. For more information, see Disks.

Limits

- The cloud disk and the ECS instances in the cluster must be in the same region and zone.
- Cloud disk data volumes only support being mounted to a single machine, but does not support the shared mode.
- A cloud disk data volume can be used by only one container at the same time.

Prerequisites

- Create a cloud disk manually in the ECS console before using the cloud disk data volume.
- Upgrade your Agent to the latest version. For more information, see Upgrade Agent.
- Deploy the acsvolumedriver application in the cluster. We recommend that you upgrade the acsvolumedriver application to the latest version.

  You can deploy and upgrade the acsvolumedriver application by upgrading the system services. For more information, see Upgrade system services.

Procedure

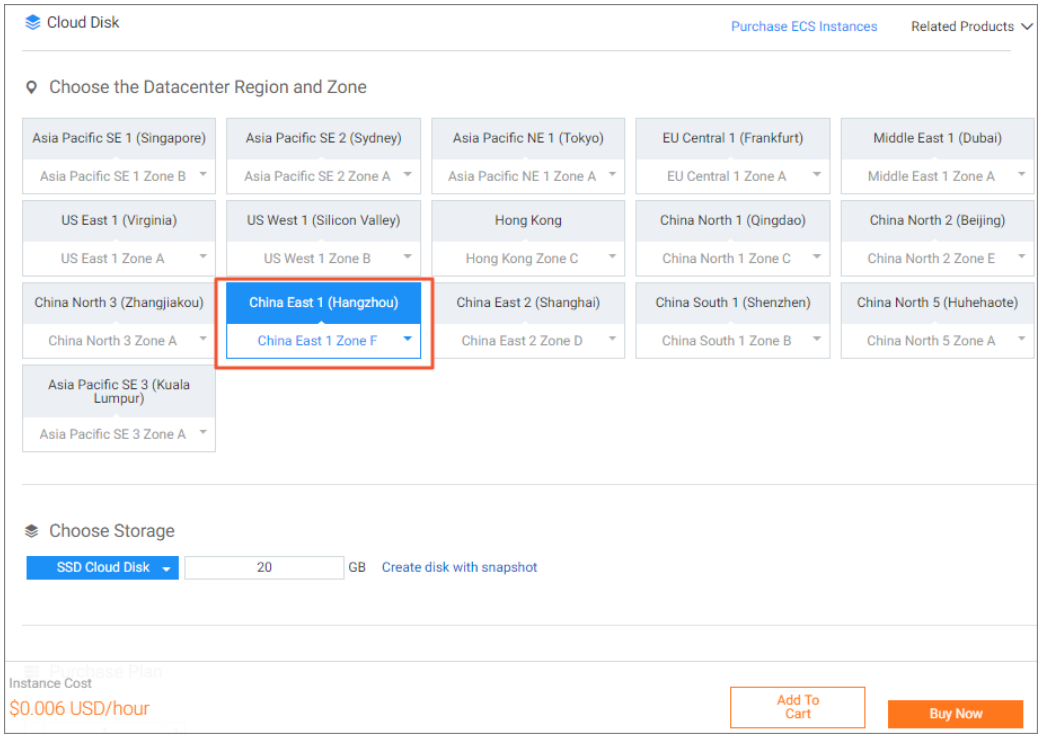
Step 1 Create a cloud disk

In this example, create a cloud disk that is in the same region and zone as the cluster.

1. Log on to the ECS console.
2. Click **Cloud Disks** in the left-side navigation pane.
3. On the Disk List page, click **Create Cloud Disk** in the upper-right corner.

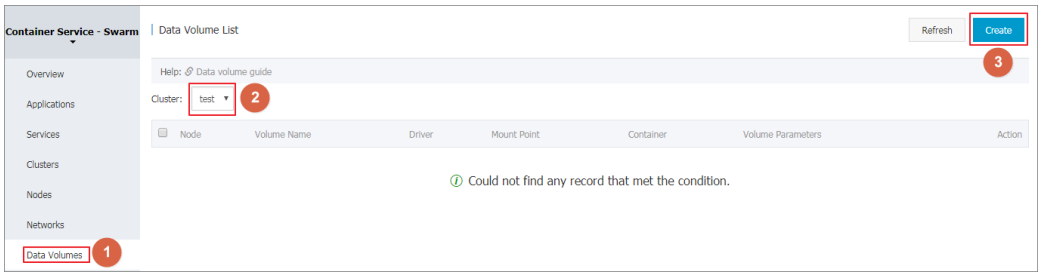
4. Configure the parameters for the cloud disk. Select the corresponding region and zone. Create the cloud disk according to the guidance on the page.

> **Note** The purchased cloud disk can be mounted only when you select the same zone as the server. The cloud disk cannot be mounted across zones or regions.

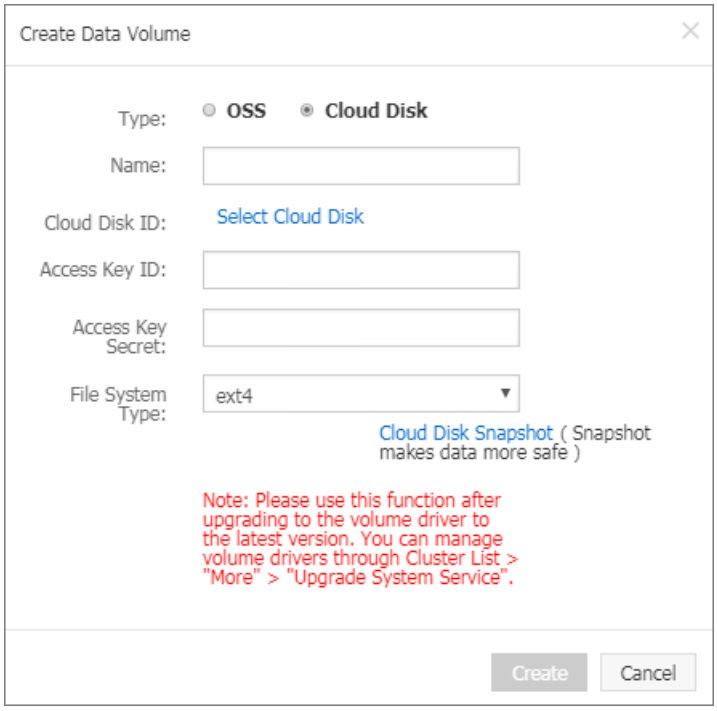


Step 2 Create data volumes by using the cloud disk

1. Log on to the Container Service console.
2. Click **Data Volumes** in the left-side navigation pane.
3. Select the cluster in which you want to create the data volume from the Cluster list and then click **Create** in the upper-right corner.



4. In the displayed dialog box, select **Cloud Disk** as the **Type**, configure the data volume parameters and then click **Create**. Container Service will create a data volume with the same name on each cluster node.

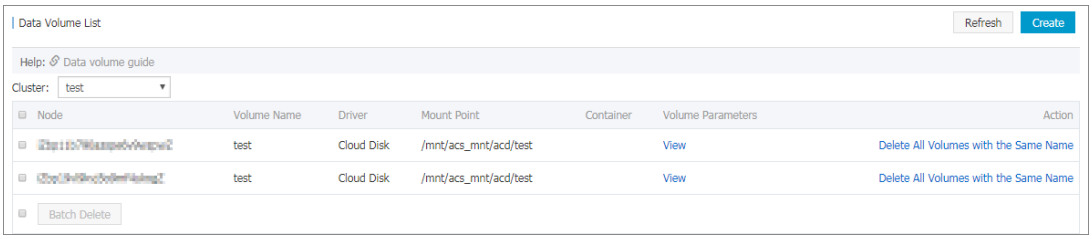


- **Name**: The name of the data volume, The data volume name must be unique within the cluster.
- **Cloud Disk ID**: Select the cloud disk to be mounted and is in the same region and zone as the cluster. In this example, select the ID of the cloud disk created in step 1.
- **AccessKey ID** and **AccessKey Secret**: The AccessKey of your account.

  > **Note** Container Service begins to support the STS Token function since December 5, 2017. If your cluster is created after that, you must enter the AccessKey when you create a cloud disk data volume in the cluster.

- File System Type: You can select the data type in which data is stored to the cloud disk. The supported types include ext4, ext3, xfs, and vfat.

After the data volume is successfully created, you can view the cloud disk data volume on the Data Volume List page.



Subsequent operations

You can manage the cloud disk data volumes, including deleting all the data volumes with the same name and viewing data volume parameters.

12.5. View and delete data volumes

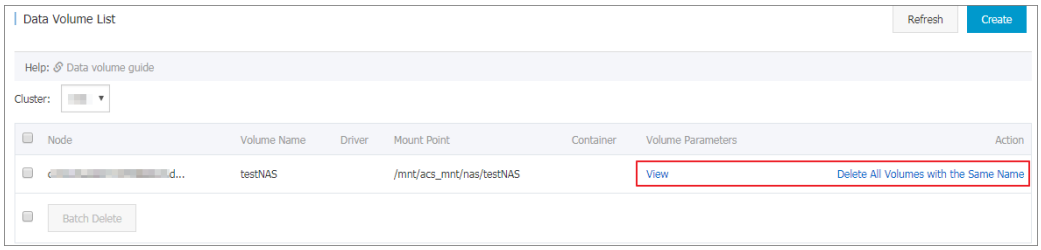
You can view and delete the created data volumes.

Procedure

1. Log on to the Container Service console.
2. Click **Data Volumes** in the left-side navigation pane and select the target cluster.

   All the data volumes in the selected cluster are displayed on the **Data Volume List** page, including the local data volumes and third-party data volumes.

   On this page, you can view the containers that reference the data volumes.



   For local data volumes, the data volume name is in the format of `node_name/volume_name`.

   For third-party data volumes, you can click **View** under Volume Parameters to view the parameters of the data volumes.

   When you create a third-party data volume, Container Service creates the data volume with the same name on each node in the cluster, allowing containers to be migrated between nodes. You can also select to **Delete all volumes with the same name**.

   > **Note** Data volumes referenced by containers cannot be deleted. The Data Volume List page displays the containers that reference the data volume. You must delete the containers that references the data volume before you can delete the data volume.

12.6. Use third-party data volumes

Third-party data volumes are used in the same way as local data volumes.

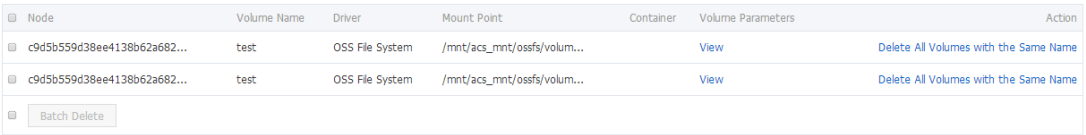
You can set the data volumes when creating an application or changing the configurations of an existing application.

Prerequisite

You have created a data volume in Container Service console. For details, see Create an OSSFS data volume .

Procedure

Take the OSSFS data volume test in the test cluster as an example.

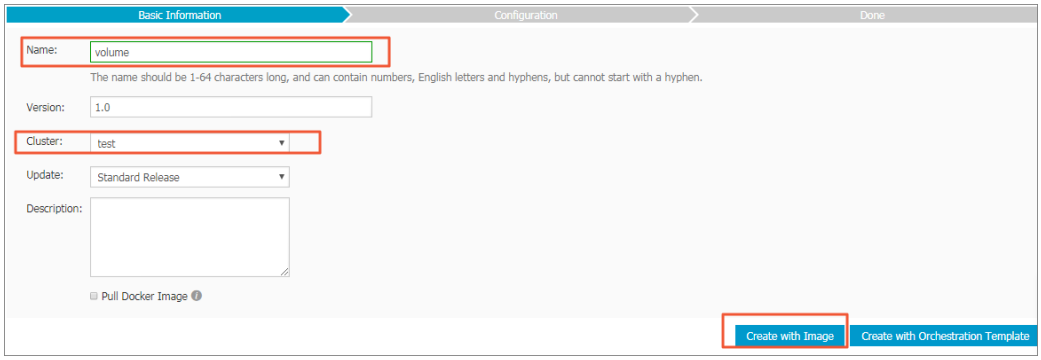


Create an application by using an image

1. Log on to the Container Service console.

2. Click **Applications** in the left-side navigation pane.
3. Click **Create Application** in the upper-right corner.
4. Enter the basic information for the application you want to create and then click **Create with Image**. In this example, enter volume as the Name and select test as the Cluster.

> **Note** The cluster on which the application will be deployed must be the same as the one of the OSSFS data volume that you want to use.



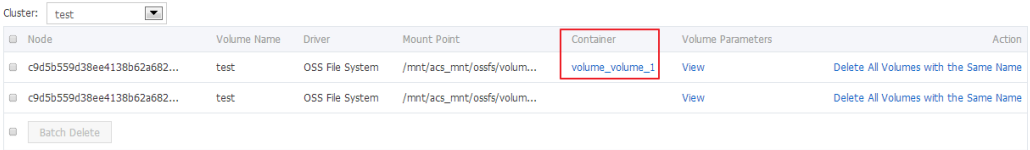
5. Select the image you want to use and complete the other configurations.

> **Note** For how to create an application by using an image, see Create an application.

6. Click the plus icon in the **Volume** section. Enter the data volume name in the **Host Path** or Data Volume Name field. Enter the Container Path and select RW or RO as the data volume permission.

7. Click **Create** at the right of the page after completing the settings.

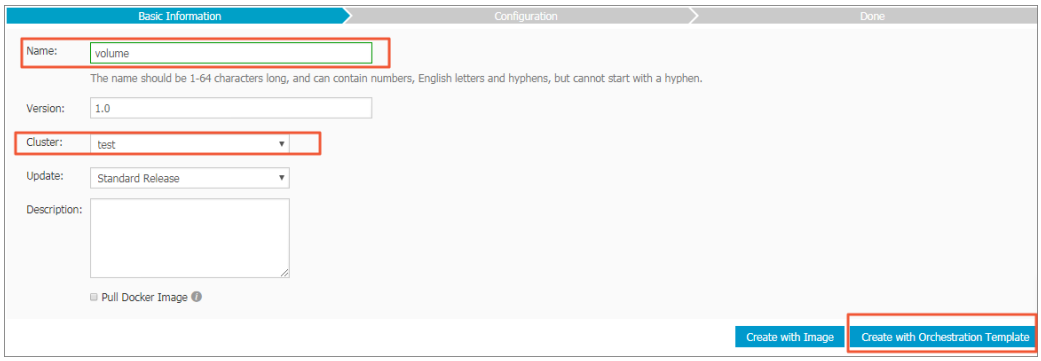
   On the Data Volume List page, you can see that the OSSFS data volume test is referenced by the container of the volume application.



Create an application by using an orchestration template

1. Log on to the Container Service console.
2. Click **Applications** in the left-side navigation pane.
3. Click **Create Application** in the upper-right corner.
4. Enter the basic information for the application you want to create and then click **Create with Orchestration Template**. In this example, enter volume as the Name and select test as the Cluster.

> **Note** The cluster on which the application will be deployed must be the same as the one of the OSSFS data volume that you want to use.



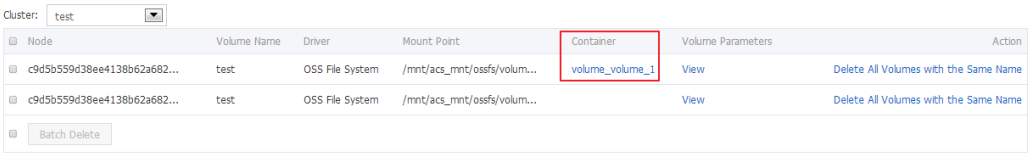
5. Click **Use Existing Orchestration Template** or use your own orchestration template.

> **Note** For how to create an application by using an orchestration template, see Create an application.

6. In the `volumes` section of the template, enter the data volume name, container path, and permission.

7. Click **Create and Deploy** after completing the settings.

   On the Data Volume List page, you can see that the OSSFS data volume test is referenced by the container of the volume application.



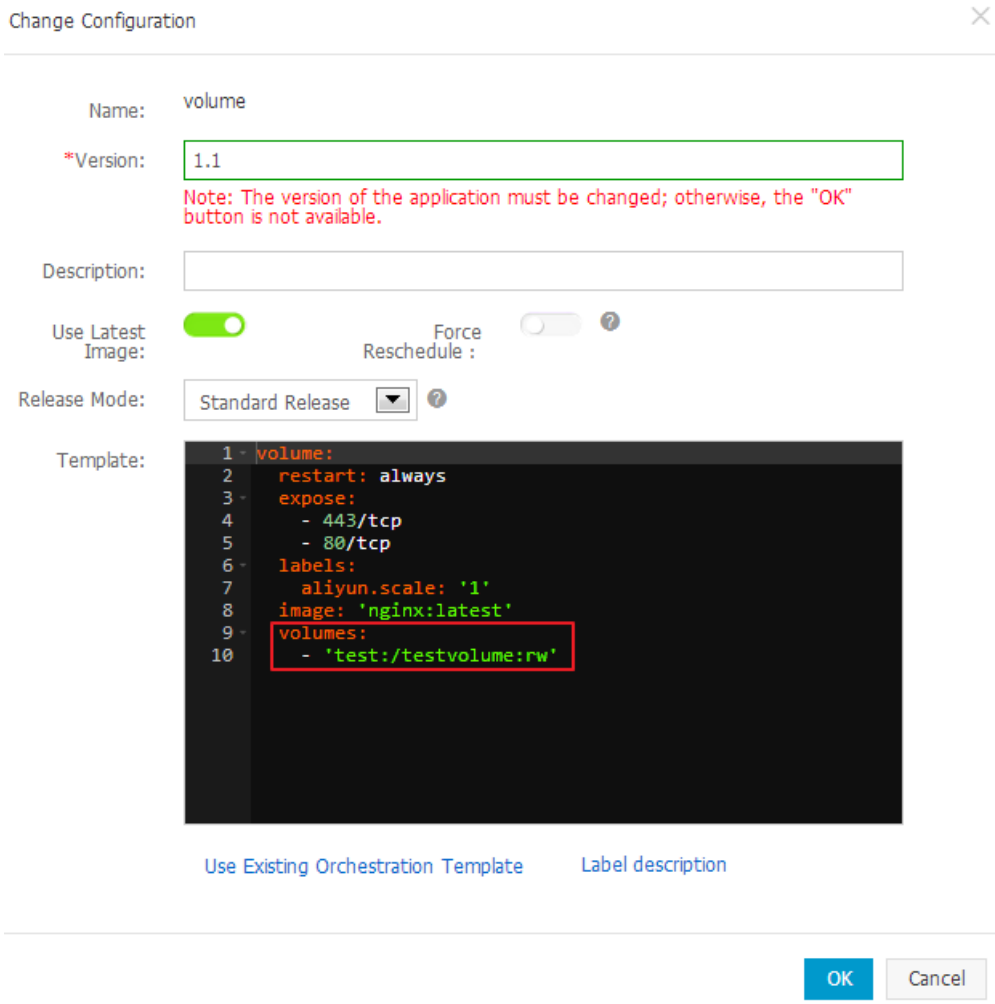
Change the configurations of an existing application

1. Log on to the Container Service console.
2. Click **Applications** in the left-side navigation pane.
3. Select the cluster (the test cluster in this example) in which the application resides from the Cluster list. Click **Update** next to the application you want to change the configurations.

   For how to change the application configurations, see Change application configurations.

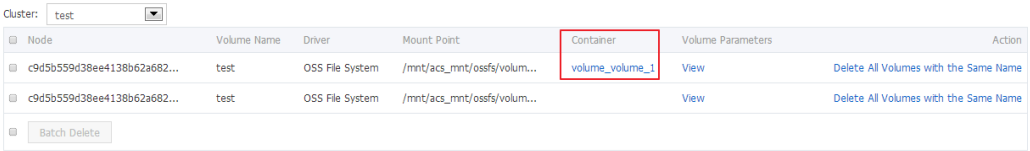
> **Note** Make sure the application and the OSSFS data volume you want to use are in the same cluster.

4. The Change Configuration dialog box appears. In the `volumes` section of the template, enter the data volume name, container path, and permission.



5. Click **OK** after completing the modifications.

   On the Data Volume List page, you can see that the OSSFS data volume test is referenced by the container of the volume application.



12.7. FAQ

The container fails to be launched and the system reports an error such as chown /mnt/acs_mnt/ossfs/XXXX: input/output error if you use the third-party data volume in the method of Data volume name: an existing directory in the image (for example, `o1:/data`, when the /data directory exists in the image).

This error occurs because for named data volumes, Docker copies the existing files in the image to the data volumes and uses `chown` to set the relevant user permissions. However, Linux prohibits the use of chown for mount points.

To solve this issue, you can use one of the following solutions:

- Upgrade Docker to version 1.11 or later versions. Upgrade Agent to the latest version and specify `nocopy` in the orchestration template. Docker will not copy the data and thereby, no `chown` error will occur.

```
volumes:
    - o1:/data:nocopy
    - /tmp:/bbb
```

- If you need to copy the data, use the mount point path instead of the data volume name to set the data volume. For example, */mnt/acs_mnt/ossfs/XXXX:/data*. However, this method bypasses the volume driver. When the machine is restarted, the container might be started before the OSSFS is successfully mounted and the container might be attached to a local data volume. To avoid this issue, use two data volumes at the same time. One is set by the data volume name and the other is set by the mount point path. The data volume set by the data volume name is only used for synchronizing with the volume driver and is not used for storage.

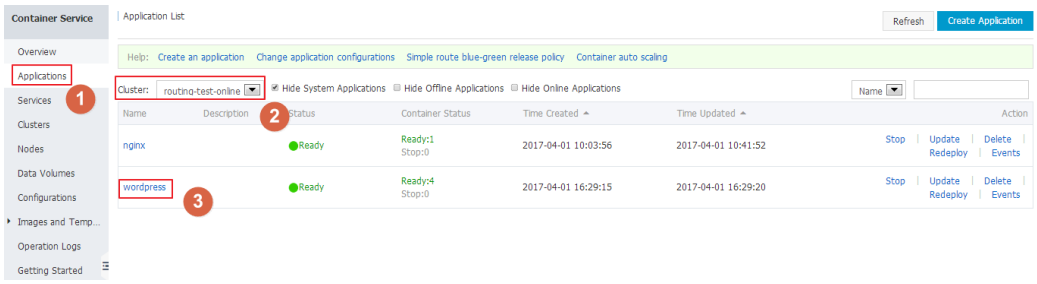
```
volumes:
    - o1:/nouse
    - /mnt/acs_mnt/ossfs/XXXX:/data
    - /tmp:/bbb
```

13.Logs

13.1. View logs

View application logs

1. Log on to the Container Service console.
2. Click Swarm > **Applications** in the left-side navigation pane.
3. Select the cluster in which the application you want to view the logs resides from the Cluster drop-down list.
4. Click the name of the application that you want to view the logs. As shown in the following figure.

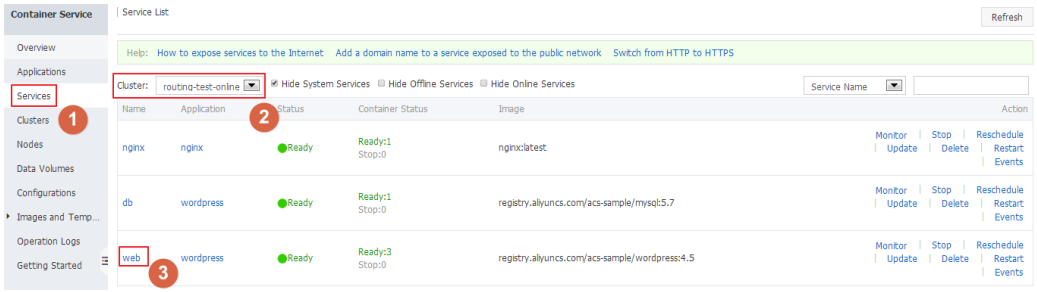


5. Click the **Logs** tab to view the application logs.

   You can select how many log entries are displayed for each container and download all the logs to your local device.

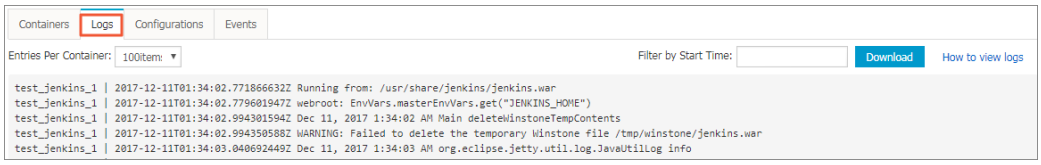
View service logs

1. Log on to the Container Service console.
2. Click Swarm > **Services** in the left-side navigation pane.
3. Select the cluster in which the service you want to view the logs resides from the Cluster drop-down list.
4. Click the name of the service that you want to view the logs. As shown in the following figure.



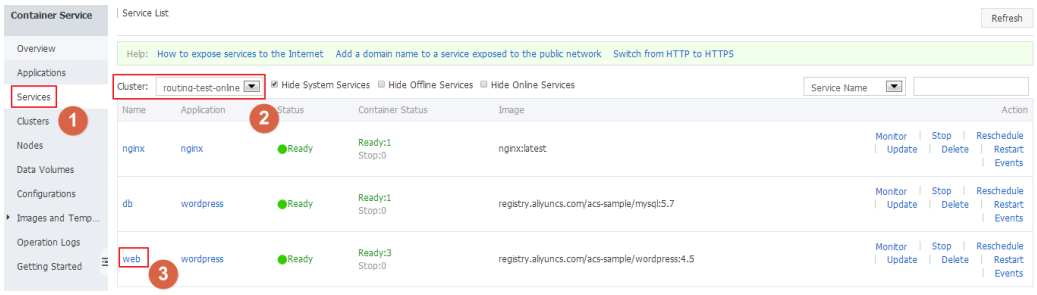
5. Click the **Logs** tab to view the service logs.

   You can select how many log entries are displayed for each container and download all the logs to your local device.

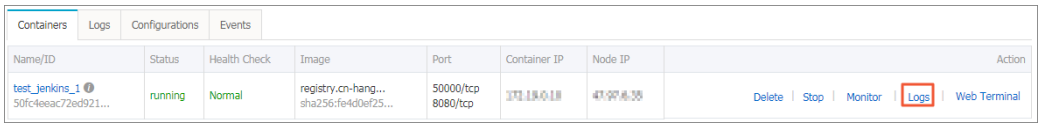


View container logs

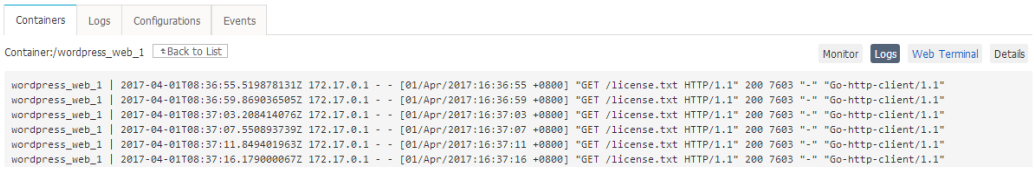
1. Log on to the Container Service console.
2. Click Swarm > **Services** in the left-side navigation pane.
3. Select the cluster from the Cluster drop-down list.
4. Click the service name. As shown in the following figure.



5. Under the Containers tab, click **Logs** at the right of the container you want to view logs. As shown in the following figure.



You can view the logs of this container.



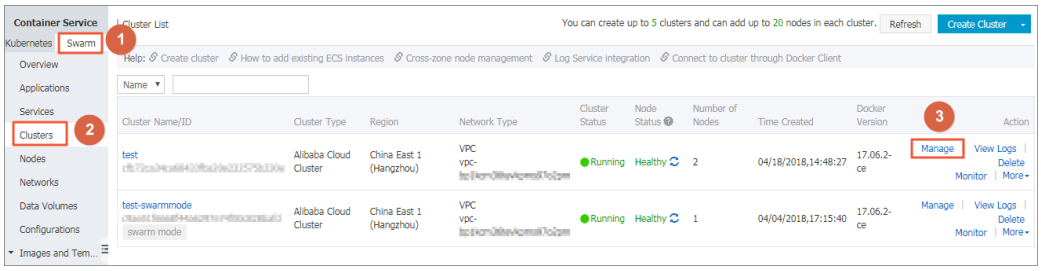
13.2. Enable Log Service

Log Service is a platform service for log scenarios. You can collect, distribute, ship, and query logs quickly without development, which is applicable to scenarios such as log transfer, monitoring, performance diagnosis, log analysis, and audit. Container Service integrates with Log Service, which allows you to send the application logs to Log Service.

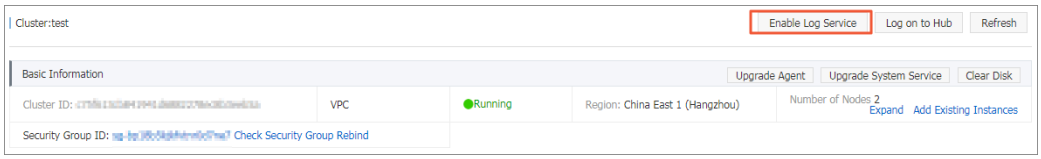
> **Note** On the cluster management page, choose **Enable Log Service > OK**. After Log Service is successfully enabled, the log index is created for each automatically created Logstore by using the built-in Resource Access Management (RAM) account. With this feature enabled, you are charged for the Alibaba Cloud Log Service usage after configuring the following settings. For more information, see Pay-as-you-go. Make sure you know your log volume to avoid large unexpected costs.

Enable Log Service

1. Log on to the Container Service console.
2. Click **Clusters** in the left-side navigation pane.
3. Click **Manage** at the right of the cluster.

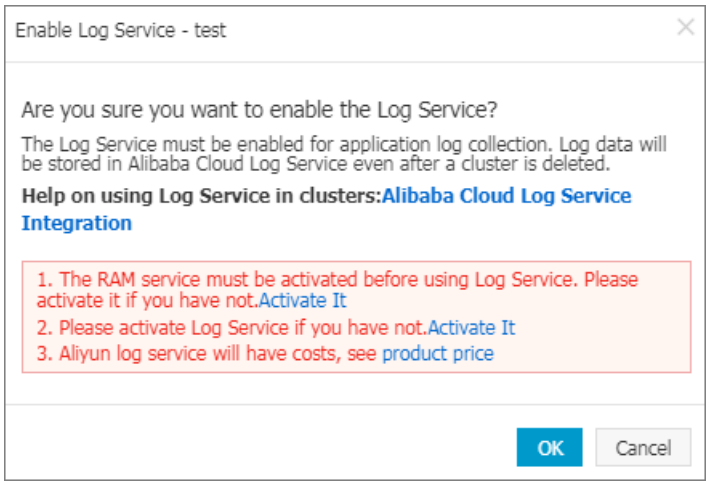


4. Click **Enable Log Service** in the upper-right corner.



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

   Before enabling Log Service in Container Service, activate the RAM service and Log Service first. Click **Activate It** to activate the RAM service and Log Service if they are not activated yet. The created Log Service project is displayed after Log Service is successfully enabled.

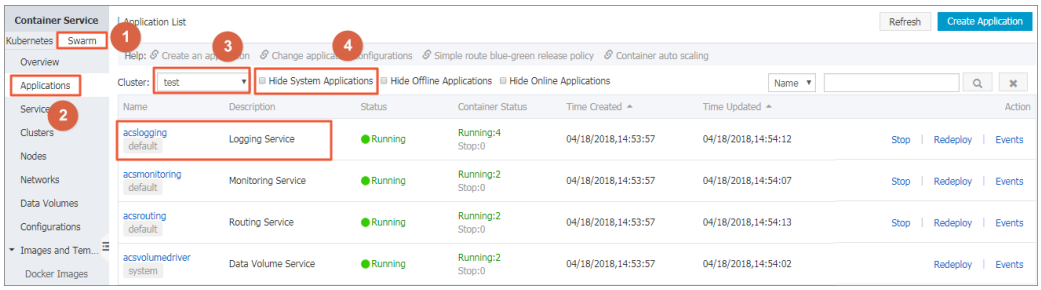


Check installation result of acslogging service

Container Service installs the Agent required by Log Service on your machine if this is the first time Log Service is enabled.You can use Log Service after the application is installed successfully. You can find this application on the Application List page. You can use Log Service after the application is installed successfully.

1. Log on to the Container Service console.
2. Click **Applications** in the left-side navigation pane.
3. Select the cluster from the Cluster list and clear the **Hide System Applications** check box.

   The acslogging application is successfully installed.



The system creates a corresponding project in Alibaba Cloud Log Service. You can view the project in the Log Service console. The project name contains the Container Service cluster ID.

Use Log Service in orchestration files

Most Docker applications write the logs directly to stdout, now you can do this as well (for the scenarios of writing logs to files, see Use file logs in the following section). After enabling Log Service, stdout logs are automatically collected and sent to Alibaba Cloud Log Service.

In the following example a WordPress application is created. It contains two services: WordPress service and MySQL service. Logs are collected to Alibaba Cloud Log Service. which contains two services: WordPress service and MySQL service. Logs are collected to Alibaba Cloud Log Service.

MySQL and WordPress

```
mysql:
  image: mysql
  ports:
      - 80
  labels:
      aliyun.scale: "1"
  environment:
      - MYSQL_ROOT_PASSWORD=password
web:
  image: registry.aliyuncs.com/jiangjizhong/wordpress
  ports:
     - 80
  labels:
     aliyun.routing.port_80: wordpress-with-log
     aliyun.log_store_dbstdout: stdout  # Collect stdout logs to the dbstdout Logstore.
     aliyun.log_ttl_dbstdout: 30  # Set the data retention time for the dbstdout Logstore t
o 30 days.
  links:
      - mysql
```

In the preceding orchestration file:

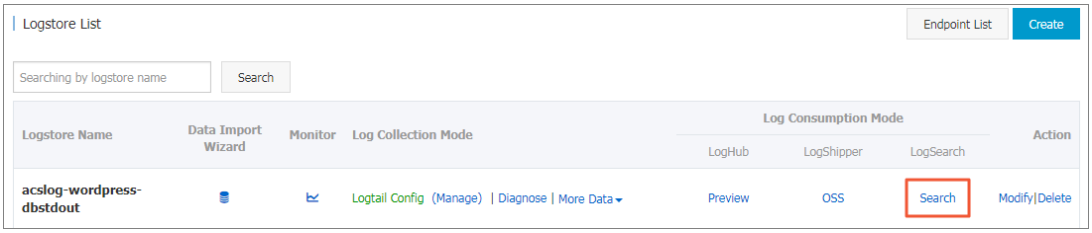
- `aliyun.log_store_dbstdout: stdout` indicates to write the container standard to the Logstore `acslog-wordpress-dbstdout`. The label format is `aliyun.log_store_{name}: {logpath}`。Wherein:
  - `name` is the name of the Alibaba Cloud Log Service Logstore. The actually created Logstore name is `acslog-${app}-${name}`.
  - `app` is the application name.
  - `logpath` is the log path in the container.
  - `stdout` is a special `logpath`, indicating the standard output.
- `aliyun.log_ttl_<logstore_name>` is used to set the data retention time (in days) for the Logstore. The value range 1-365. If left empty, logs are kept in the Logstore for two days by default.

> **Note** The value configured here is the initial configuration value. To modify the data retention time later, modify it in the Log Service console.

You can create an application named `wordpress` in the Container Service console by using the preceding orchestration file. After the application is started, you can find the Logstore `acslog-wordpress-dbstdout` in the Log Service console, in which stores the logs of application `wordpress`.

View logs in Log Service console

After deploying an application by using the preceding orchestration file, you can view the collected logs in the Alibaba Cloud Log Service console. Log on to the Log Service console. Find the Log Service project corresponding to the cluster. You can view the Logstore `acs-wordpress-dbstdout` used in the orchestration file.



Click **Search** at the right of the Logstore to view the logs.

Use file logs

To write the logs directly to files (for example, */var/log/app.log*) instead of stdout, configure as follows:

```
aliyun.log_store_name: /var/log/app.log
```

`name` is the Logstore name. */var/log/app.log* is the log path in the container.

To output multiple log files to Log Service, configure as follows to put the files under multiple directories:

```
aliyun.log_store_s1: /data/logs/access/access.log
aliyun.log_store_s2: /data/logs/error/error.log
aliyun.log_store_s3: /data/logs/exception/*.log #Wildcards are supported
```

Note Currently, multiple Logstores cannot correspond to the same log directory. The log files corresponding to the three Logstores s1, s2, and s3 in the preceding example must be under three directories.

Enable timestamp

You can select whether to add timestamp when Docker is collecting logs. Configure timestamp by using the `aliyun.log.timestamp` label in Container Service. label in Container Service. The timestamp is added by default.

- Add timestamp

  ```
  aliyun.log.timestamp: "true"
  ```

- Remove timestamp

  ```
  aliyun.log.timestamp: "false"
  ```

14.Monitoring

14.1. Container monitoring service

Context

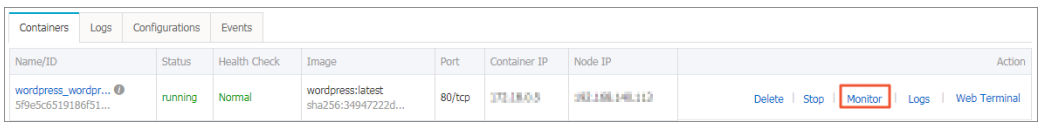
Container monitoring service depends on Alibaba Cloud CloudMonitor service, and provides container Operation & Maintenance (O&M) users with services such as default monitoring and alarm rule configurations. The monitoring service of Container Service provides the capabilities to demonstrate the monitoring data at the container dimension and give an alarm. Besides, Container Service can integrate with the third-party open-sourced monitoring solutions (for more information, see Integrate with third-party monitoring solutions).

Set alarm rules.

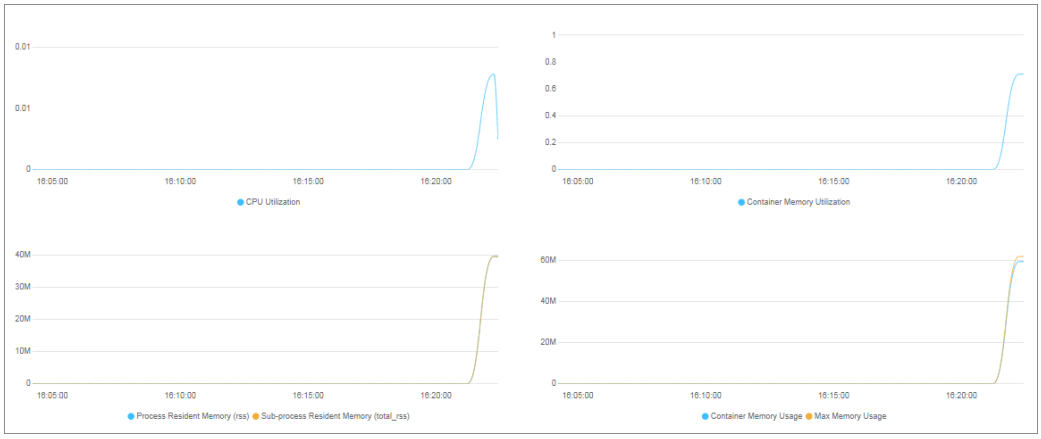
In some key services, you can add alarm rules according to your actual business situations. The container monitoring service will send an SMS notification to the cloud account contact when the monitoring metric reaches the alarm threshold.

Procedure

1. Log on to the Container Service console.
2. Go to the container list page by using:
   - **Nodes**.
     - Click **Nodes** in the left-side navigation pane.
     - Click the IP address of the node.
   - **Applications**.
     - Click **Applications** in the left-side navigation pane.
     - Select the cluster in which the application you want to view resides from the Cluster drop-down list.
     - Click the name of the application you want to view and then click the **Containers** tab.
   - **Services**.
     - Click **Services** in the left-side navigation pane.
     - Select the cluster in which the service you want to view resides from the Cluster drop-down list.
     - Click the name of the service you want to view.
3. In the container list, click **Monitor** at the right of the container you want to view.



You can view the real-time monitoring information of the container.



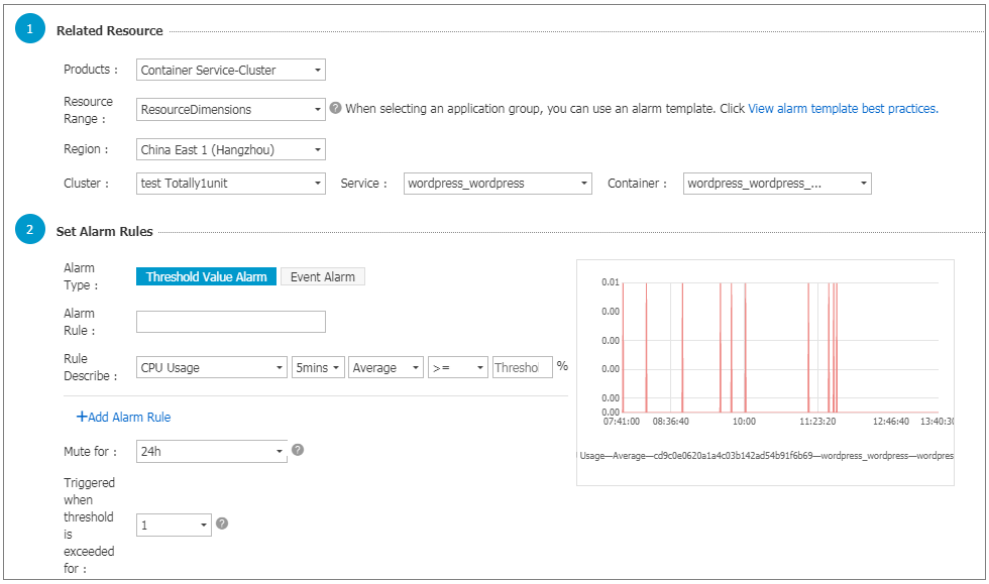
4. Log on to the CloudMonitor console. Click Cloud Service Monitoring > Container Service in the left-side navigation pane. Click **Container Service Monitoring** on the Clusters page. Select the container from the Container Service list. Click Monitoring Charts to view the history monitoring data of the container.

5. Set alarm rules.

   In some key services, you can add alarm rules according to your actual business situations. The container monitoring service will send an SMS notification to the cloud account contact when the monitoring metric reaches the alarm threshold.

   i. Click the Alarm Rules tab and then click **Create Alarm Rule** in the upper-right corner.

   ii. Based on your actual business requirements, configure the related resource, set the alarm rules,



   iii. and select the notification contact and method.

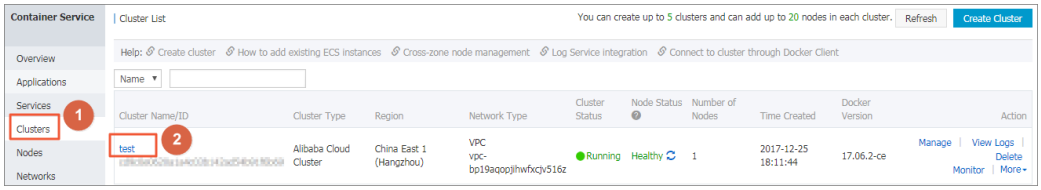
   iv. Complete the configurations as instructed on the page. Click **Confirm**.

   v. View the created alarm rule under the **Alarm Rules** tab.

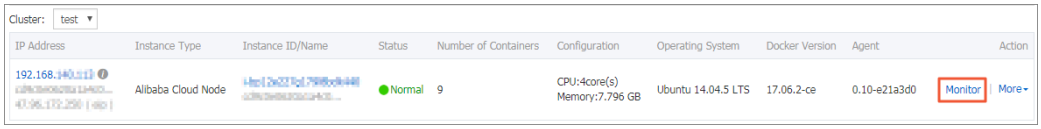
14.2. View monitoring information

Log

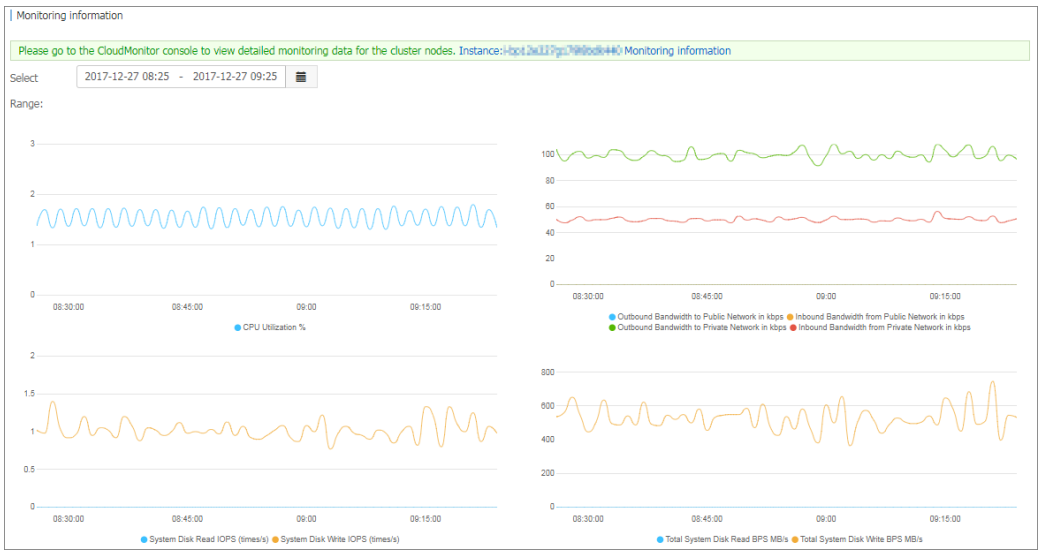
1. on to the Container Service console.
2. Click **Clusters** in the left-side navigation pane.
3. navigation pane.



4. Click **Monitor** at the right of the cluster.



You can view the monitoring information of this cluster.

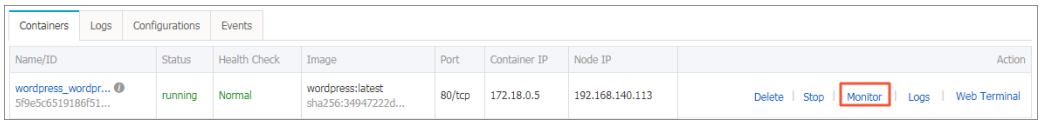


View container monitoring information

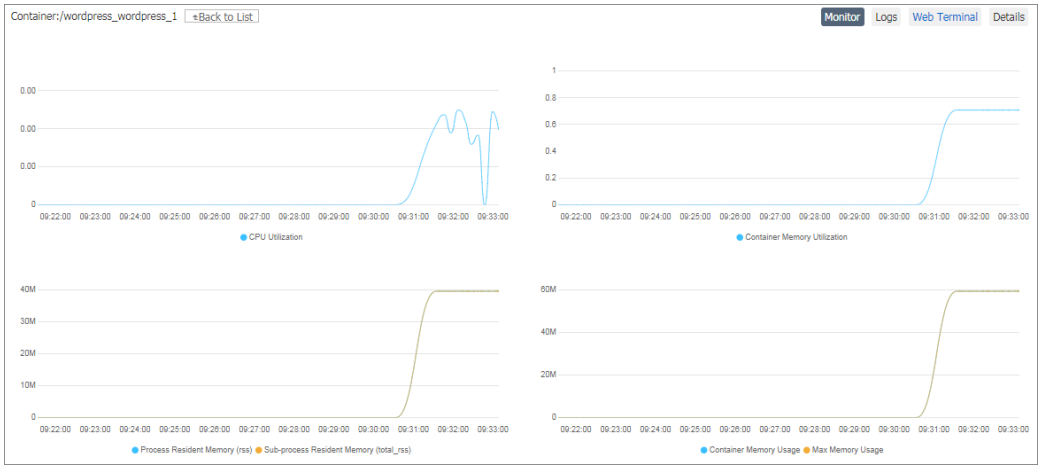
1. Log on to the Container Service console.
2. Go to the container list page by using:
   - **Nodes**.
     - Click **Nodes** in the left-side navigation pane.
     - Click the IP address of the node.
   - **Applications**.
     - Click **Applications** in the left-side navigation pane.
     - Select the cluster in which the application you want to view resides from the Cluster list.
     - Click the name of the application you want to view. Click the Containers tab.
   - **Services**.
     - Click **Services** in the left-side navigation pane.

- Select the cluster in which the service you want to view resides from the Cluster list.
- Click the name of the service you want to view.

3. In the container list, click **Monitor** at the right of the container you want to view.



You can view the monitoring information of this container.



14.3. Custom monitoring

The container monitoring service integrates with the Alibaba Cloud CloudMonitor service and provides you with monitoring and alarm services for containers, applications, clusters, and nodes. The container monitoring service meets the basic requirements for container monitoring. However, in many business scenarios, you might need custom monitoring to meet the monitoring requirements of your systems and applications. Therefore, besides the basic monitoring capabilities, the container monitoring service provides two custom monitoring modes, allowing you to report custom monitoring data by writing data and collecting scripts on your own or exposing your HTTP monitoring data interface. Container Service monitoring framework collects data every one minute by running the script or calling the HTTP interface.

Prerequisites

Before using the custom monitoring feature, you must integrate the container monitoring service with third-party monitoring solutions (for more information, see Integrate with third-party monitoring solutions).

> **Note** Currently, Container Service monitoring integration only supports InfluxDB and Prometheus by default.

Your custom monitoring data is reported to your InfluxDB or Prometheus, and then connected with your data presentation and analysis service.

Report monitoring data by using custom monitoring scripts

1. Create a Docker image and add a custom data collection script to this image.

The output data of this collection script must comply with the InfluxDB data format protocol. comply with the InfluxDB data format protocol.

```
weather,location=us-midwest temperature=82 1465839830100400200
|measurement|,tag_set| |field_set| |timestamp|
```

For more information about the data format protocol, see InfluxDB line protocol. protocol.

2. Log on to the Container Service console. Create an application by using an orchestration template. Use the `aliyun.monitoring.script` label to declare the data collection script used by the monitoring service.

   The sample template is as follows:

```
custom-script:
image: 'Your own image repository address'
labels:
       aliyun.monitoring.script: "sh gather_mem.sh"
```

   `The aliyun.monitoring.script` label defines the command in the application container that monitoring service runs to collect the monitoring data. The label is configured as follows:

```
labels:
    aliyun.monitoring.script: "command used to run the script"
```

3. Open the web interface of InfluxDB to view the database tables named after data indexes.

   For information about how to view the database tables, see Integrate with third-party monitoring solutions.

Collect data by using the custom HTTP monitoring data interface

1. Create a Docker image and expose the HTTP interface in the application.

   This interface outputs the monitoring data. You can customize the monitoring data format by conforming to the JSON syntax. In addition, the system cannot determine whether the JSON data returned from the custom HTTP interface is a data index field or a metadata tag of the data index. data returned from the custom HTTP interface is a data index field or a metadata tag of the data index. Therefore, use another configuration to specify what type of JSON data has the tag attribute. Telegraf JSON data format. For more information, see Telegraf JSON data format.

2. Log on to the Container Service console and create an application by using an orchestration template. In the template, add the aliyun.monitoring.http label to declare the data collection interface, and use aliyun.monitoring.tags: "your tag attribute name 1, your tag attribute name 2, ..." to declare what type of data fields returned from the HTTP data interface has the tag attribute. `aliyun.monitoring.http` label to declare the data collection interface, and use `aliyun.monitoring.tags: "your tag attribute name 1, your tag attribute name 2, ....."` to declare what type of data fields returned from the HTTP data interface has the tag attribute.

   Sample template:

```
nodejsapp:
command: "bash /run.sh"
ports:
      - "3000:3000"
image: 'Your own image repository address'
labels:
       aliyun.monitoring.http: "http://container:3000/metrics/data"
       aliyun.monitoring.tags: "tag1,tag2"
```

The data returned from the data interface http://container:3000/metrics/data exposed by the application nodejsapp is as follows: `http://container:3000/metrics/data exposed by the application nodejsapp` is as follows:

```
"tag1": "tag1value",
"tag2": "tag2value",
"field1": 1,
"field2": 2,
"field3": true,
"field4": 1.5
```

The `aliyun.monitoring.tags: "tag1,tag2"` label defines that in the reported JSON data, attributes tag1 and tag2 are the tags of the reported data.

3. Open the web interface of InfluxDB to view the database tables whose names consist of the `httpjson_` prefix and the container name.

   For example, if the container name is `nodejsapp_nodejsapp_1`, the name of the database table in InfluxDB is `httpjson_nodejsapp_nodejsapp_1`.

   For more information about how to view the database tables, see Integrate with third-party monitoring solutions.

14.4. Integrate with third-party monitoring solutions

Prerequisites

Create an application to be monitored. In this example, create an Nginx application. For more information, see Create an Nginx webserver from an image.

Context

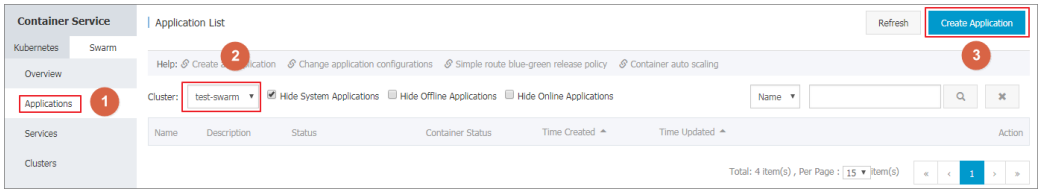
Container Service provides the capability to integrate with third-party open-sourced monitoring solutions.

> **Note** Currently, Container Service monitoring integration only supports InfluxDB and Prometheus by default.

The following example introduces how to integrate the Container Service monitoring service with third-party monitoring solutions by taking InfluxDB as an example.

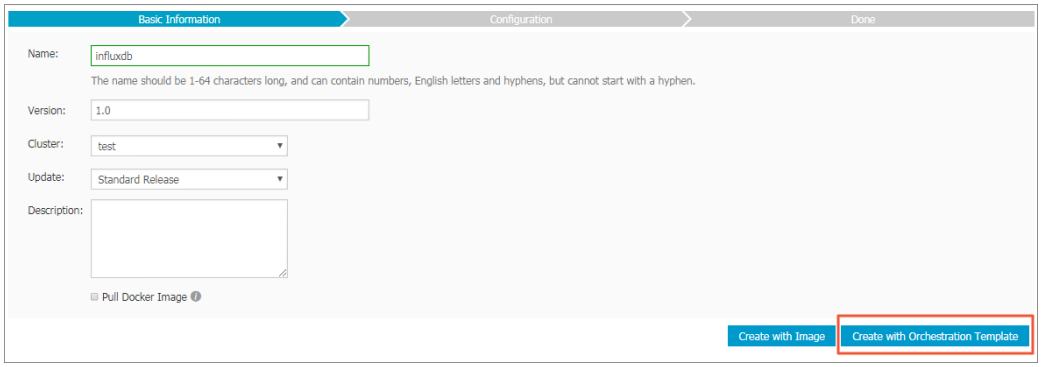
Procedure

1. Log on to the Container Service console.
2. Click **Applications** in the left-side navigation pane.
3. Click **Create Application** in the upper-right corner.



4. Enter the basic information of the application and click **Create with Orchestration Template**.

   In this example, the name of the application is influxdb.

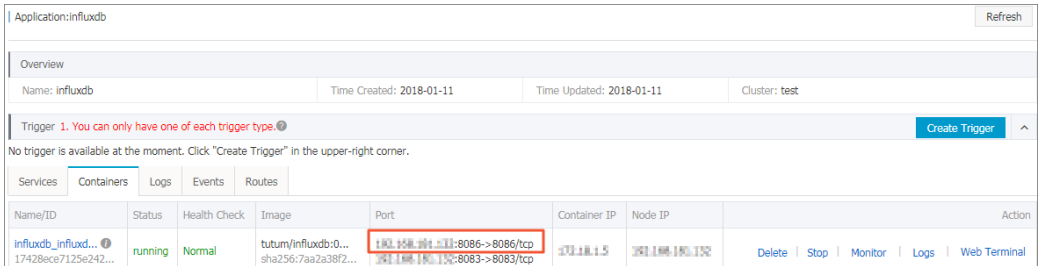


5. Enter the following orchestration template and click **Create and Deploy**.

> **Note** In a real production environment, the template in this example needs to be modified. Do not expose the port to the host in the `influxdb` service definition.

```
version: '2'
services: #Define influxdb.
 influxdb:
     image: tutum/influxdb:0.9
     volumes:
         - /var/lib/docker/influxdb:/data
     ports:
         - "8083:8083" #Expose Web interface port.
         - "8086:8086" #Expose data API Web interface port.
```

6. After the application is successfully created, click the application name influxdb on the Application List page to view the application details. Click the **Containers** tab to view the node IP and port exposed by this application. Copy the node IP and port. (In this example, copy the node IP and port number of the port 8086. These are the data reporting address exposed by influxdb.)



7. Return to the Application List page. Click **Update** at the right of influxdb. Add the following contents to the template to declare the integration of InfluxDB and the container monitoring service. Then, click **OK**.

```
labels:
     aliyun.monitoring.addon.influxdb: "http://The node IP and port are the ones you co
pied in step 7.
     aliyun.monitoring.addon.influxdb_retention_policy: "default"
```

> **Note** Currently, the third-party open-sourced monitoring integration only supports InfluxDB and Prometheus. The labels `for InfluxDB and Prometheus integration are aliyun.monitoring.addon.influxdb and` `aliyun.monitoring.addon.prometheus` respectively. The format of the label value must be `schema:hostIp:port`.

Container Service cannot use link to identify InfluxDB because the container monitoring service Agent adopts host network mode. Therefore, create the influxdb application and then add the data reporting address exposed by influxdb to the application labels so as to inform the data collection client. Then, the monitoring service automatically writes the running status data of containers collected by application influxdb to `influxdb`.

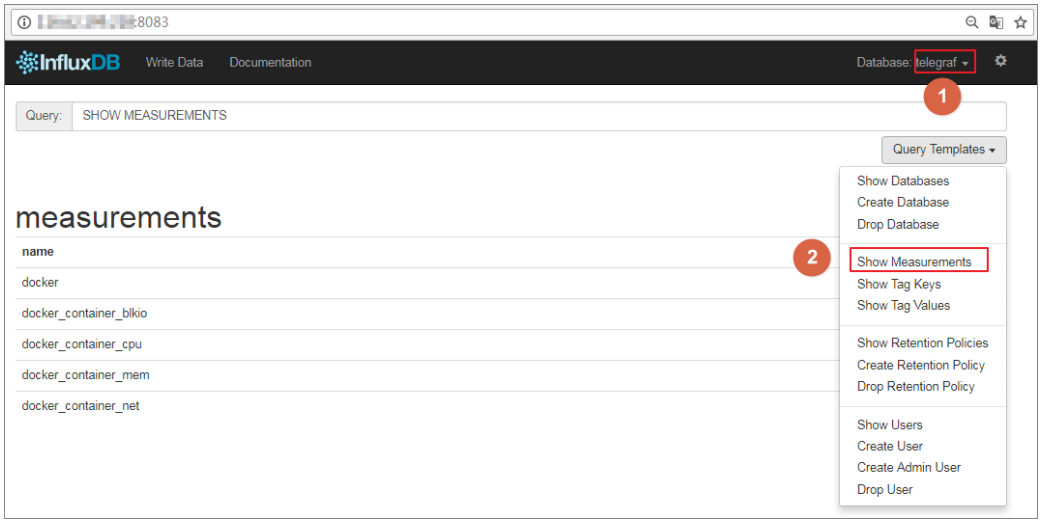
8. On the Application List page, click the application name influxdb and then click the **Containers** tab. Copy the port exposed by influxdb container.

9. We recommend that you access InfluxDB by using the Web proxy.

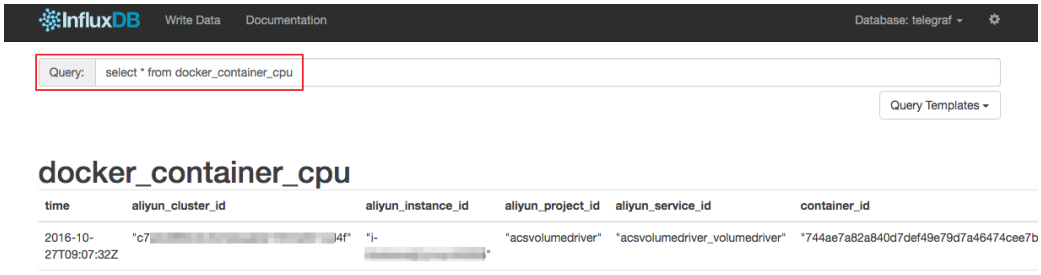
   To access InfluxDB by using Internet, follow these steps: Perform Internet authentication. Configure the inbound security group rules of ports 8083 and 8086 for the node in which the application influxdb resides. Access InfluxDB by using `http://EIP:8083` port.

10. Access the InfluxDB page in the browser to view the metric data written by the container monitoring service.
    i. Select **telegraf**. If **telegraf** does not exist, run `CREATE DATABASE telegraf` first and then redeploy the application that needs to be monitored.
    ii. Click **Query Templates** and select **Show Measurements** from the drop-down list.
    iii. Press Enter.

    You can view the database table.



View detailed data in a table.



What's next

After Container Service is integrated with InfluxDB, select other data charts and frameworks, such as Grafana, to display your monitoring data based on your own business situation.

14.5. Container auto scaling

To meet the demands of applications under different loads, Container Service supports auto scaling for the service, which automatically adjusts the number of containers according to the container resource usage in the service.

You can configure the container auto scaling rules when creating applications or add the container auto scaling rules for existing applications by changing application configurations.

Auto scaling policies:

- When the monitoring metric value exceeds the configured upper limit, Container Service increases the number of containers at your configured step.
- When the monitoring metric value is lower than the configured lower limit, Container Service reduces the number of containers at your configured step.

Service monitoring metrics:

- Average CPU usage
- Average memory usage
- Container inbound rate (currently only support being configured by using orchestration templates)
- Container outbound rate (currently only support being configured by using orchestration templates)

Prerequisites

- Upgrade the cluster Agent to the latest version. For more information, see Upgrade Agent.
- Upgrade the cluster monitoring service (acsmonitoring) to the latest version. For more information, see Upgrade system services.
- Activate the RAM service and update the RAM authorization information in the cluster by completing the following steps: Log on to the Container Service console. Click **Clusters** in the left-side navigation pane. Click **More** at the right of the cluster. Select **Update RAM Authorization Information** from the list.

Instructions

- When determining whether the monitoring metric value exceeds the configured upper limit or lower limit, Container Service uses the average value of the monitoring metrics (namely, the average CPU usage and the average memory usage) within a sample period (one minute). Container Service triggers scaling only when the average monitoring metrics of three consecutive sample periods all exceed the configured upper limit or lower limit so as to avoid frequent scaling caused by monitoring data jitter.
- During container contraction, the system deletes the containers in the cluster. Therefore, back up the data in advance.

Set container auto scaling

Create an application by using an image

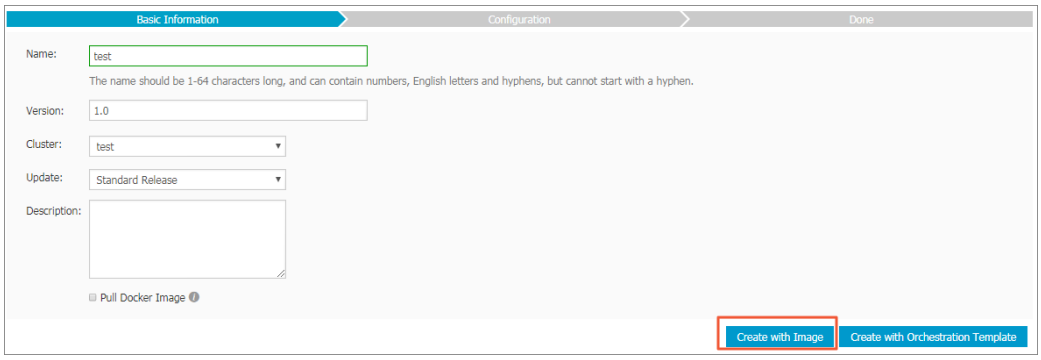
- when creating
- Create an application by using an orchestration template
- Change application configurations

> **Note** Click Create with Image when creating an application. For how to create an application, see Create an application.What if the auto scaling rule does not take effect Create with Image

when creating

1. an application.

   For how to create an application, see Create an application.

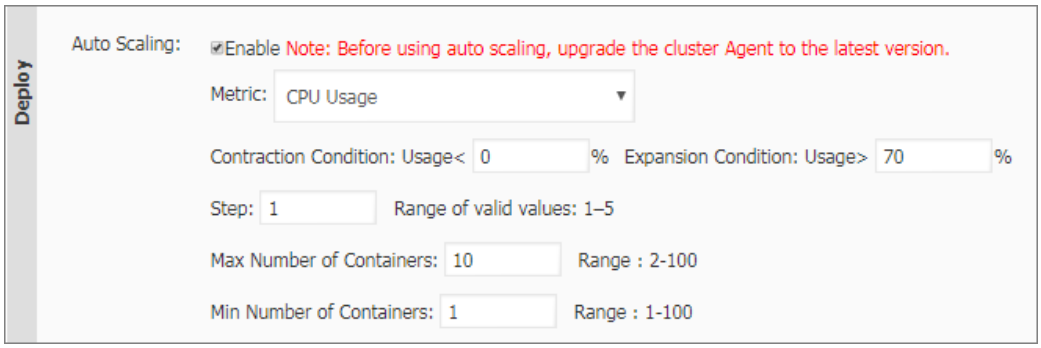


2. In the **Scaling** section at the bottom of the page, select the **Enable** check box for Auto Scaling and set the auto scaling parameters.

   Constraint rules:

- The range of the **Expansion Condition** is 50%-100%. The range of the **Contraction Condition** is 0%-50%.
- The **Expansion Condition** must be at least 30% higher than the **Contraction Condition**.
- The range of the **Step** is 1-5. The default value is 1.
- Set the **Min Number of Containers** and **Max Number of Containers**. For contraction, if the number of containers is less than or equal to the **Min Number of Containers**, contraction is not performed. For expansion, if the number of containers is greater than or equal to the **Max Number of Containers**, expansion is not performed.

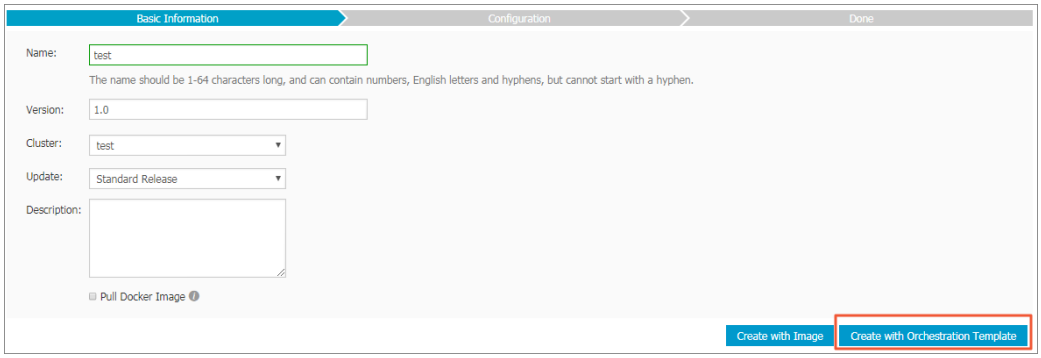
> **Note**
>
> - Set the scaling policies
> - with due care. If the application already meets the configured scaling conditions when you set the scaling rules and the application still meets the scaling conditions after the scaling, the monitoring will continuously trigger the scaling.



Create an application by using an orchestration template

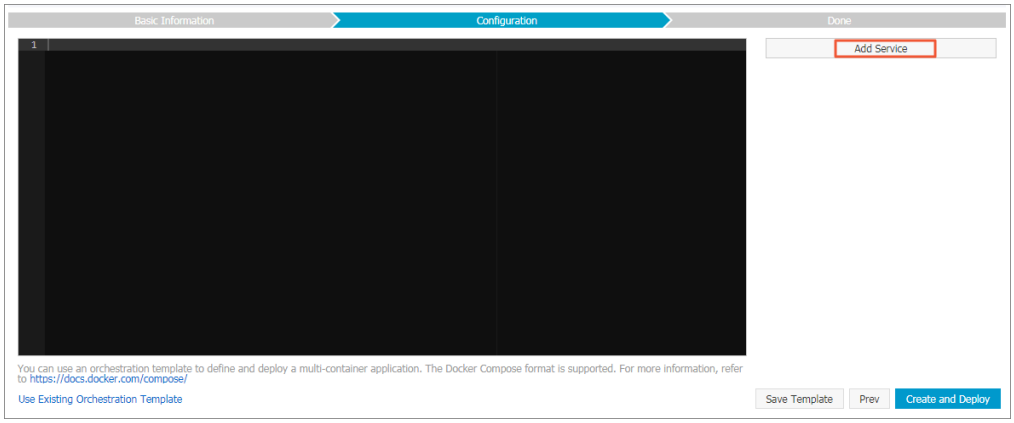
1. Click **Create with Orchestration Template** when creating an application.

   For how to create an application, see Create an application.



2. Click **Use Existing Orchestration Template** or write your own orchestration template.
3. Add the configurations of the container auto scaling by:
   - Clicking **Add Service**.

     In the displayed dialog box, select the image and configure the corresponding parameters. Click **More Settings**. Select the **Enable** check box for Auto Scaling and set the auto scaling parameters.

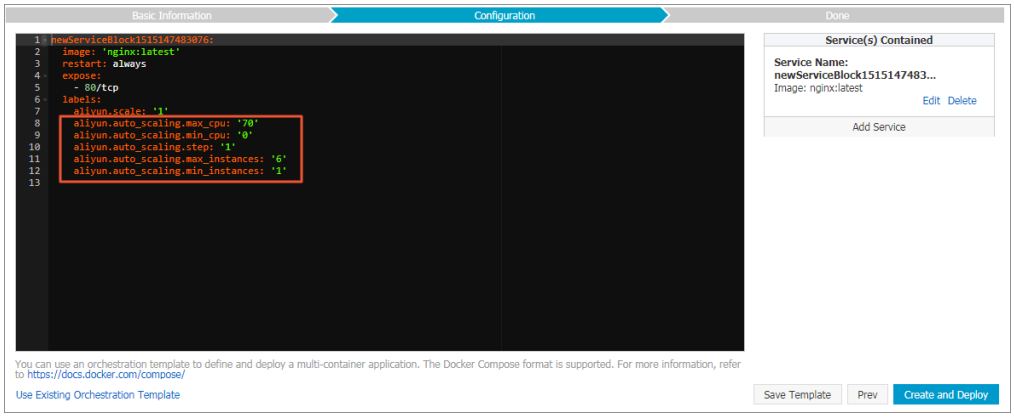


- Manually configuring in the template.

  In the `labels` configurations of the orchestration template, add the corresponding labels:

  - Specify the step (the default value is 1): `aliyun.auto_scaling.step`
  - Specify the minimum number of containers (the default value is 1): aliyun.auto_scaling.min_instances 1): `aliyun.auto_scaling.min_instances`
  - Specify the maximum number of containers (the default value is 10): `aliyun.auto_scaling.max_instances`
  - CPU usage as the metric
    - Specify the upper limit: `aliyun.auto_scaling.max_cpu`
    - Specify the lower limit: `aliyun.auto_scaling.min_cpu`
  - Memory usage as the metric
    - Specify the upper limit: `aliyun.auto_scaling.max_memory`
    - Specify the lower limit: `aliyun.auto_scaling.min_memory`
  - Outbound rate as the metric
    - Specify the upper limit: `aliyun.auto_scaling.max_internetOutRate`
    - Specify the lower limit: `aliyun.auto.scaling.min_internetOutRate`
  - Inbound rate as the metric
    - Specify the upper limit: `aliyun.auto_scaling.max_internetInRate`
    - Specify the lower limit: `aliyun.auto_scaling.min_internetInRate`

  Example

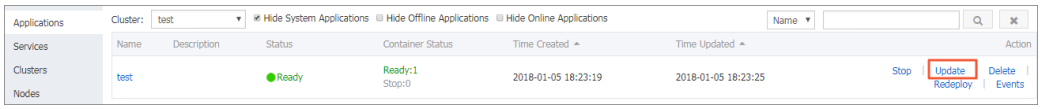


Change application configurations

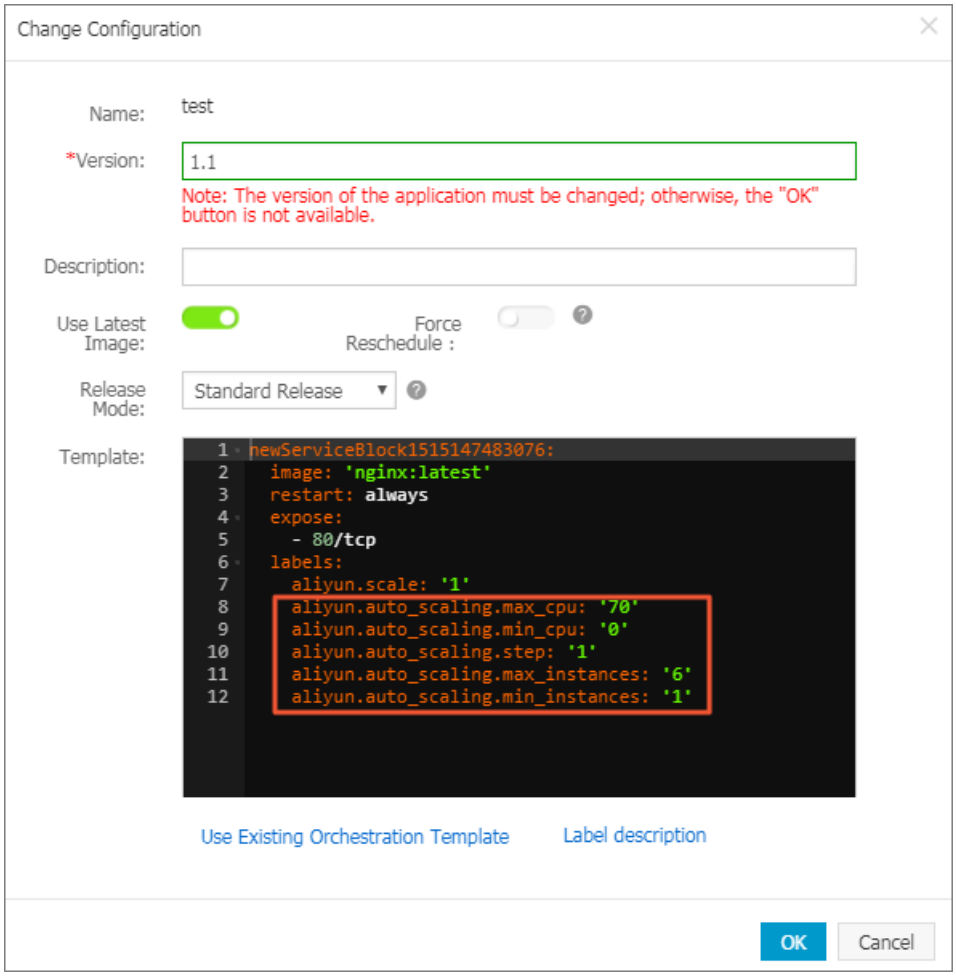
You can add container auto scaling settings by changing the configurations of an existing application.

1. On the Application List page, click **Update** at the right of the application you want to add the container auto scaling settings.

   For how to change the application configurations, see Change application configurations.



2. In the `labels` configurations in Template, add the corresponding container auto scaling labels.

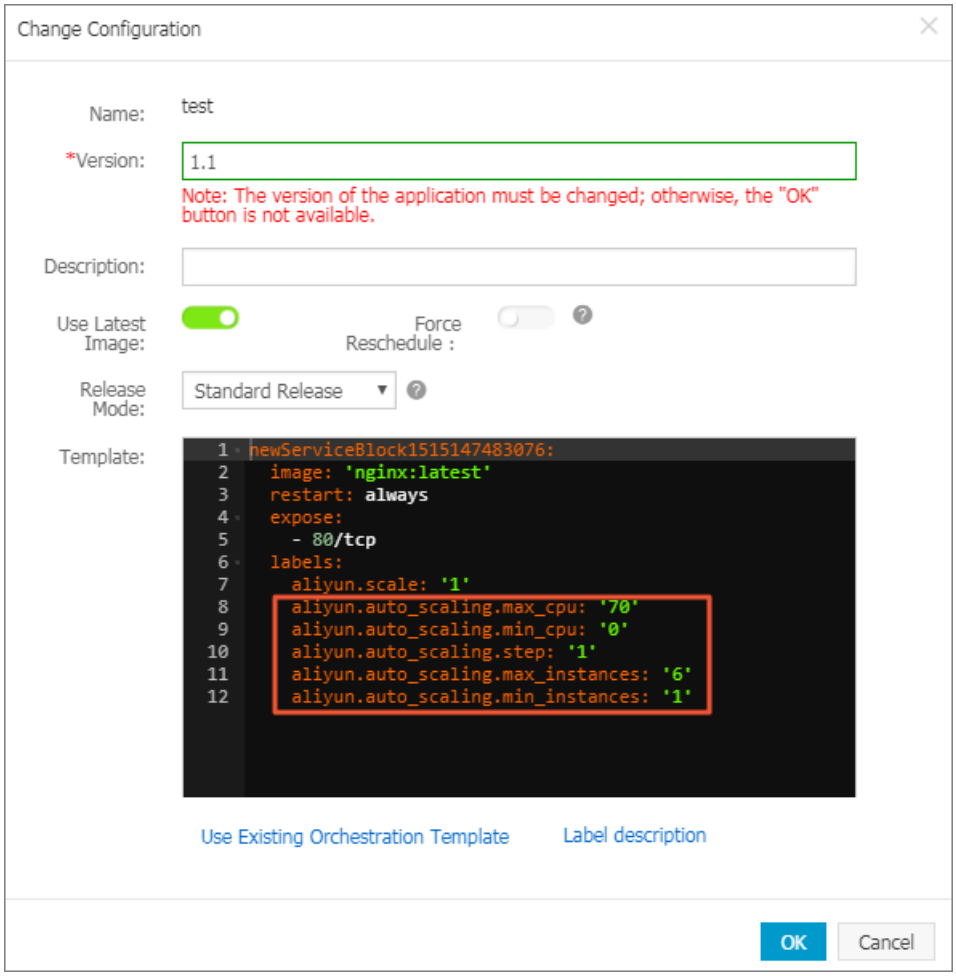


View created container scaling rules

You can view the created container scaling rules.

1. Log on to the Container Service console.
2. Click **Applications** in the left-side navigation pane.
3. On the **Application List** page, click **Update** at the right of the application.

   You can view the created container scaling rules. You can modify the container scaling rules in the Template.



14.6. Node auto scaling

To meet the demands of applications under different loads, Container Service provides the auto scaling both for containers and nodes. The node auto scaling is to automatically adjust the number of nodes by monitoring the node resource usage.

Node scaling policies:

- When the monitoring metric value exceeds the configured expansion condition, Container Service increases the number of nodes at your configured expansion step.
- When the monitoring metric value is lower than the configured contraction condition, Container Service reduces the number of nodes at the system default step 1.

Auto scaling monitoring metrics:

- Cluster CPU average usage
- Cluster memory average usage

Prerequisite

- Upgrade the cluster Agent to the latest version. For more information, see Upgrade Agent.
- Upgrade the cluster monitoring service (acsmonitoring) to the latest version. For more information, see Upgrade system services.
- Activate the RAM service and update the RAM authorization information in the cluster (Complete the

following steps: Log on to the Container Service console. Click **Clusters** in the left-side navigation pane. Click **More** at the right of the cluster. Select **Update RAM Authorization Information** from the list.

Instructions

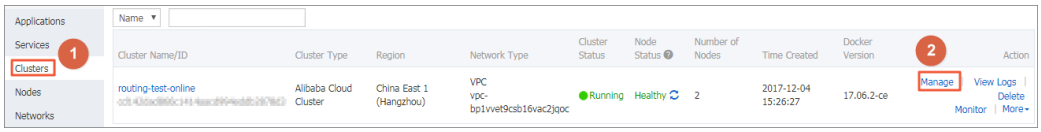
- When determining whether the monitoring metric value exceeds the configured upper limit or lower limit, Container Service uses the average value of the monitoring metrics (namely, the average CPU usage and the average memory usage) within a sample period (one minute). Container Service triggers scaling only when the average monitoring metrics of three consecutive sample periods all exceed the configured upper limit or lower limit so as to avoid frequent scaling caused by monitoring data jitter.
- Node contraction only contracts nodes that are created by means of node expansion. Your manually added or created nodes are not affected. To perform auto contraction on those manually added nodes, add the following label for those nodes:

  ```
  aliyun.reschedule==true
  ```

- During node contraction, the system deletes the Elastic Compute Service (ECS) instances in the cluster. Therefore, back up data in advance.
- Do not schedule services with statuses to nodes that can be contracted.
- The ECS instances added by expansion do not affect the deployed containers. The newly deployed containers are deployed according to the container deployment rules.
- During node contraction, Container Service migrates containers on the deleted ECS instances to other ECS instances.

Create node scaling rules

1. Log on to the Container Service console.
2. Click **Clusters** in the left-side navigation pane.
3. On the **Cluster List** page, click **Manage** at the right of the cluster.



4. Click **Node Scaling** in the left-side navigation pane and click **Create Scaling Rule**.

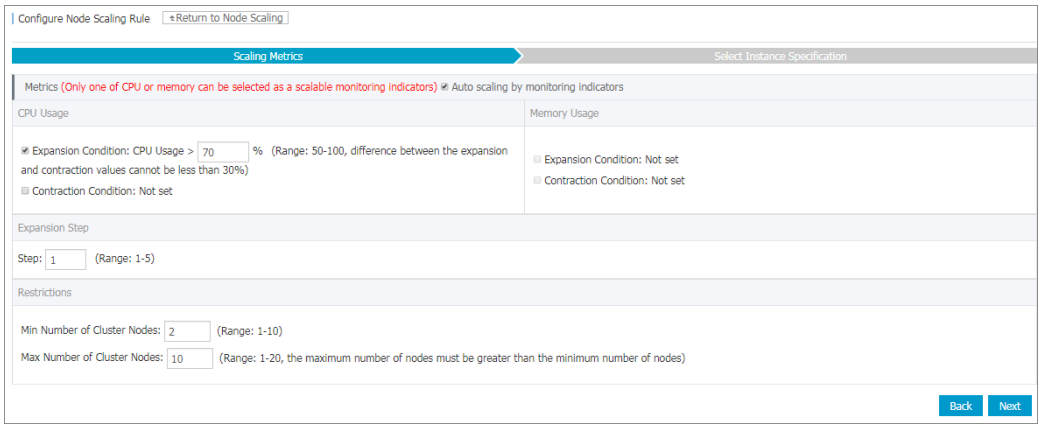
5. Configure the scaling rule and click **Next**.

   Constraint rules:

   - The range of the **Expansion Condition** is 50%-100%. The range of the **Contraction Condition** is 0%-50%.
   - The Expansion Condition must be at least 30% higher than the Contraction Condition.

- The range of the expansion step is 1-5. Currently, the default **contraction step** is 1 and cannot be configured.
- Set the **Min Number of Cluster Nodes** and **Max Number of Cluster Nodes**. For contraction, if the number of nodes is less than or equal to the **Min Number of Cluster Nodes**, contraction is not performed. For expansion, if the number of nodes is greater than or equal to the **Max Number of Cluster Nodes**, expansion is not performed.

> **Note** Set the scaling policies with due care. If the cluster already meets the configured scaling conditions when you set the scaling rule and the cluster still meets the scaling conditions after the scaling, the monitoring will continuously trigger the scaling.



6. Configure the instance specifications and click **Submit**.

   Configure the specifications for the nodes added by expansion after setting the expansion condition. For more information about how to configure the instance specifications, see Create a cluster.

   You can also configure whether or not to add the IP addresses of the nodes added by expansion to the RDS instance whitelist, which facilitates the ECS instances to access the RDS instances.

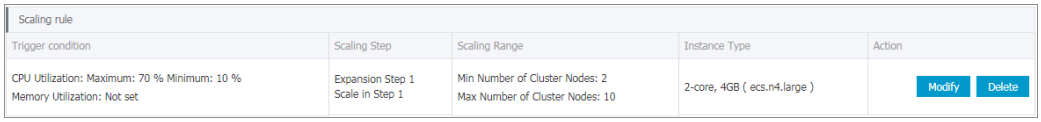
   > **Note** The ECS instance must be in the same region as the RDS instance so that the IP address of the ECS instance can be added to the RDS instance whitelist.

   Click **Select RDS Instances** in the Advanced section at the bottom of the page. The Add to RDS instance whitelist dialog box appears. Select the RDS instances and then click **OK**.

View created node scaling rules

You can view the crated node scaling rules.

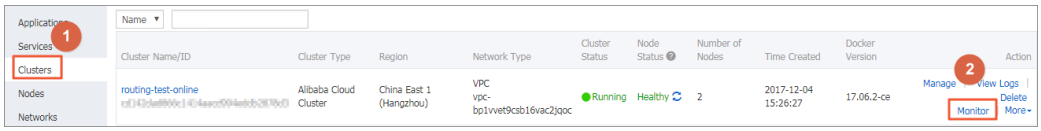
1. Log on to the Container Service console.
2. Click **Clusters** in the left-side navigation pane.
3. On the **Cluster List** page, click **Manage** at the right of the cluster.
4. Click **Node Scaling** in the left-side navigation pane. You can view the created node scaling rules.



You can click **Modify** to modify the node scaling rule or click **Delete** to delete this rule.

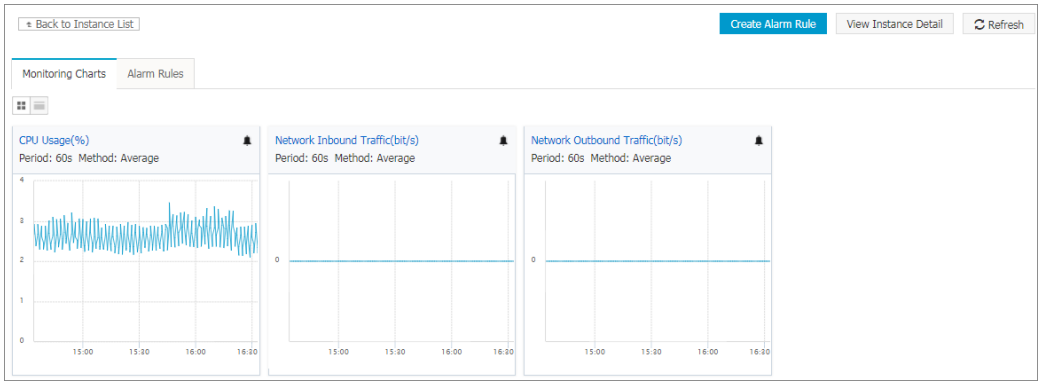
View monitoring metrics

1. Click **Clusters** in the left-side navigation pane.
2. On the **Cluster List page**, click **Monitor** at the right of the cluster.

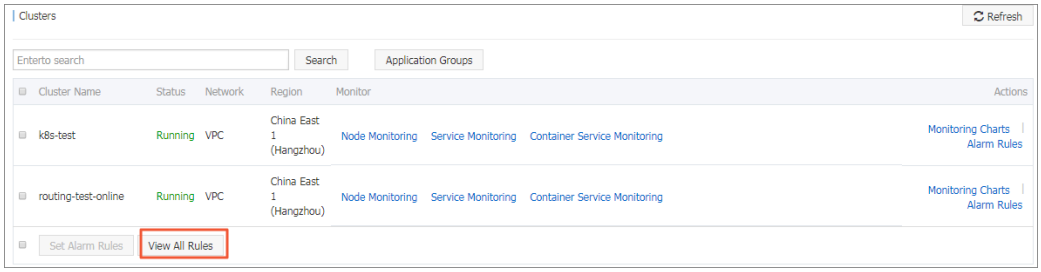


You are redirected to the CloudMonitor console and can view the cluster monitoring information.

> **Note** If the monitoring data does not exist, check whether or not the monitoring service (acsmonitoring) is correctly installed. If not, redeploy the monitoring service (acsmonitoring). Check whether or not the cluster Agent is in the latest version. If not, upgrade the cluster Agent. Check whether or not the monitoring service (acsmonitoring) is in the latest version. If not, upgrade the monitoring service (acsmonitoring).

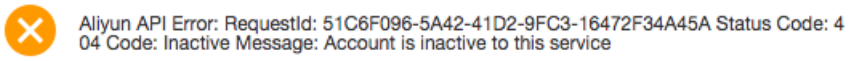


3. Click **Container Service** in the left-side navigation pane. Click **View All Rules**.



4. On the Alarm Rules page, you can view the automatically configured alarm rules of auto scaling.

   If no monitoring alarm rule exists, update the RAM authorization information in the cluster (On the **Cluster List** page, click **More > Update RAM Authorization Information** at the right of the cluster). Activate the RAM service before updating the RAM authorization information. Otherwise, the system reports an error.



5. Select an alarm rule to modify the alarm conditions and notification contacts who can be notified in SMSs, e-mails, and other ways. You can also disable the alarm rule.

Troubleshoot

If your configured node auto scaling rule does not take effect, see What if the auto scaling rule does not take effect for troubleshooting.

14.7. Monitoring metrics

Container Service provides multi-dimensional monitoring services. You can view the monitoring information in the Container Service console or CloudMonitor console.

The monitoring metrics of Container Service are divided into two parts: the automatic monitoring metrics and the cloud monitoring plug-in metrics. memory, network, I/O, and exception of Container Service helps you understand the usage of Container Service. You can log on to the CloudMonitor console and enter the Container Service page to view the monitoring details, and set alarm rules on monitoring metrics to receive alarm notifications when the metrics encounter an exception. Monitoring the metrics such as CPU usage,Monitoring the metrics such as CPU usage, memory, network, I/O, and exception of Container Service helps you understand the usage of Container Service. You can log on to the CloudMonitor console and enter the Container Service page to view the monitoring details, and set alarm rules on monitoring metrics to receive alarm notifications when the metrics encounter an exception.

Container Service provides monitoring information in the following dimensions:

- Cluster
- Node
- Service
- Container

> **Note** Install the cloud monitoring plug-in when creating a cluster to obtain the cluster and node memory metrics. For more information, see Create a cluster. To view more detailed metrics of Elastic Compute Service (ECS) instances, go to the CloudMonitor console to view the ECS monitoring metrics. For more information, see Metrics.

Monitoring metrics description

Cluster monitoring metrics

Monitoring metrics	Dimension	Unit	Minimum monitoring granularity	Means of aggregation
CPU usage	Cluster dimension	Percentage	60 seconds	Average
Network inbound traffic	Cluster dimension	bit/s	60 seconds	Average

Monitoring metrics	Dimension	Unit	Minimum monitoring granularity	Means of aggregation
Network outbound traffic	Cluster dimension	bit/s	60 seconds	Average
Memory usage	Cluster dimension	Percentage	60 seconds	Average
GPU memory usage	Cluster dimension	Bytes	60 seconds	Average
GPU usage	Cluster dimension	Percentage	60 seconds	Average
GPU temperature	Cluster dimension	Degree centigrade	60 seconds	Average

Node monitoring metrics

Monitoring metrics	Dimension	Unit	Minimum monitoring granularity	Means of aggregation
CPU usage	Node dimension	Percentage	60 seconds	Average
Network inbound traffic	Node dimension	bit/s	60 seconds	Average
Network outbound traffic	Node dimension	bit/s	60 seconds	Average
Memory usage	Node dimension	Percentage	60 seconds	Average
GPU memory usage	Node dimension	bytes	60 seconds	Average
GPU usage	Node dimension	Percentage	60 seconds	Average
GPU temperature	Node dimension	Degree centigrade	60 seconds	Average

Service monitoring metrics

Monitoring metrics	Dimension	Unit	Minimum monitoring granularity	Means of aggregation
CPU usage	Service dimension	Percentage	60 seconds	Average
Network inbound rate	Service dimension	Bytes/s	60 seconds	Average
Network outbound rate	Service dimension	Bytes/s	60 seconds	Average
Service I/O read	Service dimension	Bytes	60 seconds	Average

Monitoring metrics	Dimension	Unit	Minimum monitoring granularity	Means of aggregation
Service I/O write	Service dimension	Bytes	60 seconds	Average
Memory usage	Service dimension	Bytes	60 seconds	Average
Memory usage	Service dimension	Percentage	60 seconds	Average
Network inbound traffic	Service dimension	Bytes	60 seconds	Average
Network outbound traffic	Service dimension	Bytes	60 seconds	Average
Container I/O read rate	Container dimension	Bytes/s	60 seconds	Average
Container I/O write rate	Container dimension	Bytes/s	60 seconds	Average
Service GPU memory usage	Service dimension	Bytes	60 seconds	Average

Container monitoring metrics

Monitoring metrics	Dimension	Unit	Minimum monitoring granularity	Means of aggregation
CPU usage	Container dimension	Percentage	60 seconds	Average
Network inbound rate	Container dimension	Bytes/s	60 seconds	Average
Network outbound rate	Container dimension	Bytes/s	60 seconds	Average
I/O read	Container dimension	Bytes	60 seconds	Average
I/O write	Container dimension	Bytes	60 seconds	Average
Memory usage	Container dimension	Percentage	60 seconds	Average
Memory usage	Container dimension	Bytes	60 seconds	Average
Network inbound traffic	Container dimension	Bytes	60 seconds	Average

Monitoring metrics	Dimension	Unit	Minimum monitoring granularity	Means of aggregation
Network outbound traffic	Container dimension	Bytes	60 seconds	Average
Container I/O read rate	Container dimension	Bytes/s	60 seconds	Average
Container I/O write rate	Container dimension	Bytes/s	60 seconds	Average
GPU memory usage	Container dimension	Bytes	60 seconds	Average

Precautions:

Go to the CloudMonitor console to view history monitoring data or set monitoring alarm rules.

- Monitoring data is preserved for at most 31 days.
- You can view monitoring data for at most 14 consecutive days.
- You can set the alarm rules in batches.

14.8. What if the auto scaling rule does not take effect

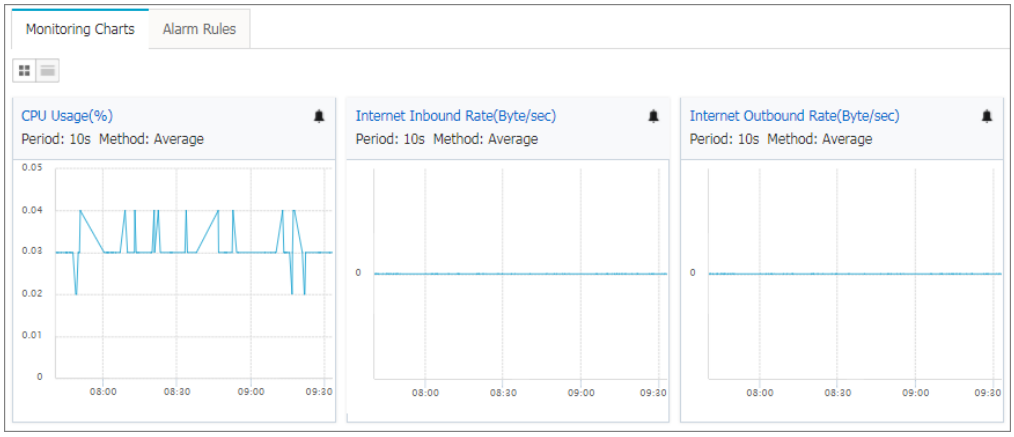
If your created container auto scaling rule or node auto scaling rule does not take effect, you can perform troubleshooting according to the following methods.

View monitoring metrics, and confirm that monitoring metrics have data and the data reaches the configured threshold for a certain period of time

- **View the monitoring metrics of container auto scaling**
  i. Click **Services** in the left-side navigation pane.
  ii. Select the cluster in which the service resides from the Cluster drop-down list.
  iii. Click **Monitor** at the right of the service.

  You are redirected to the CloudMonitor console and can view the container monitoring information. Confirm that monitoring metrics have data and the data reaches the configured threshold for a certain period of time.

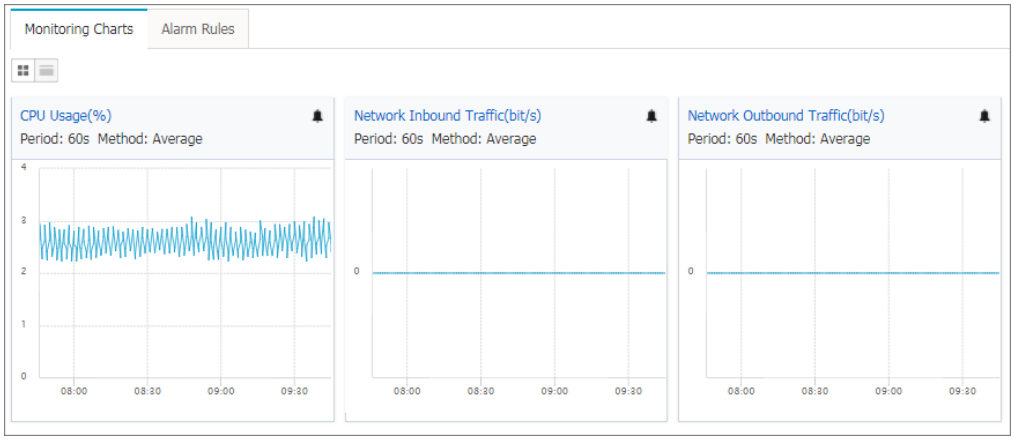
  > **Note** When determining whether the monitoring metric value exceeds the configured upper limit or lower limit, Container Service uses the average value of the monitoring metrics (namely, the average CPU usage and the average memory usage) within a sample period (one minute). Container Service triggers scaling only when the average monitoring metrics of three consecutive sample periods all exceed the configured upper limit or lower limit so as to avoid frequent scaling caused by monitoring data jitter.



- **View the monitoring metrics of node auto scaling**
  i. Click **Clusters** in the left-side navigation pane.
  ii. Click **Monitor** at the right of the cluster.

  You are redirected to the CloudMonitor console and can view the cluster monitoring information. Confirm that monitoring metrics have data and the data reaches the configured threshold for a certain period of time.

> **Note** When determining whether the monitoring metric value exceeds the configured upper limit or lower limit, Container Service uses the average value of the monitoring metrics (namely, the average CPU usage and the average memory usage) within a sample period (one minute). Container Service triggers scaling only when the average monitoring metrics of three consecutive sample periods all exceed the configured upper limit or lower limit so as to avoid frequent scaling caused by monitoring data jitter.



If no monitoring data is displayed

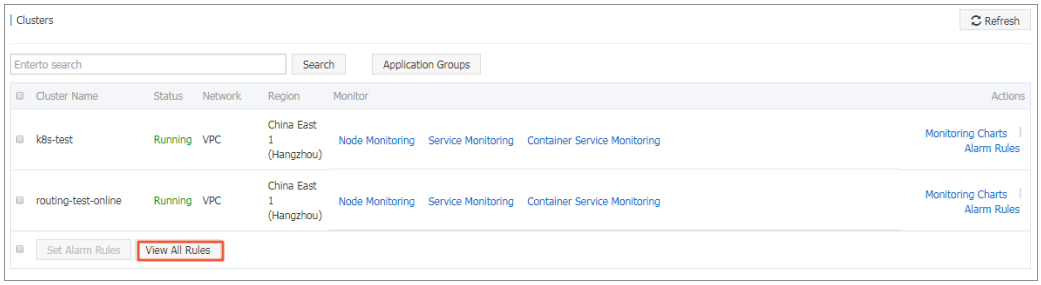
Check the following possibilities: Whether or not the monitoring service (acsmonitoring) is correctly installed (whether or not the status and number of containers are correct). If not, redeploy the monitoring service (acsmonitoring). Whether or not the cluster Agent is in the latest version. If not, upgrade the cluster Agent. Whether or not the monitoring service (acsmonitoring) is in the latest version. If not, upgrade the monitoring service (acsmonitoring). For more information, see Upgrade system services and Upgrade Agent.

View monitoring alarm rules and status, and confirm the rules are created and in the correct status

1. In the CloudMonitor console, click **Cloud Service Monitoring > Container Service** in the left-side navigation pane.

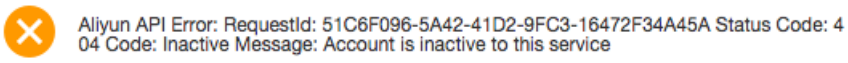
   You can view the cluster list.

2. Click **View All Rules** on the Clusters page to view the alarm rules automatically configured by auto scaling.



If no monitoring alarm rule is displayed

- Update the RAM authorization information in the cluster by completing the following steps: On the **Cluster List** page, click **More > Update RAM Authorization Information** at the right of the cluster. Activate the RAM service before updating the RAM authorization information. Otherwise, the system reports an error.



- Check the following possibilities: Whether or not the monitoring service (acsmonitoring) is correctly installed (whether or not the status and number of containers are correct). If not, redeploy the monitoring service (acsmonitoring). Whether or not the cluster Agent is in the latest version. If not, upgrade the cluster Agent. Whether or not the monitoring service (acsmonitoring) is in the latest version. If not, upgrade the monitoring service (acsmonitoring). For more information, see Upgrade system services and Upgrade Agent.

View alarm history

If the status of an alarm rule is abnormal (in the **Alarm** status) on the Alarm Rules page, you can view the alarm history of this rule to locate the problem.

Under Actions, click **View** at the right of the alarm rule.

Click the **Alarm Logs** tab, and you can select the time to view the alarms occurred within specified time period.

15.DevOps

15.1. Jenkins-based continuous delivery

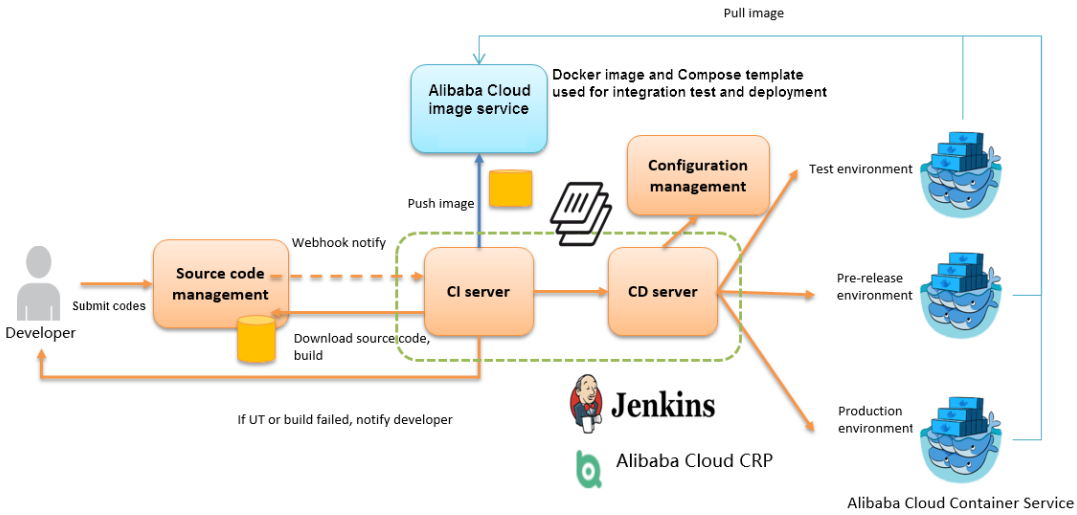
As an important step in agile development, continuous integration aims to maintain high quality while accelerating product iteration. Every time codes are updated, an automated test is performed to test the codes and function validity. The codes can only be delivered and deployed after they pass the automated test. This document mainly introduces how to integrate Jenkins, one of the most popular continuous integration tools, with Alibaba Cloud Container Service to realize automated test and image building push.

The following example demonstrates how to perform automated test and build a Docker image by using Alibaba Cloud Container Service Jenkins, which realizes high-quality continuous integration.

Background information

Every time codes are submitted to nodejs project in GitHub, Alibaba Cloud Container Service Jenkins will automatically trigger a unit test. If the test is successful, Jenkins continues to build images and then pushes them to a target image repository. Finally, Jenkins notifies you of the results by email.

A general process is as follows.



Slave-nodejs is a slave node used for unit test and building and pushing the image.

Jenkins introduction

Jenkins is an open-sourced continuous integration tool developed on Java. It monitors and triggers continuously repeated work and supports expansion of multiple platforms and plug-ins. Jenkins is an open-sourced tool featuring easy installation and interface-based management. It uses job to describe every work step, and node is a project execution environment. The master node is a default execution environment of a Jenkins job and also the installation environment for Jenkins applications.

Master/slave

Master/slave is equivalent to the server/agent concept. A master provides Web interface with which you manage the job and slave. The job can run on the master or be assigned to the slave. One master can be associated with several slaves to serve different jobs or different configurations of the same job.

Several slaves can be configured to prepare a separate test and building environment for different projects.

> **Note** The Jenkins job and project mentioned in this document all refer to a build unit of Jenkins, namely, an execution unit.

Step 1 Deploy Jenkins applications and slave nodes

The building and testing of different applications need different dependencies. The best practice is to use different slave containers with corresponding runtime dependencies and tools to perform the test and building. By using the slave images and sample templates provided by Alibaba Cloud Container Service for different environments such as Python, Node.js, and Go, you can quickly and easily generate Jenkins applications and various slave nodes, configure node information in Jenkins applications, and specify the execution nodes in the build projects so as to implement the entire continuous integration process.

> **Note** For images provided by Alibaba Cloud Container Service for developing slave nodes, see https://github.com/AliyunContainerService/jenkins-slaves.

1.1 Create a Jenkins orchestration template

Create a template and create the orchestration based on the following contents.

The labels supported by Alibaba Cloud Container Service Jenkins master are: 1.651.3, 2.19.2, and 2.32.2.

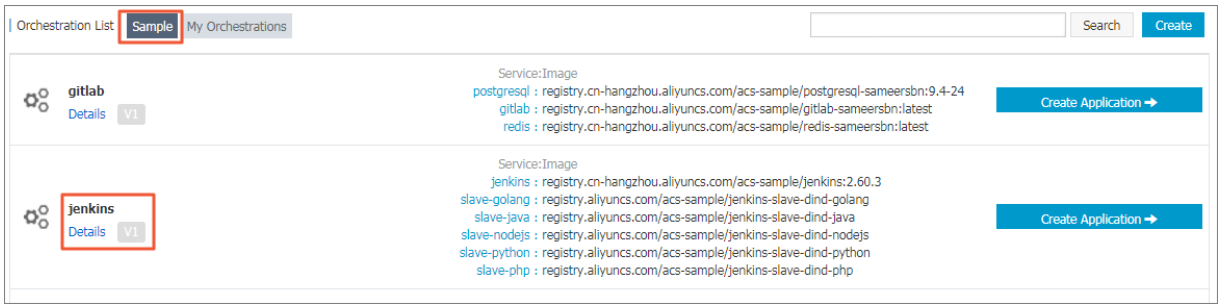
> **Note** For how to create an orchestration template, see Create an orchestration template.

```
jenkins:
    image: 'registry.aliyuncs.com/acs-sample/jenkins:1.651.3'
    volumes:
        - /var/lib/docker/jenkins:/var/jenkins_home
    restart: always
    labels:
        aliyun.scale: '1'
        aliyun.probe.url: 'tcp://container:8080'
        aliyun.probe.initial_delay_seconds: '10'
        aliyun.routing.port_8080: jenkins
    links:
        - slave-nodejs
slave-nodejs:
    image: 'registry.aliyuncs.com/acs-sample/jenkins-slave-dind-nodejs'
    volumes:
        - /var/run/docker.sock:/var/run/docker.sock
    restart: always
    labels:
        aliyun.scale: '1'
```

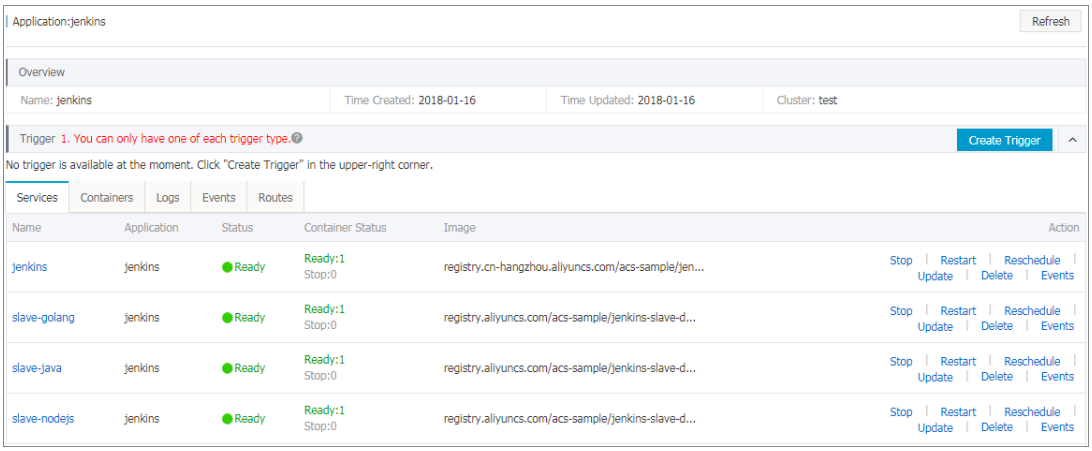
1.2 Use the template to create Jenkins application and slave node

Use the orchestration template created in the preceding section or the Jenkins sample template provided by Alibaba Cloud Container Service to create the Jenkins application and slave node.

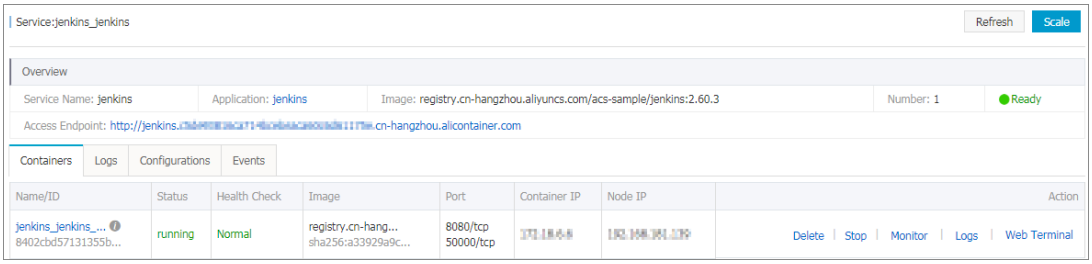
> **Note** For how to create an application by using an orchestration template, see Create an application.



After a successful creation, the Jenkins application and slave node are displayed in the service list.



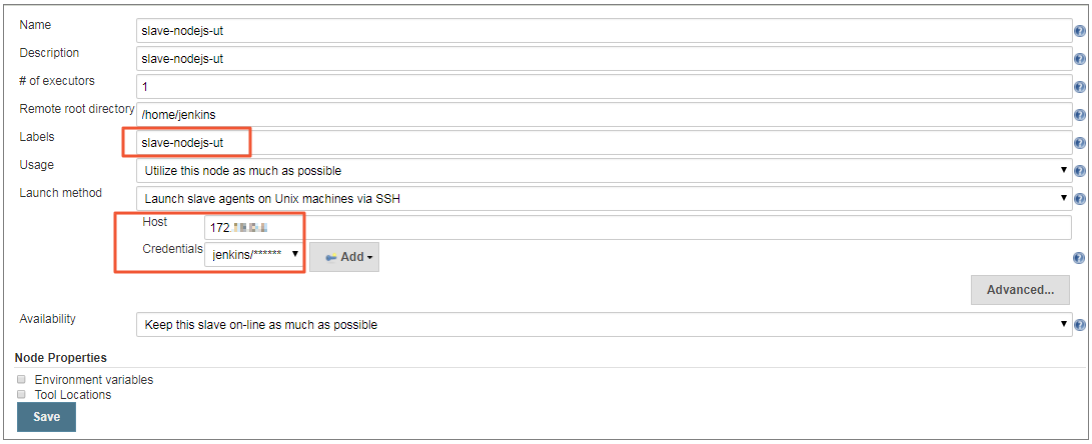
Open the access endpoint provided by Container Service to use the deployed Jenkins application.



Step 2 Realize automated test and automated build and push of image

2.1 Configure the slave container as the slave node of the Jenkins application

Open the Jenkins application. Click Manage Jenkins in the left-side navigation pane. Click Manage Nodes on the right pane. Click New Node in the left-side navigation pane. Enter the node name and then click OK. Then, complete the parameters as follows.

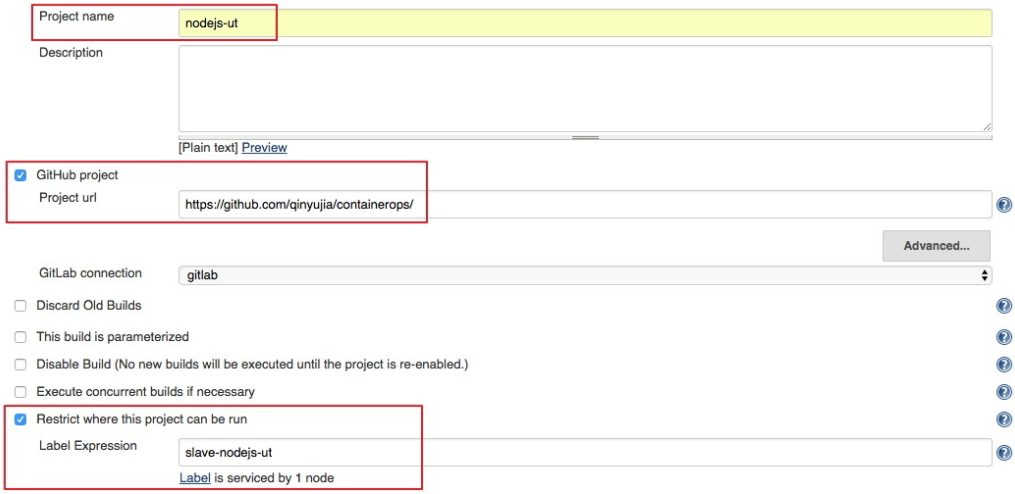


Note

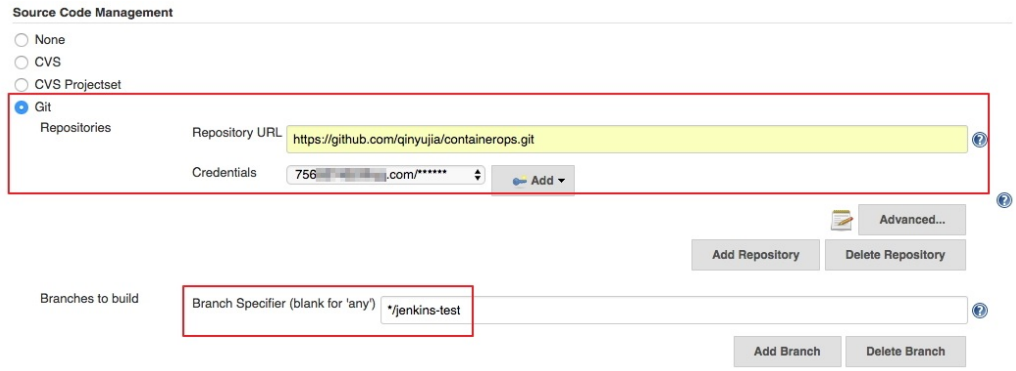
- Label is the unique identifier of the slave.
- The slave container and Jenkins container run on the Alibaba Cloud platform at the same time. Therefore, enter a container node IP address that is inaccessible to the Internet to isolate the test environment.
- When adding the credentials, use the jenkins account and password (the initial password is jenkins) in Dockerfile for the creation of the slave-nodejs image. The image Dockerfile address is jenkins-slave-dind-nodejs.

2.2 Create a project to implement automated test

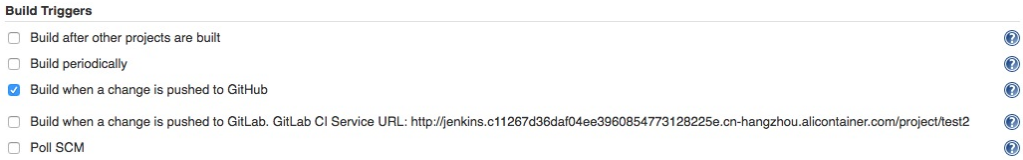
1. Go back to the Jenkins home page. Click New Item in the left-side navigation pane. Enter the item name, select Freestyle project, and then click OK.
2. Enter the project name and select a node for running the project. In this example, enter the slave-nodejs-ut node prepared in the preceding section.



3. Configure the source code management and code branch. In this example, use GitHub to manage source codes.



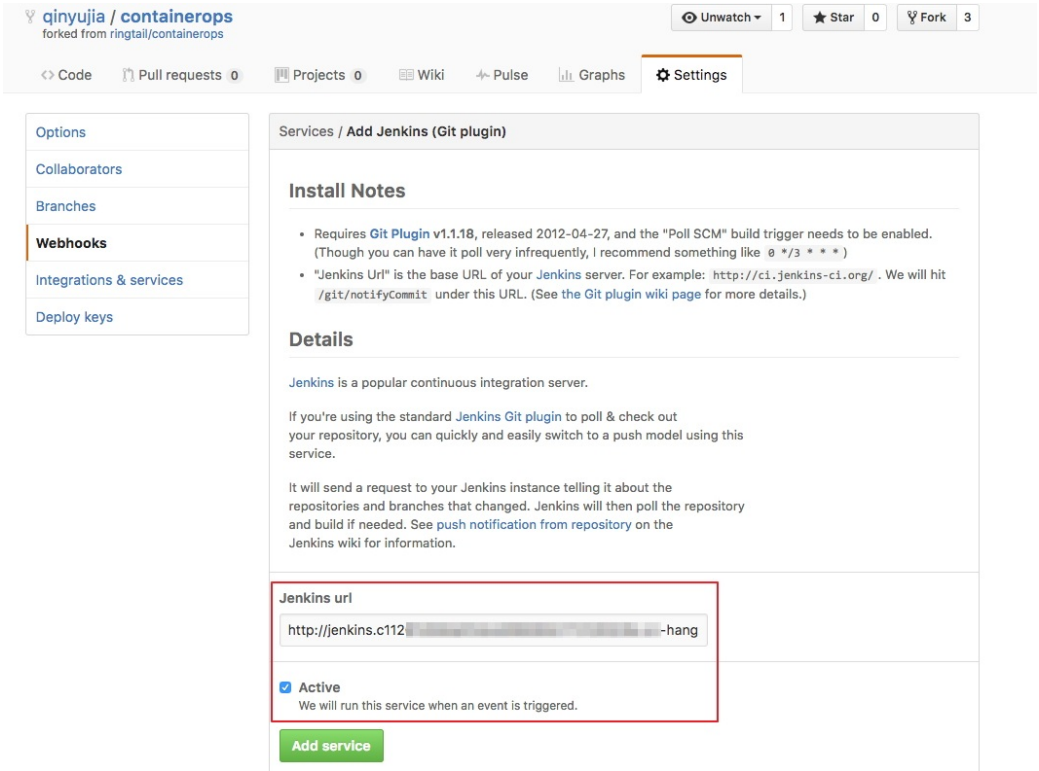
4. Configure the build trigger. In this example, automatically trigger project execution by combining GitHub Webhooks & services.



5. Add the Jenkins service hook to GitHub to implement automatic triggering.

   On the GitHub project home page, click the **Settings**. Click **Webhooks & services**, click **Add Service**, and then select **Jenkins(Git plugin)** from the drop list. In the dialog box of Jenkins hook url, enter `${Jenkins IP}/github-webhook/`. For example:

```
http://jenkins.cd****************.cn-beijing.alicontainer.com/github-webhook/
```



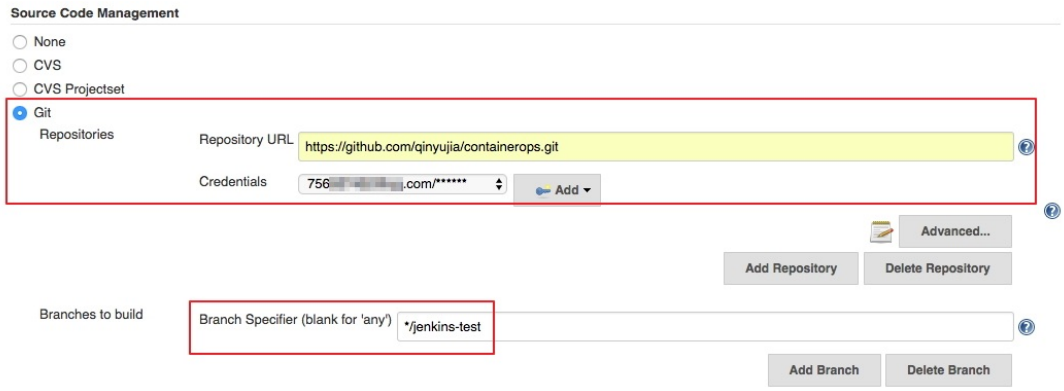
6. Add a build step of Execute shell type and write shell scripts to perform the test.

The commands in this example are as follows:

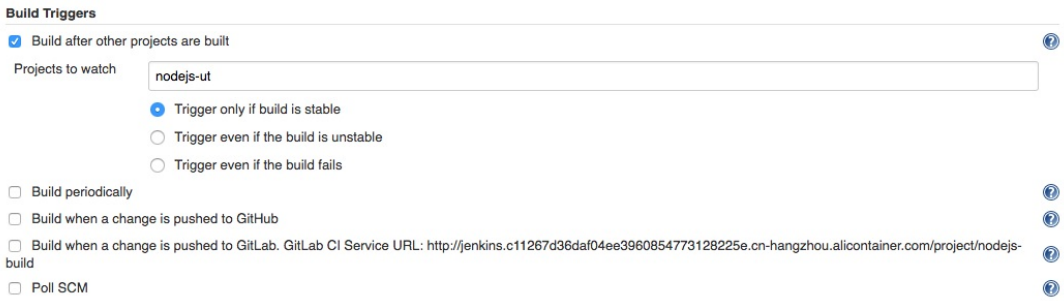
```
pwd
 ls
 cd chapter2
 npm test
```

SVN source code example:

Select **Subversion** in Source Code Management and enter the SVN repository address in the Repository URL field (if the Jenkins master and SVN server are in different time zones, add `@HEAD` at the end of the repository address). Add the username and password of the SVN server in Credentials .



Configure the build trigger. In this example, Post-commit hook is used to automatically trigger the project execution. Enter your configured token in Token Name .



Log on to the SVN server. Create a *post-commit* file in the *hooks* directory of the code repository (svn-java-demo).

```
cd /home/svn/svn-java-demo/hooks
cp post-commit.tmpl post-commit
chmod 755 post-commit
```

Add the curl -u ${Jenkins_account}:${password}

```
${Jenkins_url}/job/svn/build?
        token=${token}  command
```

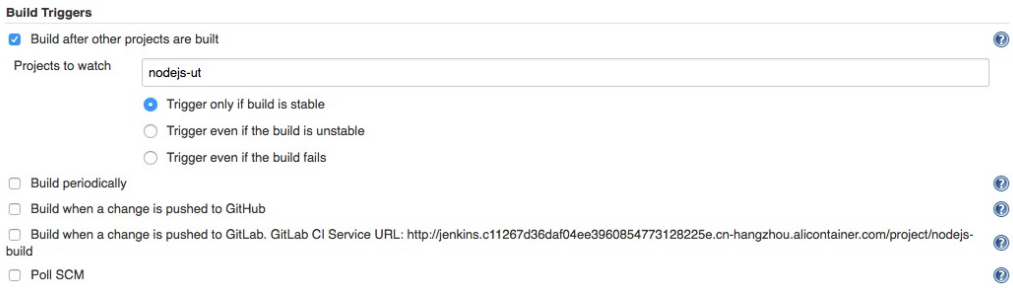
in the <g id="1">post-commit</g> file. For example:

```
curl -u test:test
      http://127.0.0.1:8080/jenkins/job/svn/build?token=qinyujia
```

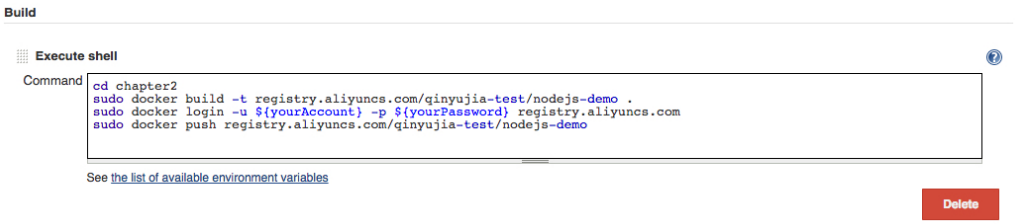
.

2.3 Create a project to automatically build and push images

1. Go back to the Jenkins home page. Click New Item in the left-side navigation pane. Enter the item name, select Freestyle project, and then click OK.
2. Enter the project name and select a node for running the project. In this example, enter the slave-nodejs-ut node prepared in the preceding section.
3. Configure the source code management and code branch. In this example, use GitHub to manage source codes.
4. Add the following trigger and set to automatically build the image only after the unit test is successful.



5. Write the shell script for building and pushing images.



The commands in this example are as follows:

```
cd chapter2
 sudo docker build -t registry.aliyuncs.com/qinyujia-test/nodejs-demo .
 sudo docker login -u ${yourAccount} -p ${yourPassword} registry.aliyuncs.com
 sudo docker push registry.aliyuncs.com/qinyujia-test/nodejs-demo
```

Step 3 Automatically redeploy the application

3.1 Deploy the application for the first time

Use the orchestration template to deploy the image created in step 2.3 to Container Service and create the nodejs-demo application.

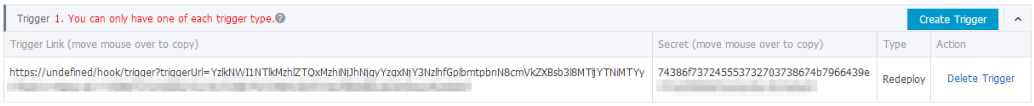
Example:

```
express:
image: 'registry.aliyuncs.com/qinyujia-test/nodejs-demo'
expose:
    - '22'
    - '3000'
restart: always
labels:
    aliyun.routing.port_3000: express
```

3.2 Automatic redeployment

1. Select the created application **nodejs-demo** and create the trigger.

   > **Note** For how to create a trigger, see Triggers.

2. Add a line to the shell script in 2.3. The address is the trigger link of the created trigger.

   ```
   curl 'https://cs.console.aliyun.com/hook/trigger?triggerUrl=***==&secret=***'
   ```

3. Change the command in the example of 2.3 as follows:

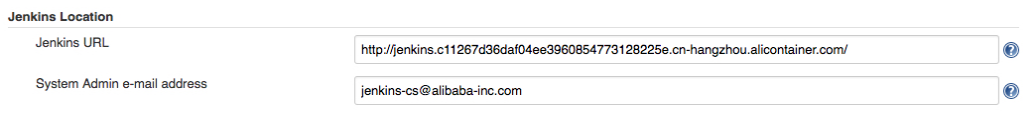
   ```
   cd chapter2
   sudo docker build -t registry.aliyuncs.com/qinyujia-test/nodejs-demo .
   sudo docker login -u ${yourAccount} -p ${yourPassword} registry.aliyuncs.com
   sudo docker push registry.aliyuncs.com/qinyujia-test/nodejs-demo
   curl 'https://cs.console.aliyun.com/hook/trigger?triggerUrl=***==&secret=***'
   ```

After pushing the image, Jenkins automatically triggers the redeployment of the nodejs-demo application.

Step 4 Configure email notification of the results

To send the unit test or image building results to relevant developers or project execution initiators by email, perform the following configurations:

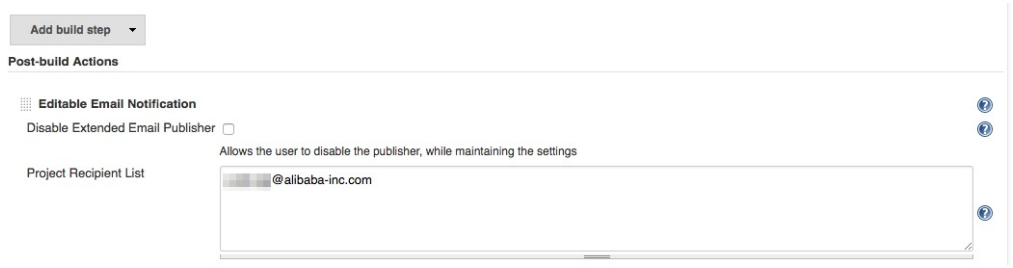
1. On the Jenkins homepage, click Manage Jenkins > Configure System, and configure the Jenkins system administrator email.

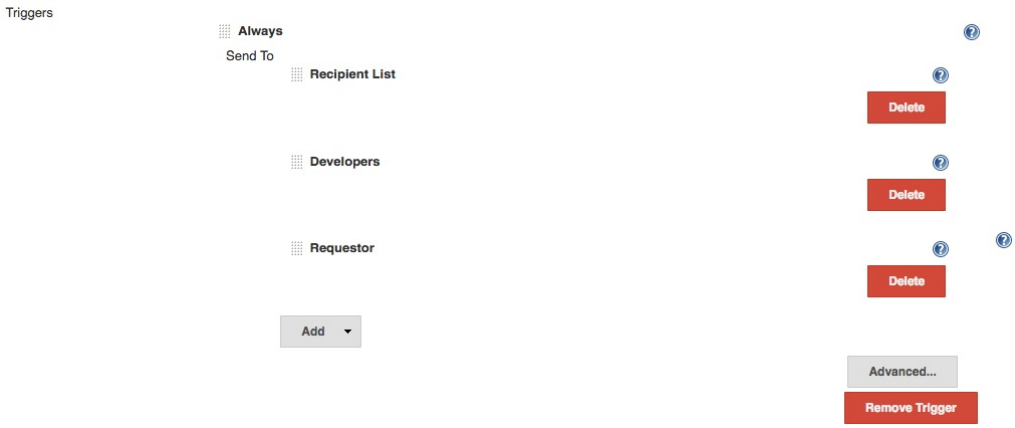
2. Install the Extended Email Notification plug-in, configure the SMTP server and other relevant information, and then set the default email recipient list, as shown in the following figure:

The preceding example shows the parameter settings of the Jenkins application system. The following example shows the relevant configurations for Jenkins projects whose results are to be pushed by email.

3. Add post-building steps in the Jenkins project, select Editable Email Notification and enter the email recipient list.



4. Add a trigger to send emails.



16.Service discovery and load balancing

16.1. Overview

Service discovery and Server Load Balancer mainly solves the issue of communication reliability. To guarantee the reliability, Container Service introduces Server Load Balancer. Communication is divided into two types: communication that exposes services and communication between internal services. See the following scenarios for different solutions.

Scenario 1

We recommend that you use simple routing service for simple Layer-7 protocol Server Load Balancer and web service reverse proxy. For more information, see Simple routing - supports HTTP and HTTPS, Simple routing - Configure domain names, and Simple routing - Change HTTP to HTTPS.

Scenario 2

Server Load Balancer distributes the loads evenly to containers with the same functions in Layer-4 protocol Server Load Balancer and services of non-container clusters access the services of containers in container clusters when a traditional architecture is migrated to a container architecture. We recommend that you use Server Load Balancer routing.

Scenario 3

Services in the same cluster need to discover and communicate with each other, and need the Server Load Balancer capabilities. We recommend that you use 集群内服务间路由和负载均衡

Scenario 4

Services in the same cluster need to discover and communicate with each other, but do not need the Server Load Balancer capabilities. We recommend that you use Service discovery between containers.

Scenario 5

Server Load Balancer and service discovery have high customization requirements, such as the support for extensive domain names, custom error page, record accessing logs, selection of backend services based on URL parameter values, and custom HAProxy configuration files. We recommend that you use Custom routing - User guide. For more information, see Custom routing - simple sample.

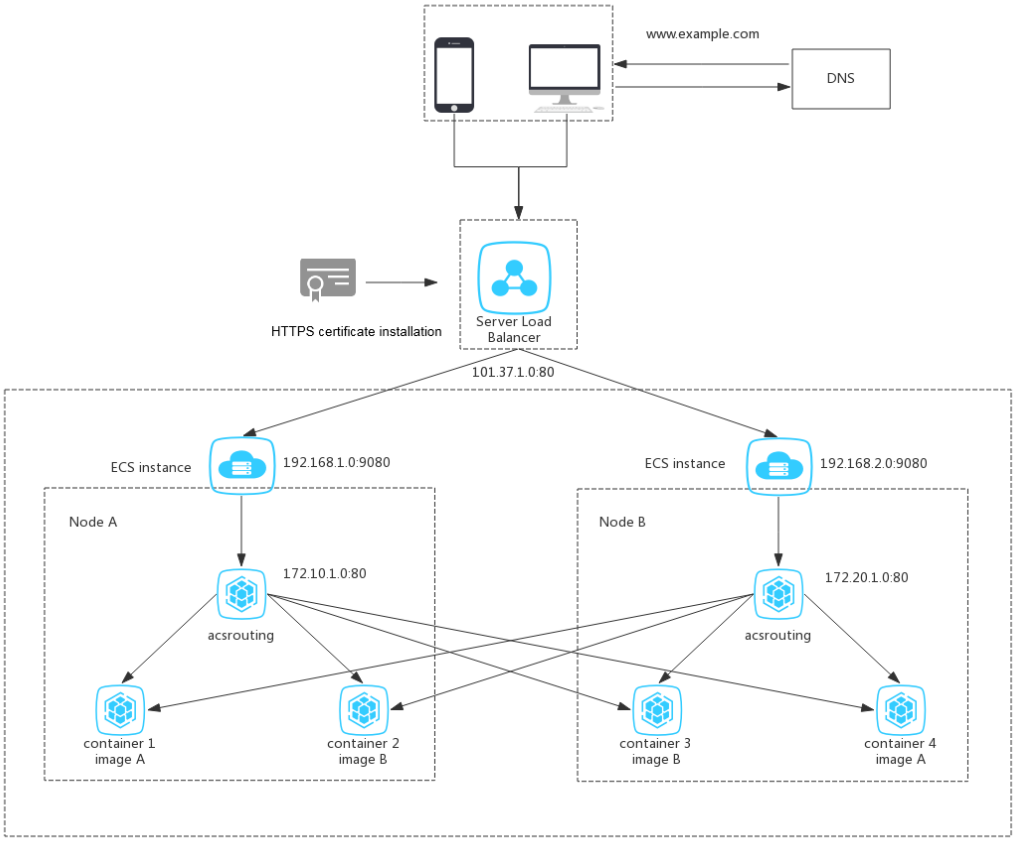
16.2. Simple routing - supports HTTP and HTTPS

Scenarios

Simple Layer-7 protocol load balancingWeb routing service Services in a container cluster access each other using Layer-7 protocol by means of communication proxy and Server Load Balancer

Principles

See the following figure. When you create a cluster, a Server Load Balancer instance is assigned to the cluster by default. The Server Load Balancer instance adds all the nodes in the cluster to the backend. Port 80 is exposed at the frontend, and port 9080 is exposed on the machines of all the backend nodes. Container Service starts a routing application acsrouting, namely, the Alibaba Cloud Container Service Routing. The Server Load Balancer instance adds all the nodes in the cluster to the backend. Port 80 is exposed at the frontend, and port 9080 is exposed on the machines of all the backend nodes. Container Service starts a routing application `acsrouting`, namely, the Alibaba Cloud Container Service Routing. This routing application has only one service, the routing service. The routing service is global, which means a copy of this service (or image), namely, a container, is deployed on each node (a node is also called a host or a virtual machine (VM) instance of Elastic Compute Service (ECS)). The routing service is global, which means a copy of this service (or image), namely, a container, is deployed on each node (a node is also called a host or a virtual machine (VM) instance of Elastic Compute Service (ECS)). is also called a host or a virtual machine (VM) instance of Elastic Compute Service (ECS)). Each node uses this container to route HTTP services or HTTPS services.



As shown in the preceding figure, for HTTP services, the mapping between Server Load Balancer instance frontend and backend ports is `80:9080`, and the port mapping between the host and the container used for routing is `9080:80`, indicating that port 80 is exposed on the containers used for routing. Any ports can be exposed on the other containers used as the web service. After you set the port mapping between host and container during container startup, the routing service can obtain the corresponding port for request routing.

Setup methods

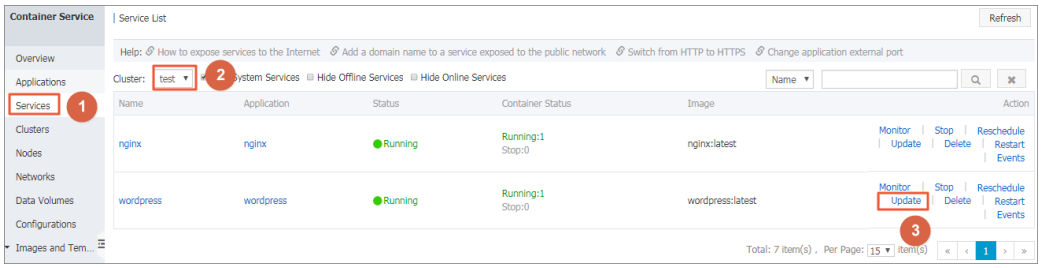
Note Make sure that the following kernel parameters (in the `/etc/sysctl.conf` file) of the related Container Service nodes are set to 0. Otherwise, the nodes might not be accessed.

```
net.ipv4.conf.default.rp_filter = 0
net.ipv4.conf.all.rp_filter = 0
net.ipv4.conf.eth0.rp_filter = 0
```

Set in the Container Service console

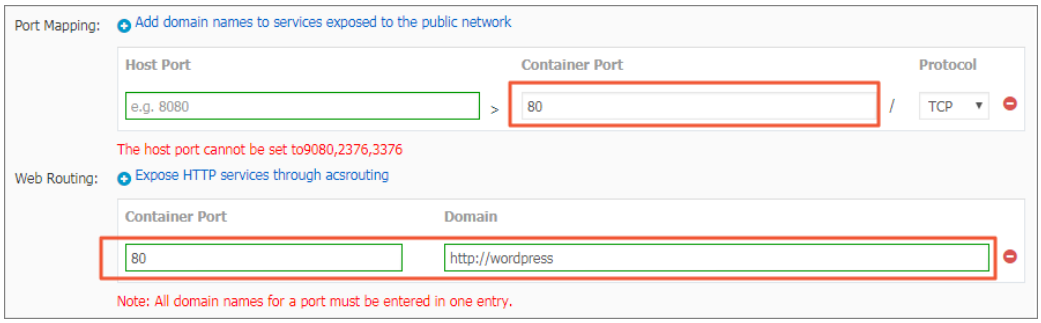
Set by Services > Update

1. Log on to the Container Service console.
2. Log on to the Container Service console.
3. Click **Services** in the left-side navigation pane.
4. Select the cluster in which the service to be exposed resides from the Cluster list.
5. Click **Update** at the right of the service to be exposed (wordpress in this example).



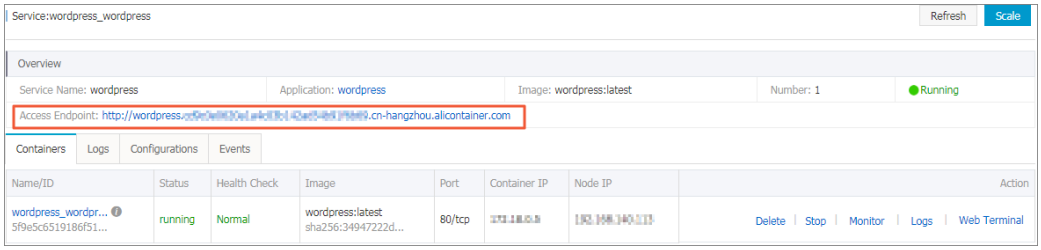
6. Configure the port mapping between host and container on the Update Service page.

   The host port is empty, indicating that a random port on the host is exposed (when HTTP or HTTPS port 80 of the wordpress service to provide the HTTP service. The protocol used is TCP. services are exposed, you do not need to know what port is exposed on the host, because the container port can be directly accessed by using an overlay network or Virtual Private Cloud (VPC) network). The container port is 80. Use port 80 of the wordpress service to provide the HTTP service. The protocol used is TCP.

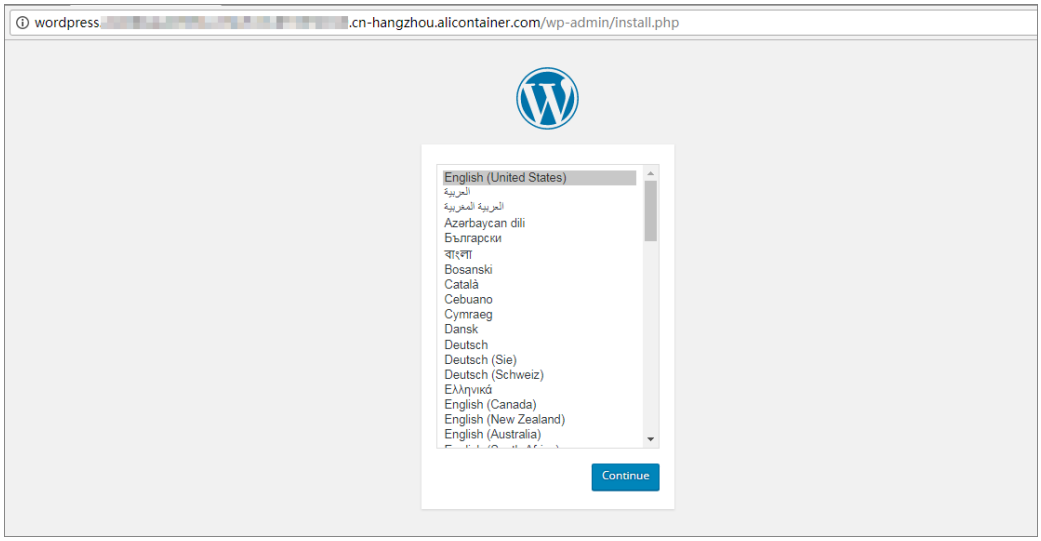


Routing configuration exposes the service by using a domain name. Specify the port to be exposed (port 80 of the wordpress service in this example). Enter the domain name prefix in the Domain field. If the domain name prefix is XXX, the domain name XXX.$cluster_id.$region_id.alicontainer.com is obtained for testing. In this example, the domain name wordpress.cb668bde43f054cd7bd515c8739f38310.cn-hangzhou.alicontainer.com is obtained. You can enter your own domain name, which needs to add the resolution to the IP address of the corresponding Server Load Balancer instance. XXX, the domain name `XXX.$cluster_id.$region_id.alicontainer.com is obtained for` is obtained for testing. In this example, the domain name `wordpress.cb668bde43f054cd7bd515c8739f38310.cn-hangzhou.alicontainer.com` is obtained. You can enter your own domain name, which needs to add the resolution to the IP address of the corresponding Server Load Balancer instance. For how to configure the container port used for routing and the domain name of HTTP services, see routing.

7. Click **Update** after completing the configurations. Click the service name on the **Service List** page and then click the access endpoint

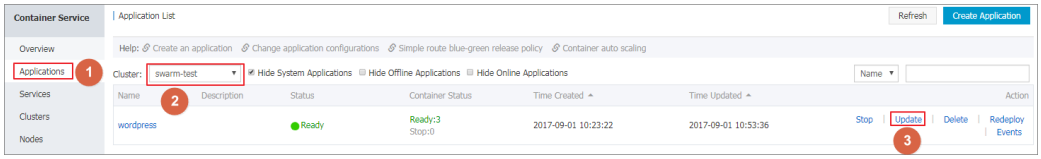


8. to access the wordpress page. The wordpress welcome page appears.

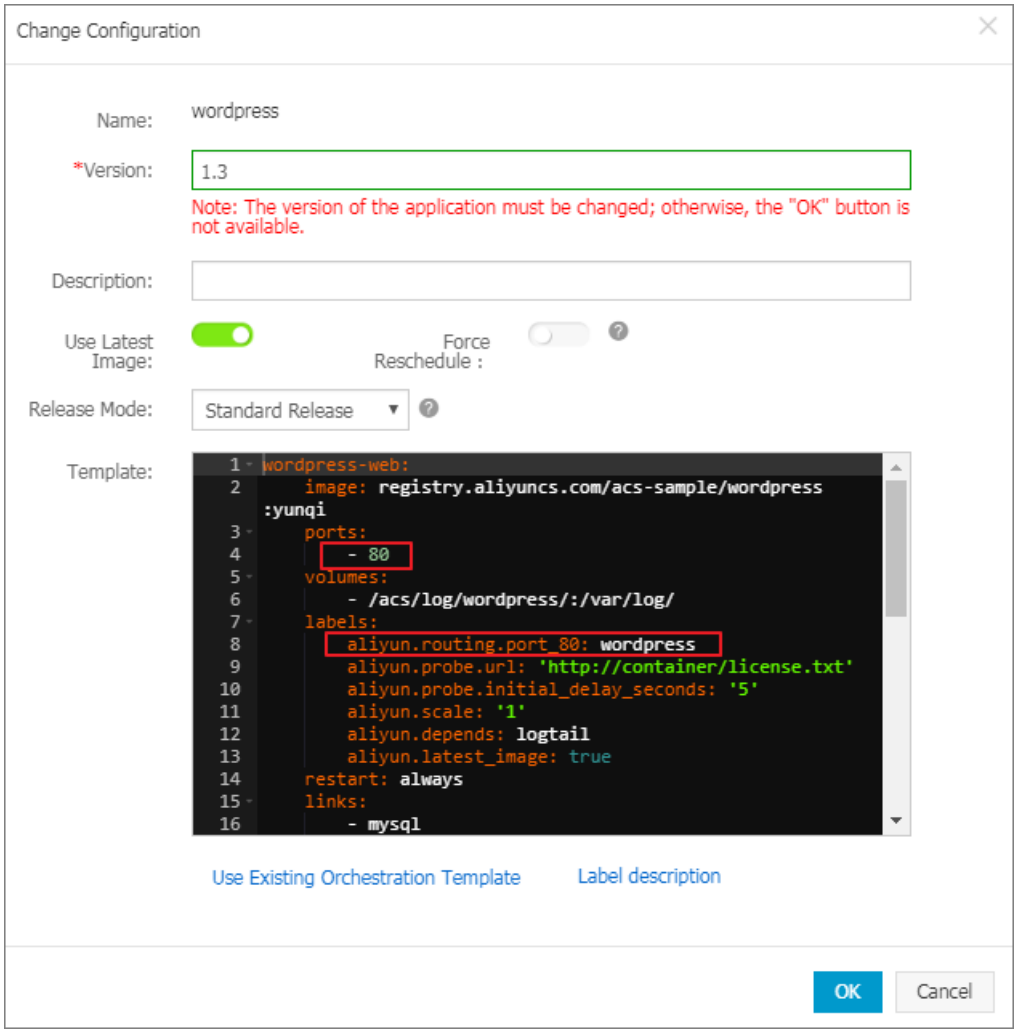


Set by application template editor

1. Log on to the Container Service console.
2. Log on to the Container Service console.
3. Click **Applications** in the left-side navigation pane.
4. Select the cluster from the Cluster list.
5. Click **Update** at the right of the application (wordpress in this example).



6. Add a routing label in Template, define the corresponding domain name or domain name prefix, update the application version, and confirm whether or not to pull the latest Docker image. routing corresponding domain name or domain name prefix, update the application version, and confirm whether or not to pull the latest Docker image. Then, click **OK** to update the domain name.



7. Click the application name on the Application List page and then click the Routes tab. Click the **Route address** to access the wordpress welcome page.

Set by client

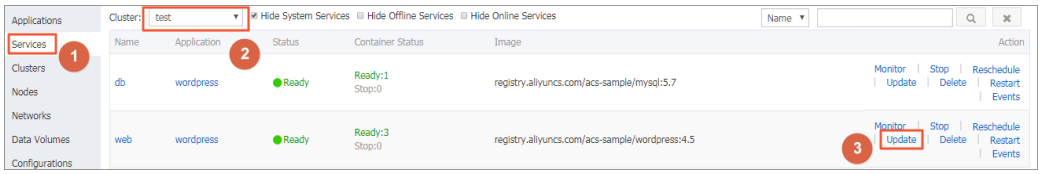
- docker help run: View the used "-p" option. Set the routing configuration in the Container Service console.
- docker-compose: View the supported "ports" option . For routing configuration rules, see routing.

16.3. Simple routing - Configure domain names

Context

Procedure

1. Log on to the Container Service console.
2. Click Swarm > **Services** in the left-side navigation pane.
3. Select the cluster in which the service you want to add a domain name resides from the Cluster drop-down list.
4. Click **Update** at the right of the service you want to add a domain name (the service web of the application wordpress in this example).

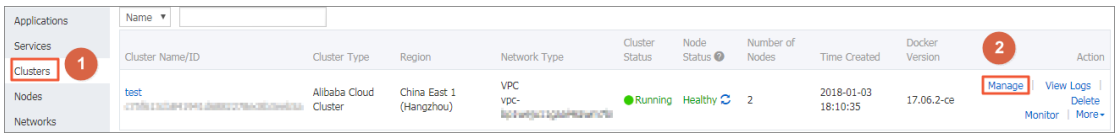


5. Click the plus icon next to **Web Routing**. Enter the domain name to be added (`www.51ili.com` is added in this example), and click **Update** to update the configurations.

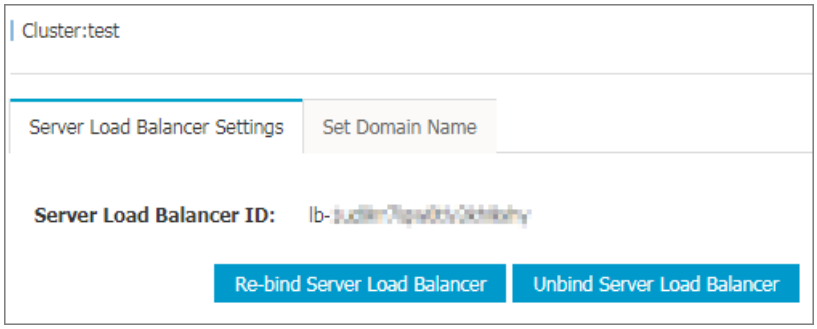
> **Note** To add multiple domain names under the same port for the same service, enter the domain names in one entry and separate them by semicolons (;).

Wait until the service finishes the update and changes to the Ready status. After that, the routing service `acsrouting_routing` finishes configuring the domain name. When requests containing the domain name `www.51ili.com` are sent to access the service `web`, the requests can be correctly resolved and forwarded to the corresponding service.

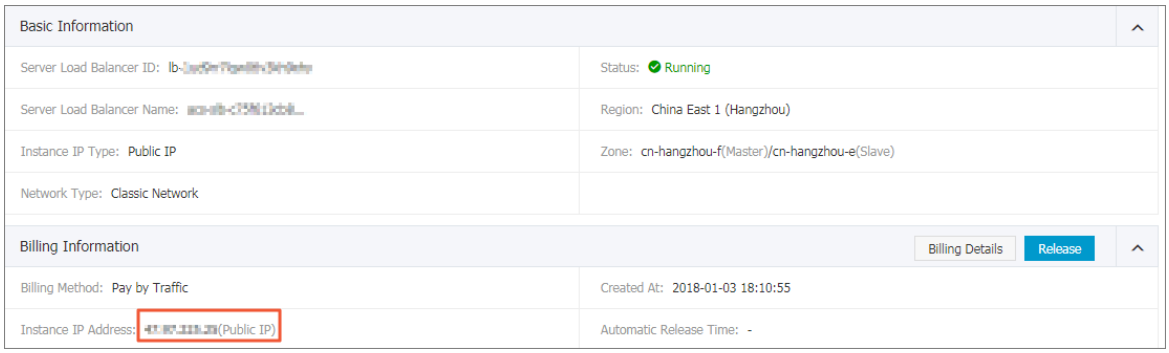
6. Resolve the domain name to the Container Service cluster. When you create a cluster, Container Service assigns a Server Load Balancer instance to the cluster by default. The assigned Server Load Balancer instance only belongs to you.
   i. Log on to the Container Service console.
   ii. Click Swarm > **Clusters** in the left-side navigation pane.
   iii. Click **Manage** at the right of the cluster (`test` in this example).



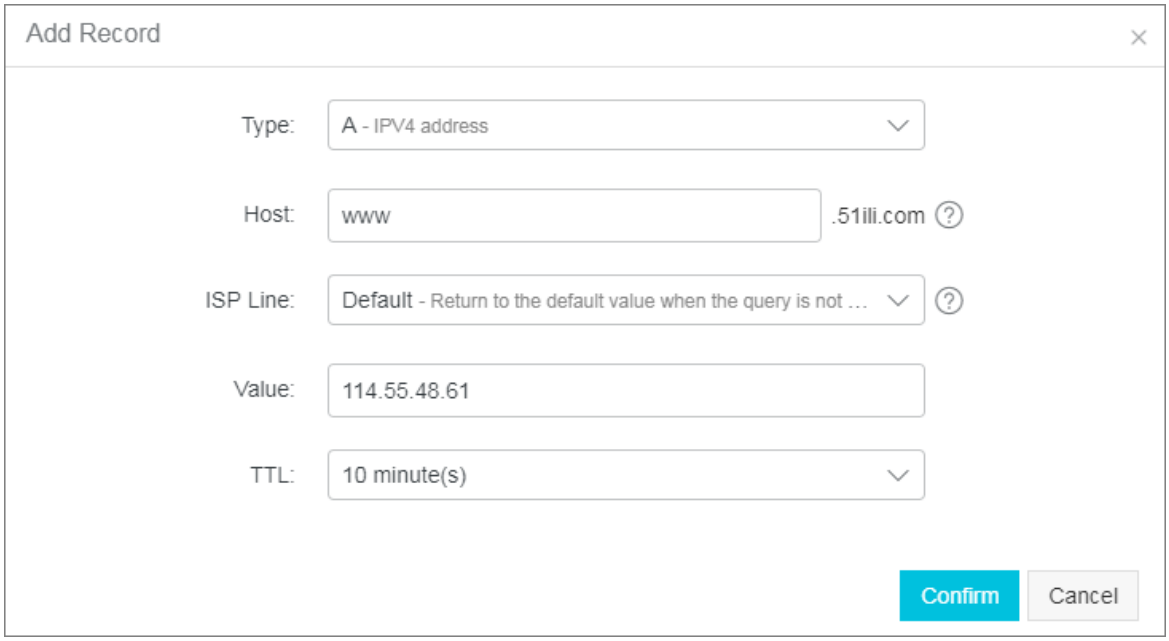
iv. Click **Load Balancer Settings** in the left-side navigation pane and view the Server Load Balancer ID.



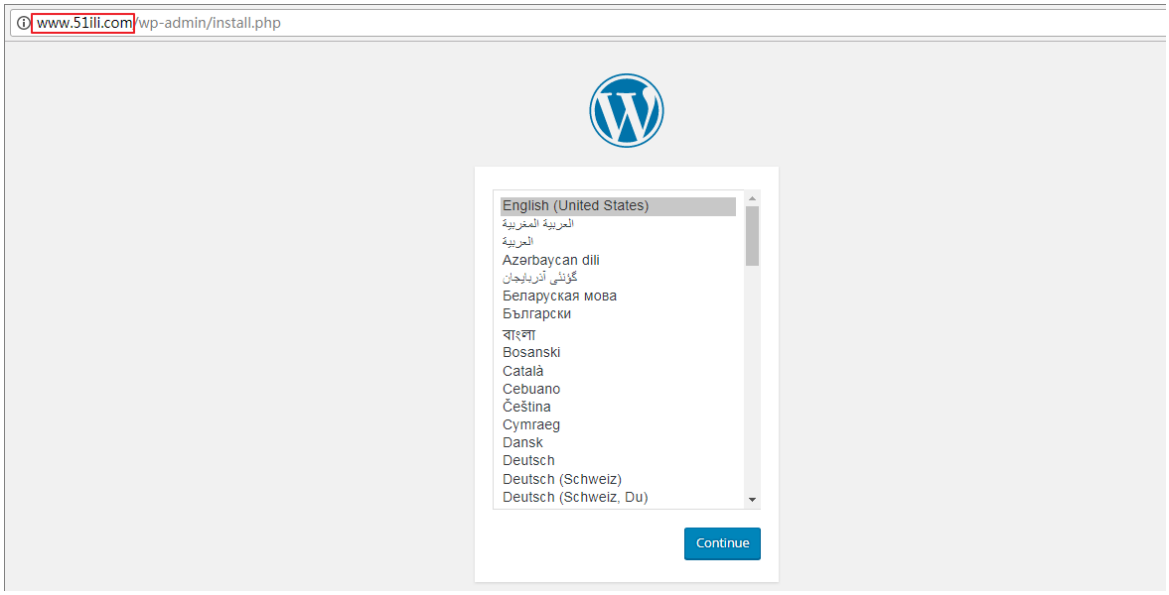
7. Click Products > Server Load Balancer to go to the Server Load Balancer console. View the **IP address** of the Server Load Balancer instance according to the instance ID.



8. Log on to the Alibaba Cloud DNS console and add a DNS record (www.51ili.com in this example).
   i. Add a domain name. Skip this step if the domain name already exists.
   ii. Add a DNS record.
   - Type: Select A - IPV4 address.
   - Host: Enter www. This is the domain name prefix. You can also enter another prefix.
   - ISP Line: Select Default.
   - Value: Enter the IP address of the bound Server Load Balancer instance.
   - TTL: Select the time.



9. Access `www.51ili.com` .



16.4. Simple routing - Change HTTP to HTTPS

Prerequisites

The HTTP domain name access has been configured. For more information, see Simple routing - Configure domain names.

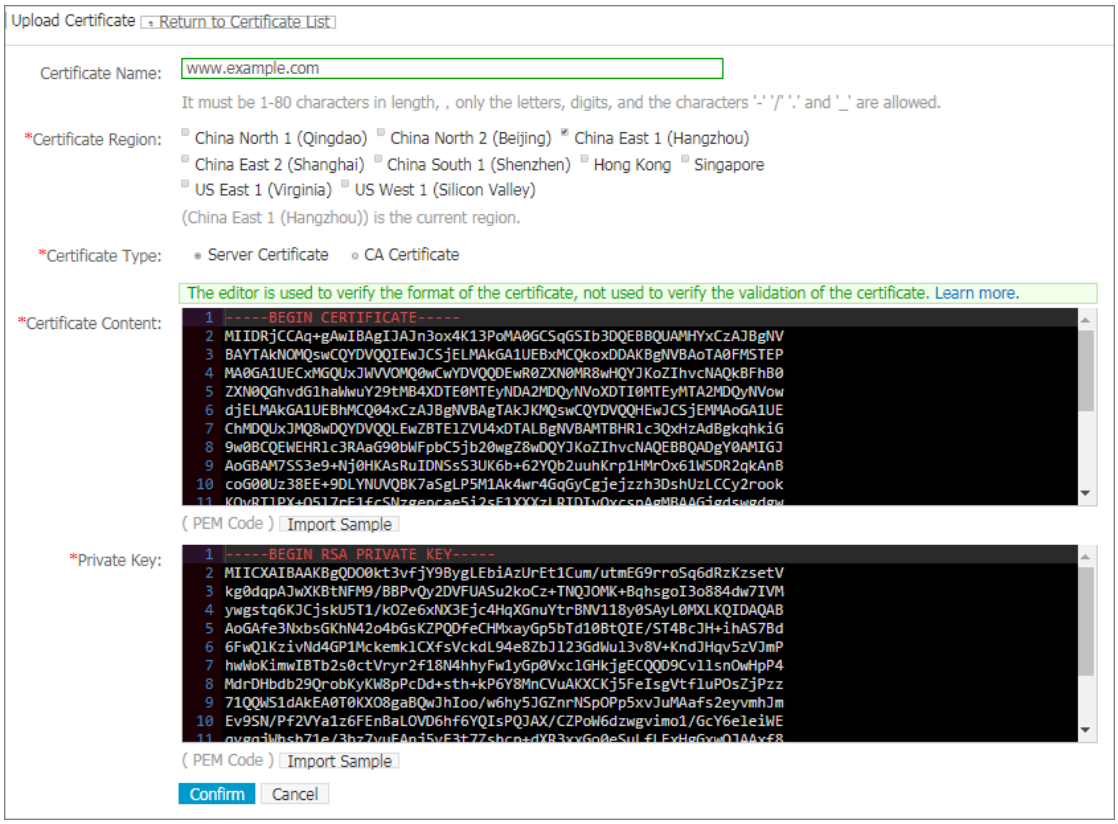
Procedure

1. HTTPS is supported at the Server Load Balancer layer. To support HTTPS, create a Server Load Balancer certificate.

i. Log on to the Server Load Balancer console.

ii. Click **Certificates** in the left-side navigation pane and click **Upload Certificate** in the upper-right corner.

iii. Enter the certificate information and click Confirm.

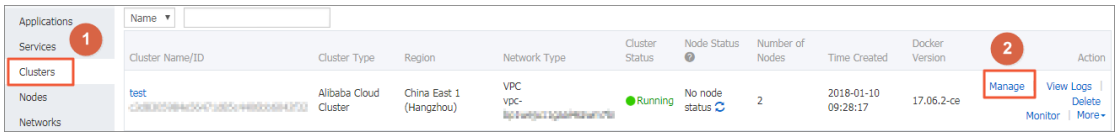


2. After the certificate is successfully created, locate the Server Load Balancer instance that is assigned during cluster creation.

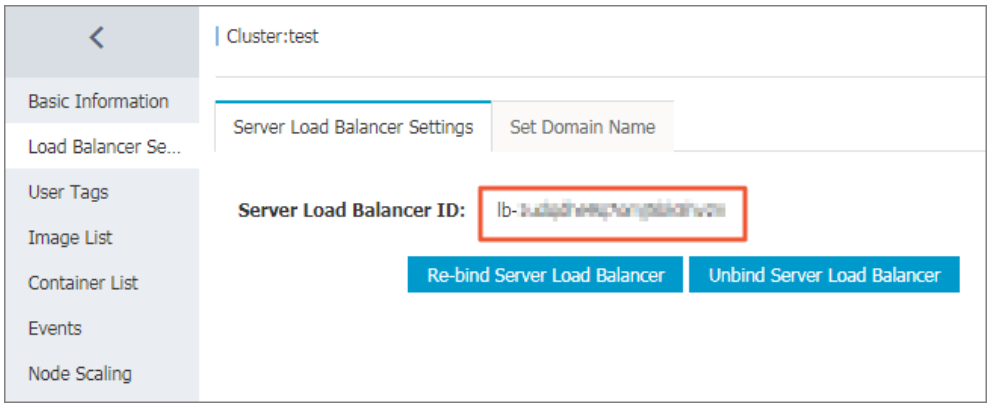
   When you create a cluster, Container Service assigns a Server Load Balancer instance to the cluster, and the instance only belongs to you.

   i. Log on to the Container Service console.

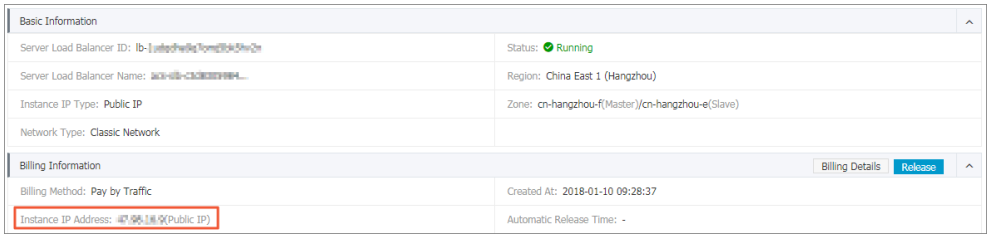
   ii. Click **Clusters** in the left-side navigation pane. Click **Manage** at the right of the cluster (test in this example).



iii. Click **Load Balancer Settings** in the left-side navigation pane and view the Server Load Balancer ID.



Click **Products** > Server Load Balancer to go to the Server Load Balancer console. View the **IP address** of the Server Load Balancer instance according to the instance ID.



3. Click **Listeners** in the left-side navigation pane and click **Add Listener**. The Add Listener dialog box appears. Enter the port information as follows:

```
+----------------+-------+------+
 | | Protocol | Port |
 +----------------+-------+------+
 | Frontend protocol (port) | HTTPS | 443 |
 +----------------+-------+------+
 | Backend protocol (port)  | HTTP  | 9080 |
 +----------------+-------+------+
```

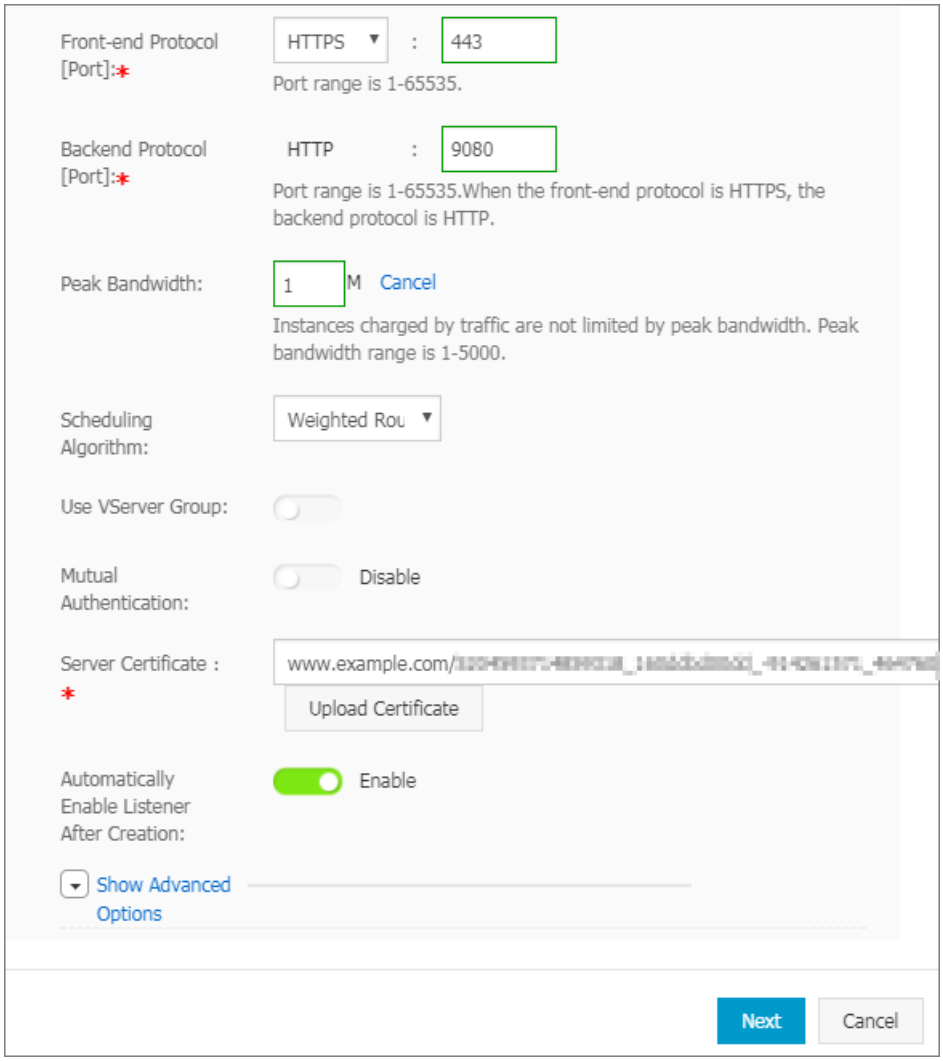
i. Select `HTTPS` for the frontend protocol.

ii. Set the frontend port to 443 and backend port to 9080 (port 9080 is exposed by the routing service **acsrouting_routing** on each Elastic Compute Service (ECS) host. According to the HTTP `HOST` header, all the HTTP requests are forwarded on the routing service **acsrouting_routing** to corresponding containers that provide various services).

iii. Select the preceding certificate `www.example.com`.

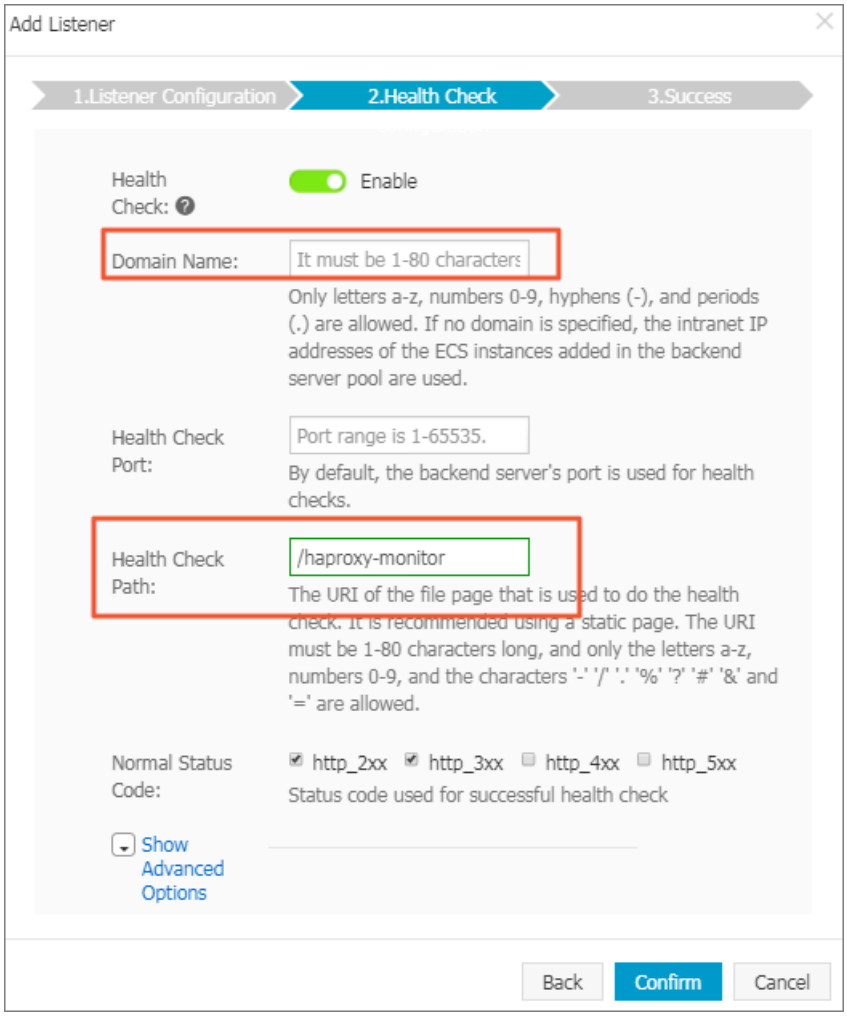
iv. Complete other settings based on your needs.

v. Click **Next**.

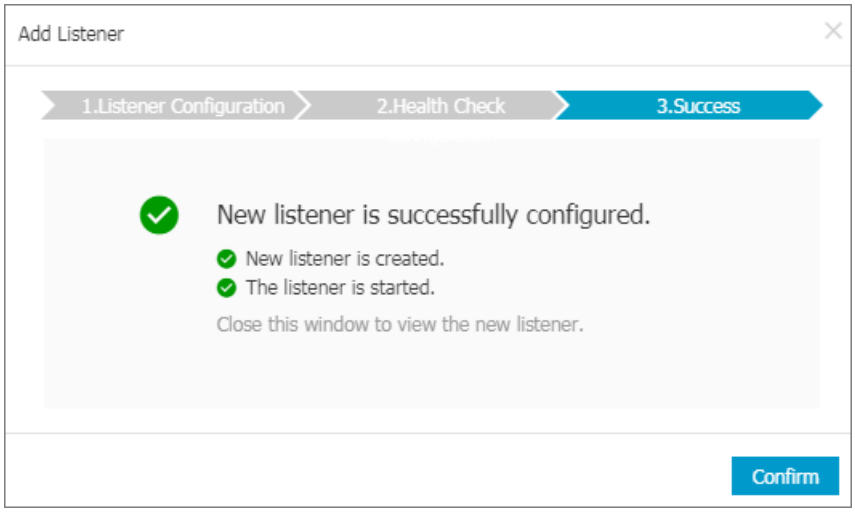


4. Complete the settings in the **Health Check Configuration** step and then click **Confirm**.

   You can select to disable or enable the health check. To enable the health check, enter your own domain name in the **Domain Name** field or enter `/haproxy-monitor` in the **Health Check Path** field. Otherwise, the health check reports an exception.



5. Click **Confirm** after completing the configurations.



6. Access the page `https://www.example.com`.

What's next

After the preceding configurations, to directly redirect to `https://www.example.com` after accessing `http://www.example.com`, see Simple routing - Force redirect from HTTP to HTTPS.

16.5. Simple routing - Force redirect from HTTP to HTTPS

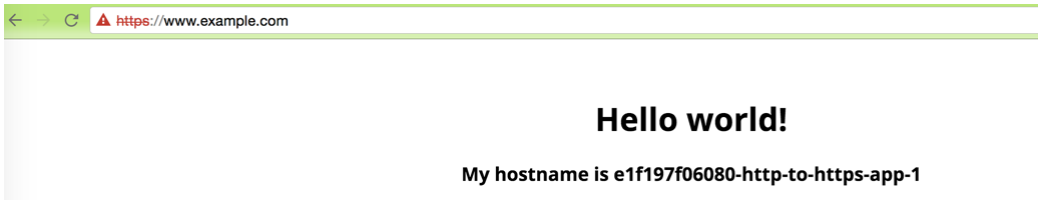
Step 1. Implement the HTTPS protocol to access the helloworld application

1. You can create a "hello world" application by using an orchestration template.

   The application template sample is as follows:

```
app:
     ports:
         - 80/tcp
     image: 'registry.cn-hangzhou.aliyuncs.com/linhuatest/hello-world:latest'
     labels:
         # http/https/ws/wss protocol here
         aliyun.routing.port_80: "http://www.example.com"
     restart: always
```

2. After configuring the Server Load Balancer, access the HTTPS website according to Simple routing - Change HTTP to HTTPS Simple routing - change HTTP to HTTPS.



Step 2. Configure the Nginx container to implement a Forced Jump to HTTPS

1. You can configure to force the HTTP request to redirect to HTTPS.

   The following example configures an Nginx container and adds the rewrite rules to the configuration file, namely, when request http://www.example.com is received, 301 is returned and the request rewrite rules to the configuration file, namely, when request the configuration file, namely,

   when request `http://www.example.com` is received, 301 is returned and the request `301` is automatically redirected to is automatically redirected to `www.example.com`.

   - Log on to each machine in the cluster. Create the Nginx configuration file /ngx/nginx.conf, which will be mounted to the Nginx container as a volume. which

```
will be mounted
 to the Nginx container
as a volume.
```

   - The */ngx/nginx.conf* is as follows:

```
user nginx;
error_log /var/log/nginx/error.log warn;
pid /var/run/nginx.pid;
events {
  worker_connections 65535;
http {
  include /etc/nginx/mime.types;
  default_type application/octet-stream;
  log_format main '$remote_addr - $remote_user [$time_local] "$request" '
                  '$status $body_bytes_sent "$http_referer" '
                  '"$http_user_agent" "$http_x_forwarded_for"';
  access_log /var/log/nginx/access.log main;
  keepalive_timeout 65;
  gzip on;
  server {
      listen 80;
      server_name localhost;
      return 301 https://$host$request_uri;
```

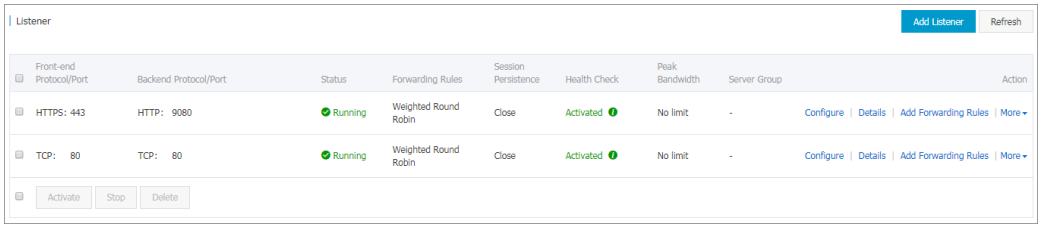
2. Create an Nginx application by using an orchestration template.

   The Nginx sample is as follows:

```
nginx:
     ports:
       - 80:80/tcp # Map to port 80 of the host.
     Image: 'nginx: Latest'
     labels:
       aliyun.global: true # Deploy an Nginx container on each machine to guarantee the
high availability.
     volumes:
       - /ngx/nginx.conf:/etc/nginx/nginx.conf
     restart: always
```

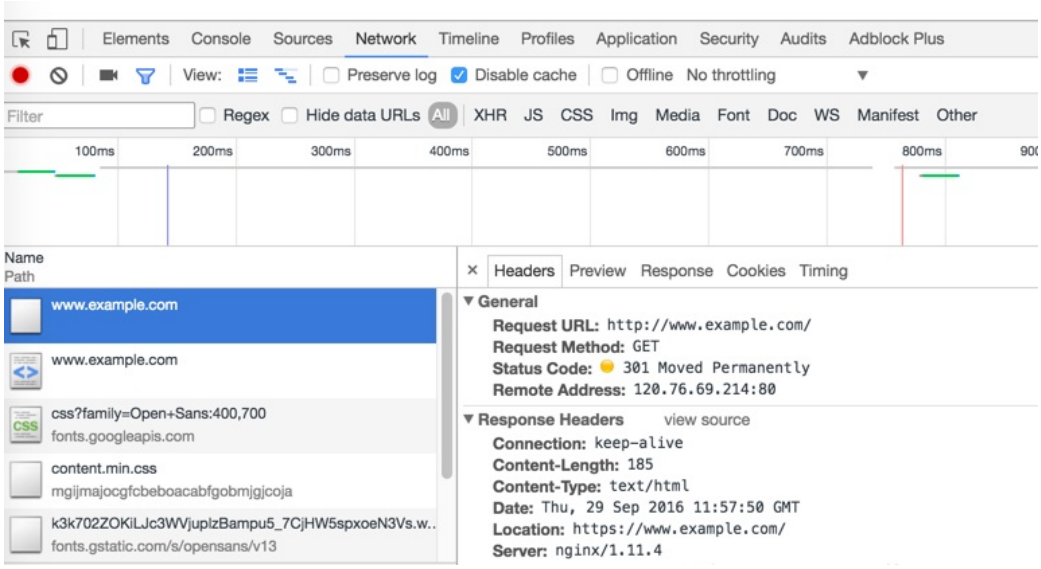
3. Configure the listening rules for the Server Load Balancer of the cluster as follows (frontend port 80 > backend port 80, namely, Server Load Balancer frontend port 80 > backend Elastic Compute Service (ECS) instance port

   as follows (frontend port 80 > backend port 80, namely, Server Load Balancer frontend port 80 > backend Elastic Compute Service (ECS) instance port 80):



4. Verify if the HTTP request is forced to redirect to HTTPS.

   When you access `http://www.example.com`, This means the request is correctly redirected to https://www.example.com. the returned HTTP is as follows. This means the request is correctly redirected to `https://www.example.com`.



16.6. Server Load Balancer routing

Expose HTTP or HTTPS services

We recommend that you use simple routing service (namely, routing) to expose HTTP or HTTPS services. To build your own routing link, activate a new intranet or Internet Server Load Balancer instance routing to the virtual machine (VM) port by using the Alibaba Cloud extension label lb, and configure the mapping between host and container to route requests.

Scenarios:

In Layer-7 protocol Server Load Balancer, a route is customized for each service. Services of non-container clusters access the services in container clusters when a traditional architecture is migrated to a container architecture.

Expose TCP or UDP services

Currently, to expose TCP services, configure a Server Load Balancer instance or a public IP address, and configure the port mapping between host and container by using the Alibaba Cloud extension label lb.

> **Note** To use Server Load Balancing routing, you need to buy a new Server Load Balancer instance. Multiple services cannot share the same Server Load Balancer instance, and the default Server Load Balancer instance of the cluster cannot be used for Server Load Balancer routing.

Scenarios:

In Layer-4 protocol Server Load Balancer, a route is customized for each service. Services of non-container clusters access the services in container clusters when a traditional architecture is migrated to a container architecture.

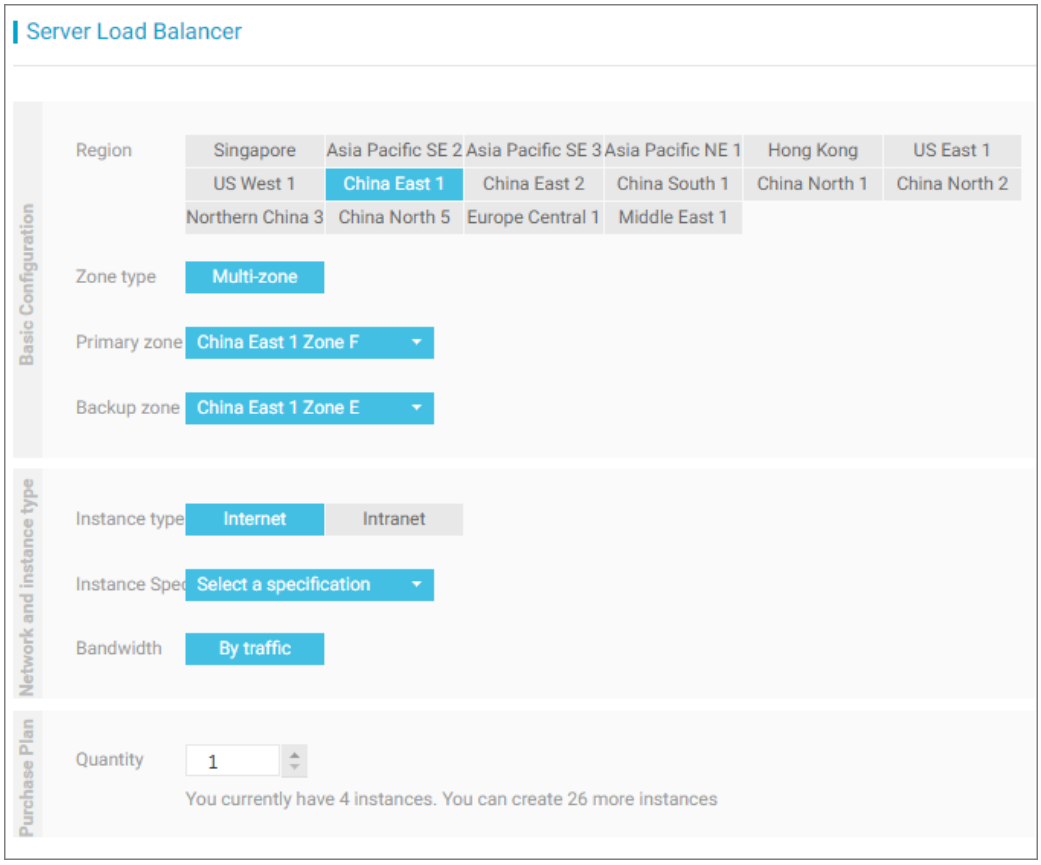
Example:

Expose the Redis service in a container cluster to the Python application outside the container cluster by customizing a Server Load Balancer instance.

1. In the Server Load Balancer console, click **Create Server Load Balancer** in the upper-right corner to purchase and create a Server Load Balancer instance used for routing.

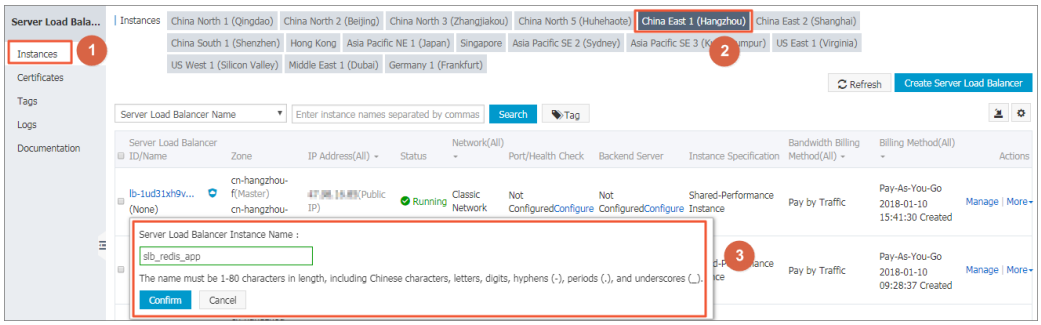
Select Internet as the Instance type in this example. Select Internet or Intranet as the Instance type based on your needs.

> **Note** Server Load Balancer does not support cross-region deployment. Therefore, select the same region in which your Container Service cluster resides.



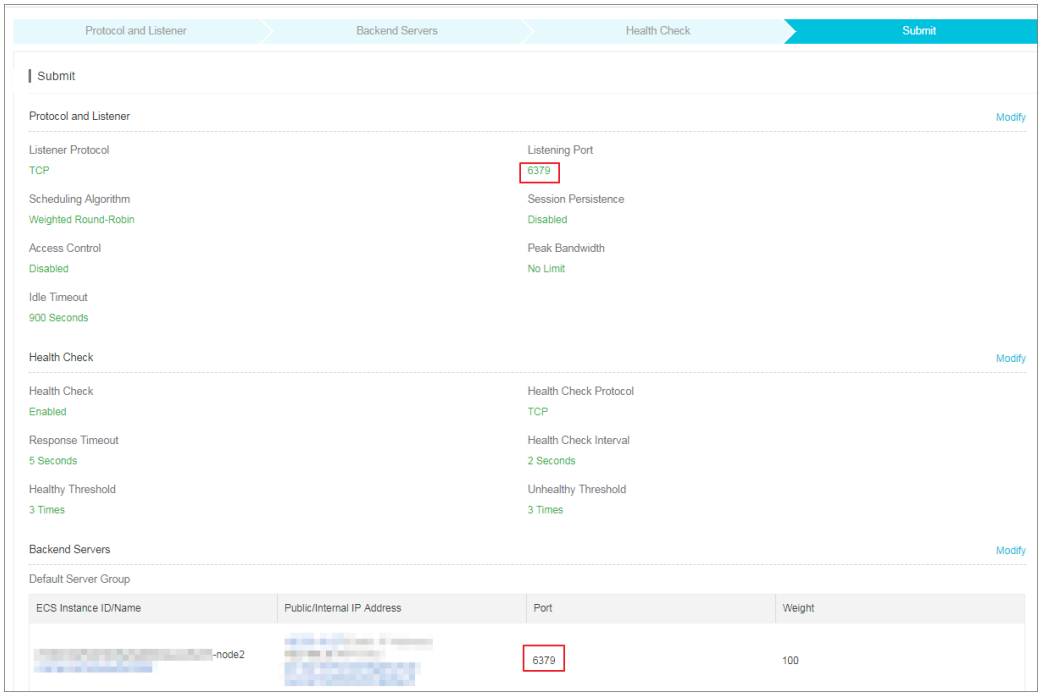
2. Return to the Server Load Balancer console and name the created Server Load Balancer instance as slb_redis_app. Container Service can use this name or the instance ID to reference the Server Load Balancer instance.

   Click **Instances > Server Load Balancer** in the left-side navigation pane. Select the region in which the Server Load Balancer instance resides. Edit the instance name and then click **OK**.



3. Create a listening port.

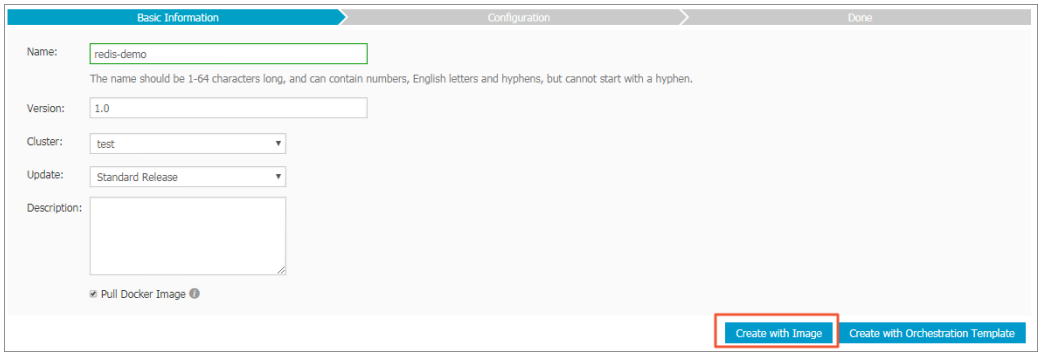
Click **Configure Listener** at the right of the instance. On the **Configure Server Load Balancer** page, configure listening rules as follows: . select TCP as the listener protocol, add a backend server, and configure the port mapping as 6379:6379. As shown in the following figure.



4. Log on to the Container Service console, select an existing cluster, create an application named redis-demo, and click **Create by Image**.

   For how to create an application, see Create an application.

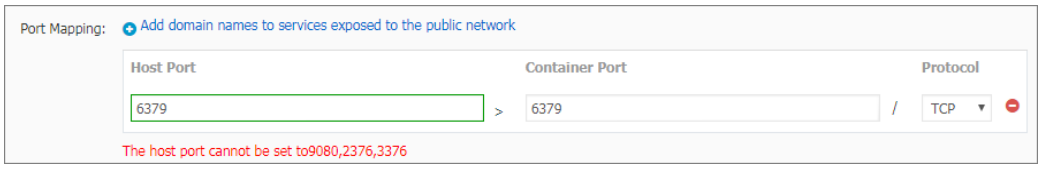
   > **Note** Server Load Balancer does not support cross-region deployment. Therefore, the used Container Service cluster must be in the same region as the Server Load Balancer instance created in the preceding steps.



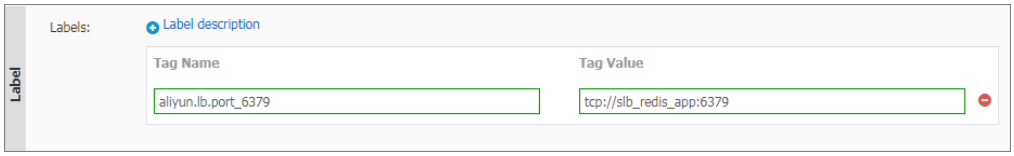
5. Select the redis image and set the **Port Mapping**.

   > **Note** The Redis image only enables port 6379 on the container. To route the created Server Load Balancer instance to this container port, you must specify the port mapping between host and container of the Redis image.

In **Port Mapping**, specify the host port as 6379, which is the backend host port bound to the Server Load Balancer instance, and select TCP as the Protocol.



6. To configure custom Server Load Balancer, let the Redis service know the information of the used Server Load Balancer instance by adding a label to the Redis service or configuring the Load Balancer.
   - Add a label to the service. In this example, the label is `aliyun.lb.port_6379: tcp://slb_redis_app:6379`.

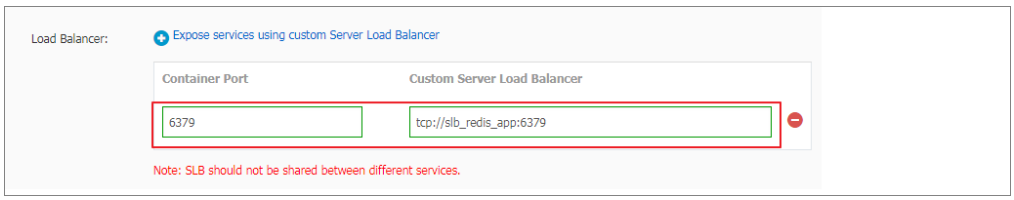
     The label syntax is as follows, where variables with `$` are placeholders.

     ```
     aliyun.lb.port_$container_port:$scheme://$[slb_name|slb_id]:$front_port
     ```

     - `$container_port` indicates the port to be exposed by the container.
     - `$scheme` indicates the protocol supported by the listening port of the Server Load Balancer instance, whose value might be `tcp`, `http`, `https`, or `udp`.
     - `$[slb_name|slb_id]` indicates the name or ID of the Server Load Balancer instance.
     - `$front_port` indicates the frontend port to be exposed by the Server Load Balancer instance.

     For more information, see Alibaba Cloud extension label lb.
   - On the **Create Application** page, click the plus icon next to **Server Load Balancer Routing Configuration** and set the information about the Server Load Balancer instance.

     This setting corresponds to the label `6379: tcp://slb_redis_app:6379`.

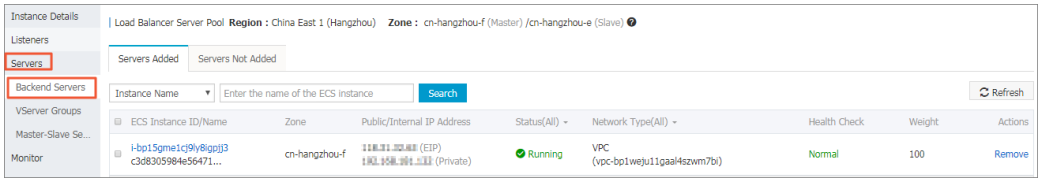
     In this example, set the destination container port as 6379, reference the Server Load Balancer instance name `slb_redis_app`, set the listening port protocol as TCP, which corresponds to the protocol configured in the port mapping between host and container, and set the frontend port of the Server Load Balancer instance as 6379.

     > **Note** In this example, set 6379 as the frontend port and backend port (namely, the host port) of the Server Load Balancer instance and the container port, you can set a different frontend port and host port as per your needs.

7. Click **Create** to create the Redis application. During the creation process, the slb_redis_app Server Load Balancer instance is automatically bound to the backend host deployed with the Redis image.
8. When the Redis application is ready, log on to the Server Load Balancer console to view the status of the Server Load Balancer instance named `slb_redis_app`.

   Click the Server Load Balancer instance ID, and then on the instance details page, click **Default Server Group**.

   The health status shows that the Server Load Balancer instance is correctly bound to the Redis backend.



9. You can view the IP address of the Server Load Balancer instance on the **Instances** page of the Server Load Balancer console, and use the command line tool `telnet $Server_Load_Balancer_IP_address 6379` to check port accessibility.
10. To test the preceding configurations, start a simple Python application locally to access Redis in the container cluster by using the slb_redis_app Server Load Balancer instance.

    > **Note** The Redis host address is the IP address of the Server Load Balancer instance.

    app.py

```
from flask import Flask
from redis import Redis
app = Flask(__name__)
redis = Redis(host='$Server_Load_Balancer_IP_address', port=6379)
@app.route('/')
def hello():
redis.incr('hits')
return 'Hello World! I have been seen %s times.' % redis.get('hits')
if __name__ == "__main__":
app.run(host="0.0.0.0", debug=True)
```

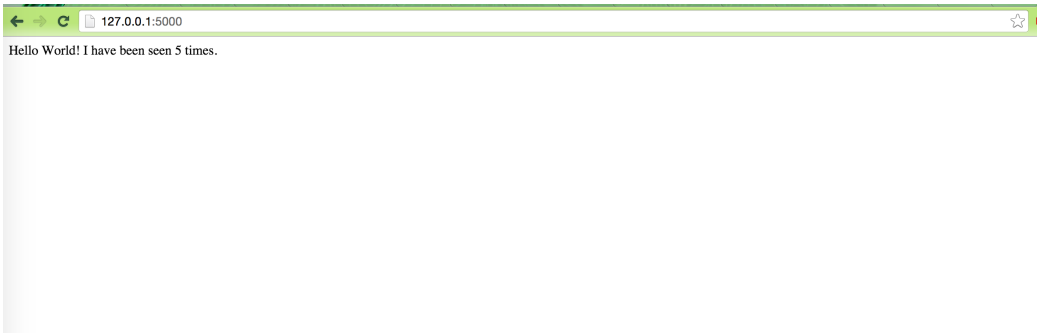
    requirements.txt

```
flask
redis
```

    shell

```
$ pip install -r requirements.txt
$ python app.py
Running on http://0.0.0.0:5000/  ## Press CTRL+C to quit
Restarting with stat
Debugger is active!
Debugger pin code: 243-626-653
```

    The access result is as follows.



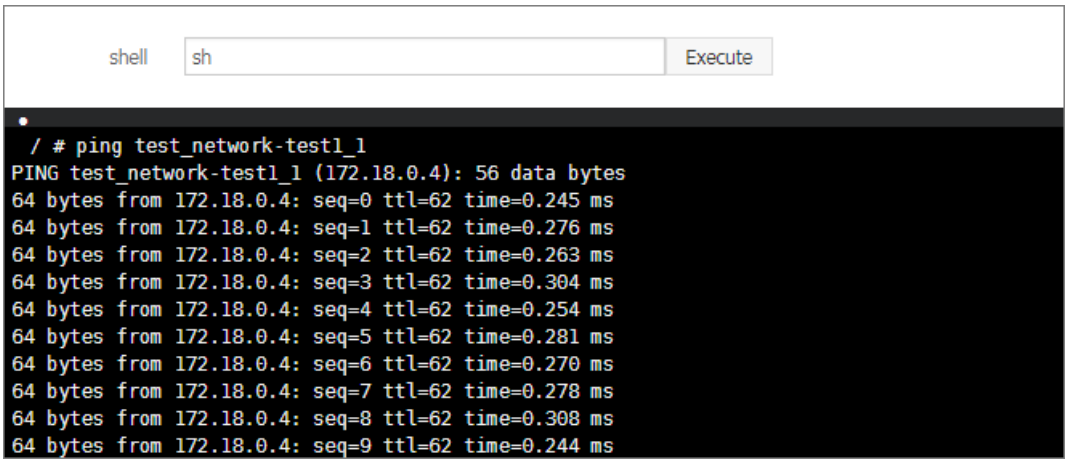
16.7. Service discovery between containers

Container Service provides multiple methods of service discovery for the services and containers in the cluster. The service can be discovered by using the container name, link, or hostname.

Container name

Container Service can be accessed by using the container IP address or the name of another container in the network. In the example described in Container network interconnection , you can access the container `test_network-test1_1` by using its container name in the container `test_network-test2_1` .

If the `container_name` is not specified in the orchestration file, the default container name is `{project-name}_{service-name}_{container-index}` . After connecting to the web terminal of a container, you can access a container of another service by using the container name to test the network interconnection. As shown in the following illustration.



Link

Container Service supports the link between services in an orchestration template. The link between services can link the containers of a service to the containers of another service. In containers, you can access the dependent containers by using the alias of the linked service. When the IP address of the dependent container changes, the alias resolved IP address will be dynamically updated. For a specific example, see the WordPress orchestration in Container Service sample orchestration. The web service in WordPress links the db:mysql service to the containers as follows. Then, the containers can access the containers of the db service by using the MySQL domain name.

```
links:
    - 'db:mysql'
```

Hostname

If the hostname configuration is defined in the orchestration template service, the container can be accessed by using this hostname in the cluster.

For example,

```
testhostname:
        image: busybox
        hostname: xxserver
        command: sleep 100000
        tty: true
```

In the cluster, you can resolve and access the container of this service by using xxserver. For more information, seeContainer network interconnection the orchestration example in Container network interconnection. If this service contains several containers, accessing the containers by using this hostname allows Server Load Balancer to take effect.

If the service does not configure the hostname, Container Service will use the container name as the internal hostname of the container. If an application in the container needs to know the container name for service registration, such as Eureka Client, register an accessible address to Eureka Server. The process in the container can obtain the container name for service registration and enable other service callers to access each other by using the container name.

16.8. Custom routing - simple sample

In this example, an acs/proxy container is deployed, services are exposed by using a Server Load Balancer instance (with the lb label) externally, and an Nginx server is attached at the backend. This example only shows the Nginx homepage, and other functions will be added based on the basic example.

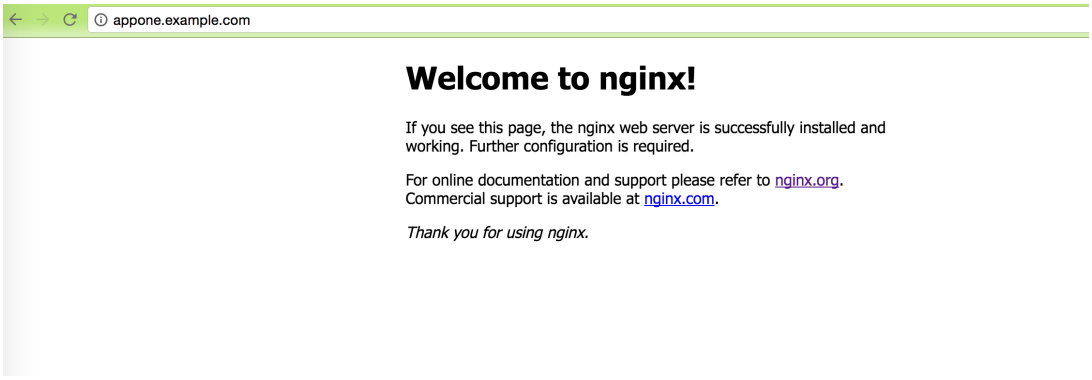
> **Note** Different services cannot share the same Server Load Balancer. Otherwise, the backend machines of Server Load Balancer will be deleted and the services will become unavailable.

Basic example

The compose template is as follows:

```
lb:
    image: registry.aliyuncs.com/acs/proxy:0.6
    ports:
            - '80:80'
    restart: always
    labels:
         # Addon allows the proxy image to function as a subscription registry center and dynamically load the service route.
        aliyun.custom_addon: "proxy"
         # A proxy image container is deployed on each virtual machine (VM).
        aliyun.global: "true"
        # A Server Load Balancer instance is bound to the frontend.
        aliyun.lb.port_80: tcp://proxy_test:80
    environment:
         # Indicates the range of backend containers that support route loading. "*" indicates the whole cluster. By default, it indicates the services in applications.
        ADDITIONAL_SERVICES: "*"
appone:
    expose: # For proxied services, use expose or ports to tell proxy containers which port is to be exposed.
        - 80/tcp
    image: 'nginx:latest'
    labels:
          # http/https/ws/wss are supported. Use your own domain name instead of the test domain name provided by Container Service.
        aliyun.proxy.VIRTUAL_HOST: "http://appone.example.com"
    restart: always
```

After the service is successfully started, the following figure appears.



Enable session persistence

```
lb:
    image: registry.aliyuncs.com/acs/proxy:0.6
    ports:
            - '80:80'
    restart: always
    labels:
         # Addon allows the proxy image to function as a subscription registry center and d
ynamically load the service route.
        aliyun.custom_addon: "proxy"
         # A proxy image container is deployed on each VM.
        aliyun.global: "true"
         # A Server Load Balancer instance is bound to the frontend.
        aliyun.lb.port_80: tcp://proxy_test:80
    environment:
         # Indicates the range of backend containers that support route loading. "*" indica
tes the whole cluster. By default, it indicates the services in applications.
        ADDITIONAL_SERVICES: "*"
appone:
    ports:
        - 80/tcp
        - 443/tcp
    image: 'nginx:latest'
    labels:
         # http/https/ws/wss are supported.
        aliyun.proxy.VIRTUAL_HOST: "http://appone.example.com"
         # Session persistence is enabled, the cookie method is applied, and the key is CON
TAINERID.
        aliyun.proxy.COOKIE: "CONTAINERID insert indirect"
    restart: always
```

Customize 503 page

When the VIP address of the Server Load Balancer instance instead of the domain name is entered, the 503 error page is returned as follows.

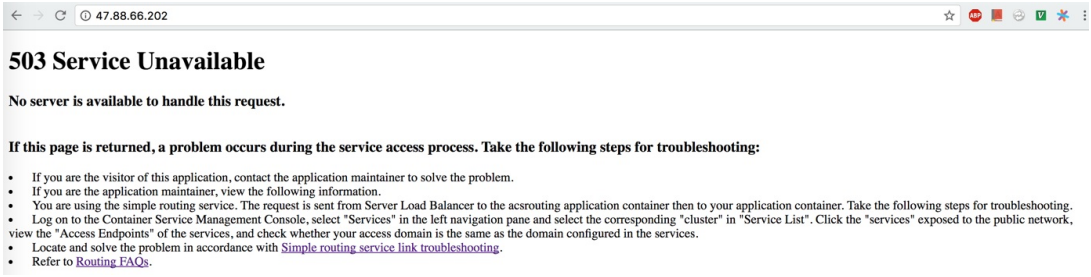
To add messages to the 503 page, add the `/errors` folder to the VM where the container resides and add the `/errors/503.http` file with the following content:

```
HTTP/1.0 503 Service Unavailable
Cache-Control: no-cache
Connection: close
Content-Type: text/html;charset=UTF-8
<html><body><h1>503 Service Unavailable</h1>
<h3>No server is available to handle this request.</h3>
<li>If you are the visitor of this application, contact the application maintainer to solve the problem. </li>
<li>If you are the application maintainer, view the following information. </li>
<li>You are using the simple routing service. The request is sent from Server Load Balancer to the acsrouting application container then to your application container. Follow these steps for troubleshooting. </li>
<li>Log on to the Container Service console. Click "Services" in the left-side navigation pane. Select the corresponding cluster on the "Service List" page. Click the name of the service exposed to the public network. View the "Access Endpoint" of the service, and check whether your access domain name is the same as the domain name configured in the corresponding service. </li>
<li>Locate and troubleshoot the problem. </li>
<li>View Routing FAQs. </li>
<li>If the problem persists, open a ticket and contact the technical staff for help. We will serve you faithfully.</li>
</body></html>
```

You can modify the error page as per your needs. The compose template is modified as follows:

```
lb:
    image: registry.aliyuncs.com/acs/proxy:0.6
    ports:
            - '80:80'
    restart: always
    labels:
        # Addon allows the proxy image to function as a subscription registry center and dynamically load the service route.
        aliyun.custom_addon: "proxy"
         # A proxy image container is deployed on each VM.
        aliyun.global: "true"
         # A Server Load Balancer instance is bound to the frontend.
        aliyun.lb.port_80: tcp://proxy_test:80
    environment:
         # Indicates the range of backend containers that support route loading. "*" indicates the whole cluster. By default, it indicates the services in applications.
        ADDITIONAL_SERVICES: "*"
        EXTRA_FRONTEND_SETTINGS_80: "errorfile 503 /usr/local/etc/haproxy/errors/503.http"
    volumes:
        - /errors/:/usr/local/etc/haproxy/errors/
appone:
    ports:
        - 80/tcp
        - 443/tcp
    image: 'nginx:latest'
    labels:
          # You can specify paths when configuring URLs. In this example, http/https/ws/wss are supported.
        aliyun.proxy.VIRTUAL_HOST: "http://appone.example.com"
    restart: always
```

After entering the VIP address of the Server Load Balancer instance, the 503 page is displayed as follows.

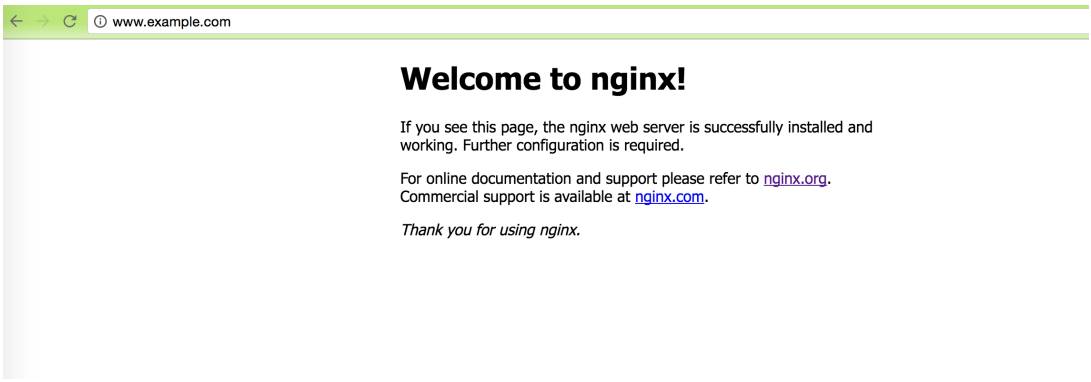


Support extensive domain names

Modify the configurations as follows to enable the backend of Nginx to support extensive domain names (that is, the Nginx homepage can be accessed by using `appone.example.com` and `*.example.com`).

```
lb:
    image: registry.aliyuncs.com/acs/proxy:0.6
    ports:
            - '80:80'
    restart: always
    labels:
        # Addon allows the proxy image to function as a subscription registry center and dy
namically load the service route.
        aliyun.custom_addon: "proxy"
         # A proxy image container is deployed on each VM.
        aliyun.global: "true"
         # A Server Load Balancer instance is bound to the frontend.
        aliyun.lb.port_80: tcp://proxy_test:80
    environment:
         # Indicates the range of backend containers that support route loading. "*" indica
tes the whole cluster. By default, it indicates the services in applications.
        ADDITIONAL_SERVICES: "*"
        EXTRA_FRONTEND_SETTINGS_80: "errorfile 503 /usr/local/etc/haproxy/errors/503.http"
    volumes:
        - /errors/:/usr/local/etc/haproxy/errors/
appone:
    ports:
        - 80/tcp
        - 443/tcp
    image: 'nginx:latest'
    labels:
        # You can specify paths when configuring URLs. In this example, http/https/ws/wss a
re supported.
        aliyun.proxy.VIRTUAL_HOST: "http://*.example.com"
    restart: always
```

Bind a host and enter the domain name `www.example.com`. The Nginx homepage is displayed as follows.



Configure default backend

Remove the URL configuration and modify the configurations as follows to enable access to Nginx at the backend by using an IP address.

```
lb:
    image: registry.aliyuncs.com/acs/proxy:0.6
    ports:
            - '80:80'
    restart: always
    labels:
         # Addon allows the proxy image to function as a subscription registry center and d
ynamically load the service route.
        aliyun.custom_addon: "proxy"
         # A proxy image container is deployed on each VM.
        aliyun.global: "true"
         # A Server Load Balancer instance is bound to the frontend.
        aliyun.lb.port_80: tcp://proxy_test:80
    environment:
         # Indicates the range of backend containers that support route loading. "*" indica
tes the whole cluster. By default, it indicates the services in applications.
        ADDITIONAL_SERVICES: "*"
        # Specify the error page when 503 is returned.
        EXTRA_FRONTEND_SETTINGS_80: "errorfile 503 /usr/local/etc/haproxy/errors/503.http"
    volumes:
        # Mount the error page to the container from the host.
        - /errors/:/usr/local/etc/haproxy/errors/
appone:
    ports:
        - 80/tcp
        - 443/tcp
    image: 'nginx:latest'
    labels:
         # Indicates that the service must be proxied.
        aliyun.proxy.required: "true"
    restart: always
```

After entering the VIP address of the Server Load Balancer instance, the Nginx homepage is displayed as follows.

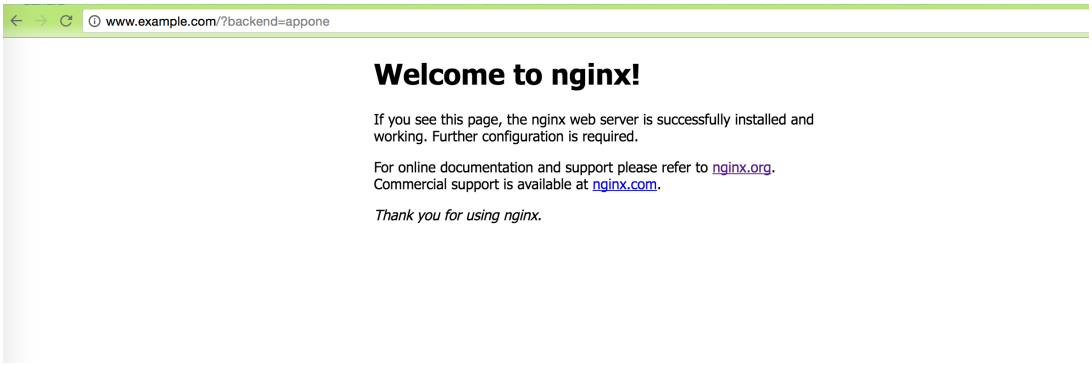
Select backend based on URL parameter values

You can use different backend proxies based on different URL parameter values.

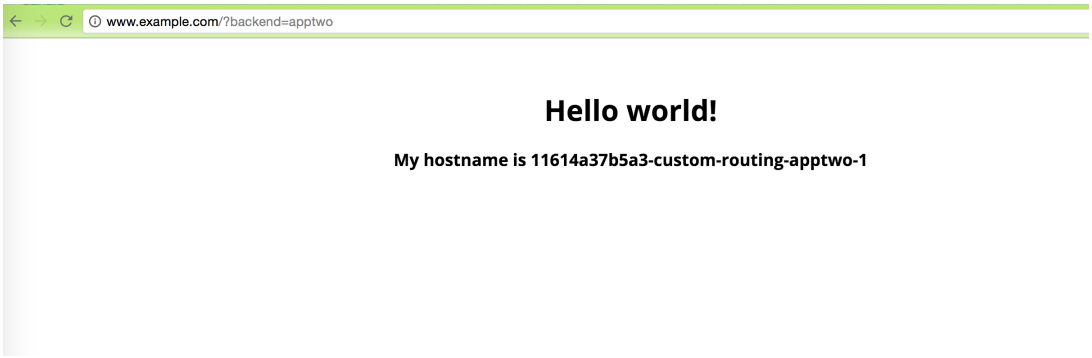
The following example shows how to access the appone service, that is, the Nginx homepage, by using `http://www.example.com?backend=appone` and how to access the apptwo service, that is, the hello world homepage, by using `http://www.example.com?backend=apptwo`. The application template codes are as follows:

```
lb:
    image: registry.aliyuncs.com/acs/proxy:0.6
    ports:
            - '80:80'
    restart: always
    labels:
         # Addon allows the proxy image to function as a subscription registry center and d
ynamically load the service route.
        aliyun.custom_addon: "proxy"
         # A proxy image container is deployed on each VM.
        aliyun.global: "true"
        # A Server Load Balancer instance is bound to the frontend.
        aliyun.lb.port_80: tcp://proxy_test:80
    environment:
         # Indicates the range of backend containers that support route loading. "*" indica
tes the whole cluster. By default, it indicates the services in applications.
        ADDITIONAL_SERVICES: "*"
         # Obtain the value of the "backend" parameter in the URL and modify the HOST heade
r to the backend domain name which needs to be matched.
        EXTRA_FRONTEND_SETTINGS_80: " http-request set-header HOST %[urlp(backend)].example
.com"
appone:
    ports:
        - 80/tcp
        - 443/tcp
    image: 'nginx:latest'
    labels:
         # You can specify paths when configuring URLs. In this example, http/https/ws/wss
are supported.
        aliyun.proxy.VIRTUAL_HOST: "http://appone.example.com"
    restart: always
apptwo:
    ports:
        - 80/tcp
    image: 'registry.cn-hangzhou.aliyuncs.com/linhuatest/hello-world:latest'
    labels:
         # You can specify paths when configuring URLs. In this example, http/https/ws/wss
are supported.
        aliyun.proxy.VIRTUAL_HOST: "http://apptwo.example.com"
    restart: always
```

Bind a host and enter the link `http://www.example.com?backend=appone`. Then, the Nginx homepage for the appone service is displayed as follows.



Bind a host and enter the link `http://www.example.com?backend=apptwo`. Then, the hello world homepage for the apptwo service is displayed as follows.



Record access logs

```
lb:
    image: registry.aliyuncs.com/acs/proxy:0.6
    ports:
            - '80:80'
    restart: always
    labels:
        # Addon allows the proxy image to function as a subscription registry center and dy
namically load the service route.
        aliyun.custom_addon: "proxy"
         # A proxy image container is deployed on each VM.
        aliyun.global: "true"
          # A Server Load Balancer instance is bound to the frontend.
        aliyun.lb.port_80: tcp://proxy_test:80
    environment:
         # Indicates the range of backend containers that support route loading. "*" indica
tes the whole cluster. By default, it indicates the services in applications.
        ADDITIONAL_SERVICES: "*"
        EXTRA_DEFAULT_SETTINGS: "log rsyslog local0,log global,option httplog"
    links:
        - rsyslog:rsyslog
rsyslog:
    image: registry.cn-hangzhou.aliyuncs.com/linhuatest/rsyslog:latest
appone:
    ports:
        - 80/tcp
        - 443/tcp
    image: 'nginx:latest'
    labels:
        # http/https/ws/wss are supported.
        aliyun.proxy.VIRTUAL_HOST: "http://appone.example.com"
    restart: always
```

Logs are printed directly to the standard output of the rsyslog container. The access logs of custom routing can be viewed by using `docker logs $rsyslog_container_name` .

Server Load Balancer between services

The following template creates a Server Load Balancer service `lb` and an application service `appone` to provide services externally with the domain name `appone.example.com` .

```
lb:
    image: registry.aliyuncs.com/acs/proxy:0.6
    Hostname: proxy # Specify the domain name of the service as proxy, which is resolved to
all containers with this image deployed.
    ports:
            - '80:80'
    restart: always
    labels:
         # Addon allows the proxy image to function as a subscription registry center and d
ynamically load the service route.
        aliyun.custom_addon: "proxy"
        # A proxy image container is deployed on each VM.
        aliyun.global: "true"
        # A Server Load Balancer instance is bound to the frontend.
        aliyun.lb.port_80: tcp://proxy_test:80
    environment:
        # Indicates the range of backend containers that support route loading. "*" indicat
es the whole cluster. By default, it indicates the services in applications.
        ADDITIONAL_SERVICES: "*"
appone:
    ports:
        - 80/tcp
        - 443/tcp
    image: 'nginx:latest'
    labels:
         # http/https/ws/wss are supported.
        aliyun.proxy.VIRTUAL_HOST: "http://appone.example.com"
    restart: always
```

The following template is used as a client to access the `appone` application service, but the access path is used to request access to the Server Load Balancer service `lb` and then provide a reverse proxy for the appone application service.

```
restclient: # Simulate rest service consumers.
  image: registry.aliyuncs.com/acs-sample/alpine:3.3
  command: "sh -c 'apk update; apk add curl; while true; do curl --head http://appone.examp
le.com; sleep 1; done'" # Access the rest service and test Server Load Balancer.
  tty: true
  external_links:
    - "proxy:appone.example.com" # Specify the domain name of the link service and the alia
s of the domain name.
```

In the containers of the `restclient` service, the `appone.example.com` domain name is resolved to the IP addresses of all containers of the Server Load Balancer service `lb`.

```
/ # drill appone.example.com
;; ->>HEADER<<- opcode: QUERY, rcode: NOERROR, id: 60917
;; flags: qr rd ra ; QUERY: 1, ANSWER: 3, AUTHORITY: 0, ADDITIONAL: 0
;; QUESTION SECTION:
;; appone.example.com.  IN A
;; ANSWER SECTION:
appone.example.com.     600 IN A 172.18.3.4
appone.example.com.     600 IN A 172.18.2.5
appone.example.com.     600 IN A 172.18.1.5
;; AUTHORITY SECTION:
;; ADDITIONAL SECTION:
;; Query time: 0 msec
;; SERVER: 127.0.0.11
;; WHEN: Mon Sep 26 07:09:40 2016
;; MSG SIZE rcvd: 138
```

Configure monitoring page

```
lb:
    image:  registry.aliyuncs.com/acs/proxy:0.6
    ports:
            -  '80:80'
            -  '127.0.0.1:1935:1935' # The port that monitoring page exposes to the public
network. Configure the port with due care because of the potential security risk.
    restart:  always
    labels:
        aliyun.custom_addon:  "proxy"
        aliyun.global:  "true"
        aliyun.lb.port_80: tcp://proxy_test:80
    environment:
        ADDITIONAL_SERVICES:  "*"
        STATS_AUTH: "admin:admin" # The logon account and password used for monitoring, whi
ch are customizable.
        STATS_PORT: "1935" # The port used for monitoring, which is customizable.
appone:
    expose:
        -  80/tcp
    image:  'nginx:latest'
    labels:
        aliyun.proxy.VIRTUAL_HOST:  "http://appone.example.com"
    restart:  always
```

Log on to each machine where the custom routing image resides (each machine can receive the request, no matter the application container is on which machine) and request the `acs/proxy` health check page.

> **Note** Configure the correct username and password according to the environment variable STATS_AUTH of the application template.

```
root@c68a460635b8c405e83c052b7c2057c7b-node2:~# curl -Ss -u admin:admin 'http://127.0.0.1:1935/' &> test.html
```

Copy the page `test.html` to a machine with browsers and open the local file `test.html` with the browser. View the stats monitoring statistics page. Green indicates the network from container `acs/proxy` to backend containers is connected and the container acs/proxy is working normally. Other colors indicate an exception.

16.9. Custom routing - Supports TCP

When Alibaba Cloud Container Service is in use, the following problem may occur to TCP Server Load Balancer: when the client image and server image of an application are deployed on the same Elastic Compute Service (ECS) instance, the application client cannot access the local server by using Server Load Balancer due to the limitation of Server Load Balancer. In this document, take the common TCP-based Redis as an example acs/proxy to describe how to solve the problem by using the custom routing acs/proxy.

> **Note** Different services cannot share the same Server Load Balancer instance. Otherwise, the backend machine of the Server Load Balancer is deleted and the services are unavailable.

Solution 1: Deploy client and server containers on different nodes by scheduling containers

The following is a sample application template (the lb label and swarm filter function are used):

```
redis-master:
    ports:
      - 6379:6379/tcp
    image: 'redis:alpine'
    labels:
        aliyun.lb.port_6379: tcp://proxy_test:6379
redis-client:
    image: 'redis:alpine'
    links:
      - redis-master
    environment:
      - 'affinity:aliyun.lb.port_6379!=tcp://proxy_test:6379'
    command: redis-cli -h 120.25.131.64
    stdin_open: true
    tty: true
```

> **Note**
>
> - Follow these steps if the scheduling does not take effect: Log on to the Container Service console. Click Swarm > **Services** in the left-side navigation pane. Select the cluster in which the service you want to reschedule resides from the Cluster drop-down list. Click **Reschedule** at the right of the service you want to reschedule. > Select the **Force Reschedule** check box in the displayed dialog box and then click **OK**.
> - The volumes of existing containers will be lost if you select the **Force Reschedule** check box. Backup and migrate the data in advance.

Solution 2: Clients inside the container cluster access the server by using links, while clients outside access the server by using Server Load Balancer

The following is a sample application template (the lb label is used):

```
redis-master:
    ports:
      - 6379:6379/tcp
    image: 'redis:alpine'
    labels:
        aliyun.lb.port_6379: tcp://proxy_test:6379
redis-client:
    image: 'redis:alpine'
    links:
      - redis-master
    command: redis-cli -h redis-master
    stdin_open: true
    tty: true
```

Solution 3: Clients inside the container cluster access the server by using Custom routing (which is based on HAProxy and serves as a proxy server), while clients outside access the server by using Server Load Balancer

The following is a sample application template (the lb label and Custom routing - simple sample are used):

```
lb:
    image: registry.aliyuncs.com/acs/proxy:0.6
    ports:
            - '6379:6379/tcp'
    restart: always
    labels:
        # Addon allows the proxy image to function as a subscription registry center and dy
namically load the service route.
        aliyun.custom_addon: "proxy"
        # A proxy image container is deployed on each virtual machine (VM).
        aliyun.global: "true"
        # A Server Load Balancer instance is bound to the frontend, and the lb label is use
d
```

```
        aliyun.lb.port_6379: tcp://proxy_test:6379
        # Indicates that the custom routing must be started after the master Redis and slave Redis are started, and the custom routing depends on the master Redis and slave Redis.
        aliyun.depends: redis-master,redis-slave
    environment:
        # Indicates the range of backend containers that support route loading. "*" indicates the whole cluster. By default, it indicates the services in applications.
        ADDITIONAL_SERVICES: "*"
        EXTRA_DEFAULT_SETTINGS: "log rsyslog local0,log global,option httplog"
        # Configures HAProxy to work in TCP mode.
        MODE: "tcp"
    links:
        - rsyslog:rsyslog
rsyslog:
    image: registry.cn-hangzhou.aliyuncs.com/linhuatest/rsyslog:latest
redis-master:
    ports:
      - 6379/tcp
    image: 'redis:alpine'
    labels:
        # Indicates that the custom routing is to expose the port 6379.
        aliyun.proxy.TCP_PORTS: "6379"
        # Indicates that the service route is to be added to the custom routing.
        aliyun.proxy.required: "true"
redis-slave:
    ports:
      - 6379/tcp
    image: 'redis:alpine'
    links:
      - redis-master
    labels:
      # Indicates that the custom routing is to expose the port 6379.
      aliyun.proxy.TCP_PORTS: "6379"
      # Indicates that the service route is to be added to the custom routing.
      aliyun.proxy.required: "true"
      # Indicates that the slave Redis depends on the master Redis and must be started after the master Redis is started.
      aliyun.depends: redis-master
    command: redis-server --slaveof redis-master 6379
redis-client:
    image: 'redis:alpine'
    links:
      - lb:www.example.com
    labels:
      aliyun.depends: lb
    command: redis-cli -h www.example.com
    stdin_open: true
    tty: true
```

This solution provides a master-slave Redis architecture and balances load by using the Server Load Balancer between services to make Container Server become highly available.

16.10. Custom routing - supports multiple HTTPS certificates

Use the acs/proxy image acs/proxy in this example.

> **Note** Services cannot use the same Server Load Balancer; otherwise, the backend machine of the Server Load Balancer will be deleted, and the service will be unavailable.

```
lb:
    image: registry.aliyuncs.com/acs/proxy:0.6
    ports:
            - '80:80'
            - '443:443' # HTTPS must expose this port
    restart: always
    labels:
        # Addon allows the proxy image to function as a subscription registry center and dy
namically load the service route.
        aliyun.custom_addon: "proxy"
        # A proxy image container is deployed on each virtual machine (VM).
        aliyun.global: "true"
        # A Server Load Balancer instance is bound to the frontend.
        aliyun.lb.port_80: tcp://proxy_test:80
        aliyun.lb.port_443:tcp://proxy_test:443
    environment:
        # Indicates the range of backend containers that support route loading. "*" indicat
es the whole cluster. By default, it indicates the services in applications.
        ADDITIONAL_SERVICES: "*"
appone:
    expose:  # For proxied services, use expose or ports to tell proxy containers which por
t is to be exposed.
        - 80/tcp
    image: 'nginx:latest'
    labels:
        # You can specify paths when configuring URLs. In this example, http/https/ws/wss a
re supported.
        aliyun.proxy.VIRTUAL_HOST: "https://appone.example.com"
        # Configure the appone certificate.
        aliyun.proxy.SSL_CERT: "-----BEGIN RSA PRIVATE KEY-----\nMIIEpQIBAAKCAQEAvgnKhephWH
KWYDEiBiSjzst7nRP0DJxZ5cIOxyXmncd2kslr\nkUIB5qT/MSiJGBL3Lr4advs6kI/JFmxloFrPtwEe2FGkLBfCDXX
DrWgxyFhbuPQY\nBLNueUu94sffIxg+4u5Mriui7ftindOAf0d21PSM9gb/ZUypxIgAd3RHCe/gtT0h\nVCn6FikXyn
XLDTODYWCthQHBwSZS88HNU+B0T9Yl65JiQ0mV+YF+h3D/c232E6Gp\nzK+8ehVB13s5hecUx3dvdUQPBUhJYvzsPjC
hgsXSMDRexiN66kbhH6dJArsrYb8t\nEBWXfCZaTcF82wkAsUe/fhlGhh97h+66lh6OQQIDAQABAoIBAQC4d8ifNWRI
9vIB\nbbAZRne7xMm5MCU2GI8q97Rgm+nAPl5bHinMVsaBnKgaj76EH+TQ+re1xyiSKwCH\nQ7FidsQqYGwQjy9NncJ
ATpAjQ4EPeLWQU2D9Ly+NjnhEKr/u0Ro6LhdA+hqt59dS\nXHvfEP/It5odN62yJzikDWBmk/hhK0tu28dPYUuPoWsw
XWFMkaNttmfLgZlagiqr\nYp7rxAFqQurzctQ2VNwezekDHQoh8ounHGEniZ+fA6sFtYi83KTKWkvFom1chZQr\nxxP
bbgANJJJlNgtkl6JZNxj6SYimmWvzmrrU25khKg/klP5EtQzIx6UFhURnuTKu\nzNgqcIABAoGBAOqUOerveEUePvsA
lta8CV/p2KKwenv+kUofQ4UpKFXfnHbQHQfr\nZHS29OQiPxqjVXYLu8gNfLRfKtUNyqV+TDrzJ1elW2RKc00GHAwPb
XxijPhmJ2fW\neskn8tlDcyXpvoqWJG34896vo4IbcL0H/eUs0jJo6OJlCQBKXik+t3gxAoGBAM9k\nVOTV2caKyrZ4
ta0Q1LKqKfOkt0j+vKz167J5pSLjVKQSUxGMyLnGwiQdDtB4iy6L\nFcCB/S0HM0UWkJWhNYAL8kHry53bVdHtQG0tu
YFYvBJo7A+Nppsn9MtlVh8KbVu4\nhOz/3MWwbQNnvIVCGK/fSltS1GhTk4rKL7PjNwMRAoGBALK0n3bqXj6Rrzs7FK
6c\na6vlE4PFXFpv8jF8pcyhMThSdPlSzHsHCe2cn+3YZSie+/FFORZLqBAlXBUZP6Na\nFyrlqLgtofVCfppUKDPL4
```

```
QXccjaeZDDIBZyPUYPQzb05WE5t2WzqNqcUOUVaMEXh\n+7uGrM94espWXEgbX6aeP9lRAoGARlJQ7t8MXuQE5GZ9w9
cnKAXG/9RkSZ4Gv+cL\nKpNQyUmoE5IbFKJWFZgtkC1CLrIRD5EdqQ7ql/APFGgYUoQ9LdPfKzcW7cnHic0W\nwW51r
kQ2UU++a2+uhIHB4Y3U6+WPO0CP4gTICUhPTo5IQC8vS8M85UZqu41LRA5W\nqnpq1uECgYEAq+6KpHhlR+5h3Y/m0n
84yJ0YuCmrl7HFRzBMdOcaW3oaYL83rAaq\n6dJqpAVgeu3HP8AtiGVZRe78J+n4d2JGYSqgtP2lFFTdF9HfhcR2P9b
UBNYtWols\nEs3iw53t8a4BndLGBwLPA3lklf7J5stYanRv6NqaRaLq4FQMxsW1A0Q=\n-----END RSA PRIVATE K
EY-----\n-----BEGIN CERTIFICATE-----\nMIIDvDCCAqSgAwIBAgIBATANBgkqhkiG9w0BAQUFADBgMQswCQYDV
QQGEwJDTjER\nMA8GA1UECBMIWmhlamlhbmcxETAPBgNVBAcTCEhhbmd6aG91MRQwEgYDVQQKEwth\nbGliYWJhLmNv
bTEVMBMGA1UECxMMd3d3LnJvb3QuY29tMB4XDTE1MDIwOTA1MzQx\nOFoXDTE2MDIwOTA1MzQxOFowZjELMAkGA1UEB
hMCQ04xETAPBgNVBAgTCFpoZWpp\nYW5nMRQwEgYDVQQKEwthbGliYWJhLmNvbTEVMBMGA1UECxMMd3d3LnJvb3QuY2
9t\nMRcwFQYDVQQDEw53d3cubGluaHVhLmNvbTCCASIwDQYJKoZIhvcNAQEBBQADggEP\nADCCAQoCggEBAL4JyoXqY
VhylmAxIgYko87Le50T9AycWeXCDscl5p3HdpLJa5FC\nAeak/zEoiRgS9y6+Gnb7OpCPyRZsZaBaz7cBHthRpCwXwg
11w61oMchYW7j0GASz\nbnlLveLH3yMYPuLuTK4rou37Yp3TgH9HdtT0jPYG/2VMqcSIAHd0Rwnv4LU9IVQp\n+hYpF
8p1yw0zg2FgrYUBwcEmUvPBzVPgdE/WJeuSYkNJlfmBfodw/3Nt9hOhqcyv\nvHoVQdd7OYXnFMd3b3VEDwVISWL87D
4woYLF0jA0XsYjeupG4R+nSQK7K2G/LRAV\nl3wmWk3BfNsJALFHv34ZRoYfe4fuupYejkECAwEAAaN7MHkwCQYDVR0
TBAIwADAs\nBglghkgBhvhCAQ0EHxYdT3BlblNTTCBHZW5lcmF0ZWQgQ2VydGlmaWNhdGUwHQYD\nVR0OBBYEFM6ESm
kDKrqnqMwBawkjeONKrRMQMB8GA1UdIwQYMBaAFFUrhN9ro+Nm\nrZnl4WQzDpgTbCBhMA0GCSqGSIb3DQEBBQUAA4I
BAQCQ2D9CRiv8brx3fnr/RZG6\nFYPEdxjY/CyfJrAbij0PdKjzZKk1O67chM1Oxs2JhJ6tMqg2sv50bGx4XmbSPmEe
\nYTJjIXMY+jCoJ/Zmk3Xgu4K1y1LvD25PahDVhRrPN8H4WjsYu51pQNshil5E/3iQ\n2JoV0r8QiAsPiiY5+mNCD1f
m+QN1tyUabczi/DHafgWJxf2B3M66e3oUdtbzA2pf\nYHR8RVeSFrjaBqudO8ir+uYcRbRkroYmY5Vm+4Yp64oetrPp
KUPWSYaAZ0uRtpeL\nB5DpqXz9GEBb5m2Q4dKjs5Hm6vyFUORCzZcO4XexDhcgdLOH5qznmh9oMCk9QvZf\n-----EN
D CERTIFICATE-----\n"
    restart: always
apptwo:
    expose:  # For proxied services, use expose or ports to tell proxy containers which por
t is to be exposed.
        - 80/tcp
    image: 'registry.cn-hangzhou.aliyuncs.com/linhuatest/hello-world:latest'
    labels:
        # You can specify paths when configuring URLs. In this example, http/https/ws/wss a
re supported.
        aliyun.proxy.VIRTUAL_HOST: "https://apptwo.example.com"
        # Configure the apptwo certificate.
        aliyun.proxy.SSL_CERT: "-----BEGIN RSA PRIVATE KEY-----\nMIIEpQIBAAKCAQEAvgnKhephWH
KWYDEiBiSjzst7nRP0DJxZ5cIOxyXmncd2kslr\nkUIB5qT/MSiJGBL3Lr4advs6kI/JFmxloFrPtwEe2FGkLBfCDXX
DrWgxyFhbuPQY\nBLNueUu94sffIxg+4u5Mriui7ftindOAf0d21PSM9gb/ZUypxIgAd3RHCe/gtT0h\nVCn6FikXyn
XLDTODYWCthQHBwSZS88HNU+B0T9Yl65JiQ0mV+YF+h3D/c232E6Gp\nzK+8ehVB13s5hecUx3dvdUQPBUhJYvzsPjC
hgsXSMDRexiN66kbhH6dJArsrYb8t\nEBWXfCZaTcF82wkAsUe/fhlGhh97h+66lh6OQQIDAQABAoIBAQC4d8ifNWRI
9vIB\nbbAZRne7xMm5MCU2GI8q97Rgm+nAPl5bHinMVsaBnKgaj76EH+TQ+re1xyiSKwCH\nQ7FidsQqYGwQjy9NncJ
ATpAjQ4EPeLWQU2D9Ly+NjnhEKr/u0Ro6LhdA+hqt59dS\nXHvfEP/It5odN62yJzikDWBmk/hhK0tu28dPYUuPoWsw
XWFMkaNttmfLgZlagiqr\nYp7rxAFqQurzctQ2VNwezekDHQoh8ounHGEniZ+fA6sFtYi83KTKWkvFom1chZQr\nxxP
bbgANJJJlNgtkl6JZNxj6SYimmWvzmrrU25khKg/klP5EtQzIx6UFhURnuTKu\nzNgqcIABAoGBAOqUOerveEUePvsA
lta8CV/p2KKwenv+kUofQ4UpKFXfnHbQHQfr\nZHS29OQiPxqjVXYLu8gNfLRfKtUNyqV+TDrzJ1elW2RKc00GHAwPb
XxijPhmJ2fW\neskn8tlDcyXpvoqWJG34896vo4IbcL0H/eUs0jJo6OJlCQBKXik+t3gxAoGBAM9k\nVOTV2caKyrZ4
ta0Q1LKqKfOkt0j+vKz167J5pSLjVKQSUxGMyLnGwiQdDtB4iy6L\nFcCB/S0HM0UWkJWhNYAL8kHry53bVdHtQG0tu
YFYvBJo7A+Nppsn9MtlVh8KbVu4\nhOz/3MWwbQNnvIVCGK/fSltS1GhTk4rKL7PjNwMRAoGBALK0n3bqXj6Rrzs7FK
6c\na6vlE4PFXFpv8jF8pcyhMThSdPlSzHsHCe2cn+3YZSie+/FFORZLqBAlXBUZP6Na\nFyrlqLgtofVCfppUKDPL4
QXccjaeZDDIBZyPUYPQzb05WE5t2WzqNqcUOUVaMEXh\n+7uGrM94espWXEgbX6aeP9lRAoGARlJQ7t8MXuQE5GZ9w9
cnKAXG/9RkSZ4Gv+cL\nKpNQyUmoE5IbFKJWFZgtkC1CLrIRD5EdqQ7ql/APFGgYUoQ9LdPfKzcW7cnHic0W\nwW51r
kQ2UU++a2+uhIHB4Y3U6+WPO0CP4gTICUhPTo5IQC8vS8M85UZqu41LRA5W\nqnpq1uECgYEAq+6KpHhlR+5h3Y/m0n
84yJ0YuCmrl7HFRzBMdOcaW3oaYL83rAaq\n6dJqpAVgeu3HP8AtiGVZRe78J+n4d2JGYSqgtP2lFFTdF9HfhcR2P9b
UBNYtWols\nEs3iw53t8a4BndLGBwLPA3lklf7J5stYanRv6NqaRaLq4FQMxsW1A0Q=\n-----END RSA PRIVATE K
EY-----\n-----BEGIN CERTIFICATE-----\nMIIDvDCCAqSgAwIBAgIBATANBgkqhkiG9w0BAQUFADBgMQswCQYDV
QQGEwJDTjER\nMA8GA1UECBMIWmhlamlhbmcxETAPBgNVBAcTCEhhbmd6aG91MRQwEgYDVQQKEwth\nbGliYWJhLmNv
bTEVMBMGA1UECxMMd3d3LnJvb3QuY29tMB4XDTE1MDIwOTA1MzQx\nOFoXDTE2MDIwOTA1MzQxOFowZjELMAkGA1UEB
```

```
hMCQ04xETAPBgNVBAgTCFpoZWpp\nYW5nMRQwEgYDVQQKEwthbGliYWJhLmNvbTEVMBMGA1UECxMMd3d3LnJvb3QuY2
9t\nMRcwFQYDVQQDEw53d3cubGluaHVhLmNvbTCCASIwDQYJKoZIhvcNAQEBBQADggEP\nADCCAQoCggEBAL4JyoXqY
VhylmAxIgYko87Le50T9AycWeXCDscl5p3HdpLJa5FC\nAeak/zEoiRgS9y6+Gnb7OpCPyRZsZaBaz7cBHthRpCwXwg
11w61oMchYW7j0GASz\nbnlLveLH3yMYPuLuTK4rou37Yp3TgH9HdtT0jPYG/2VMqcSIAHd0Rwnv4LU9IVQp\n+hYpF
8p1yw0zg2FgrYUBwcEmUvPBzVPgdE/WJeuSYkNJlfmBfodw/3Nt9hOhqcyv\nvHoVQdd7OYXnFMd3b3VEDwVISWL87D
4woYLF0jA0XsYjeupG4R+nSQK7K2G/LRAV\nl3wmWk3BfNsJALFHv34ZRoYfe4fuupYejkECAwEAAaN7MHkwCQYDVR0
TBAIwADAs\nBglghkgBhvhCAQ0EHxYdT3BlblNTTCBHZW5lcmF0ZWQgQ2VydGlmaWNhdGUwHQYD\nVR0OBBYEFM6ESm
kDKrqnqMwBawkjeONKrRMQMB8GA1UdIwQYMBaAFFUrhN9ro+Nm\nrZnl4WQzDpgTbCBhMA0GCSqGSIb3DQEBBQUAA4I
BAQCQ2D9CRiv8brx3fnr/RZG6\nFYPEdxjY/CyfJrAbij0PdKjzZKk1O67chM1Oxs2JhJ6tMqg2sv50bGx4XmbSPmEe
\nYTJjIXMY+jCoJ/Zmk3Xgu4K1y1LvD25PahDVhRrPN8H4WjsYu51pQNshil5E/3iQ\n2JoV0r8QiAsPiiY5+mNCD1f
m+QN1tyUabczi/DHafgWJxf2B3M66e3oUdtbzA2pf\nYHR8RVeSFrjaBqudO8ir+uYcRbRkroYmY5Vm+4Yp64oetrPp
KUPWSYaAZ0uRtpeL\nB5DpqXz9GEBb5m2Q4dKjs5Hm6vyFUORCzZcO4XexDhcgdLOH5qznmh9oMCk9QvZf\n-----EN
D CERTIFICATE-----\n"
    restart: always
```

Services appone and apptwo use `aliyun.proxy.VIRTUAL_HOST` to specify the domain names. If you must configure the certificate, set the protocol to https. Then, use `aliyun.proxy.SSL_CERT` to specify the certificate content. The method of configuring the certificate content is as follows:

Assume that the *key.pem* is a private key file, and *ca.pem* is a public key file. Run the following commands in the bash (the current directory contains the public key file and private key file).

```
$ cp key.pem cert.pem
$ cat ca.pem >> cert.pem
$ awk 1 ORS='\\n' cert.pem
```

Finally, enter the output of the `awk` command as the value of label `aliyun.proxy.SSL_CERT`. Use double quotation marks (“) for separation. For other information, such as lb label, lb see the preceding template and the corresponding Custom routing - simple sample.

17.Release policy

17.1. Introductions on release strategies

Container Service provides two release strategies: blue-green release and standard release. These two release strategies differ in:

Release strategies	Differences
Standard release	Delete the earlier version when deploying the new version of application. Your service will be temporarily interrupted in the process of release.
Blue-green release	Blue version and green version are generated when the application is updated. These two versions play an active-standby role to each other and go online or offline based on your configured route weight. This kind of release strategy has the following features: The service is not interrupted. The application never goes down. Users do not realize the restart in the process of release. The application automatically rolls back if the update fails. Multiple updates and iterations can be performed on the same resource stack. **Note** Applications created by using images cannot be updated using blue-green release strategy.

Usage scenarios of standard release

The standard release, a traditional release strategy for applications, deploys the new version of the application in the current environment. This release strategy is generally used except for special needs. The usage scenarios include:

- The new version has been fully tested, with no need of online test run.
- The application includes database service and has been updated in an unreversible manner, such as datasheet structure change. In this scenario, standard release avoids business transformation, such as data migration and rollback.

Usage scenarios of blue-green release

Application forms that are applicable to blue-green release include the frontend services and backend services. Blue-green release is often used in the incremental update of applications. The specific customer-facing business scenarios that use this kind of release strategy include:

Guarantee of the business continuity

The frequent iterations of application versions is a real challenge to Internet enterprises with quickly-changing business. It is necessary to ensure the continuity of online business. The main value of blue-green release strategy is to achieve application update with no downtime, ensure the service is not interrupted, and meet the requirements of sustainable release of updated applications in the cloud environment.

Online evaluation of new version

Blue-green release strategy enables you to perform the version test in the online environment, and fully test the service functions, performance, and security of the new version by keeping the earlier and the latest versions coexisting for a period of time. When the new version is stable, bring the earlier version offline.

17.2. Blue-green release policy with simple routing

> **Note** By default, simple routing performs session persistence. During a blue-green release, when the new service weight is set to 100%, old requests might be forwarded to the old version of the service. To forward requests to the new version of the service, disable session persistence for the application before the release, see routing.session_sticky or clear cookies after the release.

Background information

Blue-green release is a zero downtime application update policy. During a blue-green release, the old and new service versions of an application coexist, and also share routes. By adjusting route weights, you can switch traffic between different service versions. After the new version passes the verification, you can delete the old service version by confirming the release. If the new version does not pass the verification, the release is rolled back and the new version is deleted.

Prerequisites

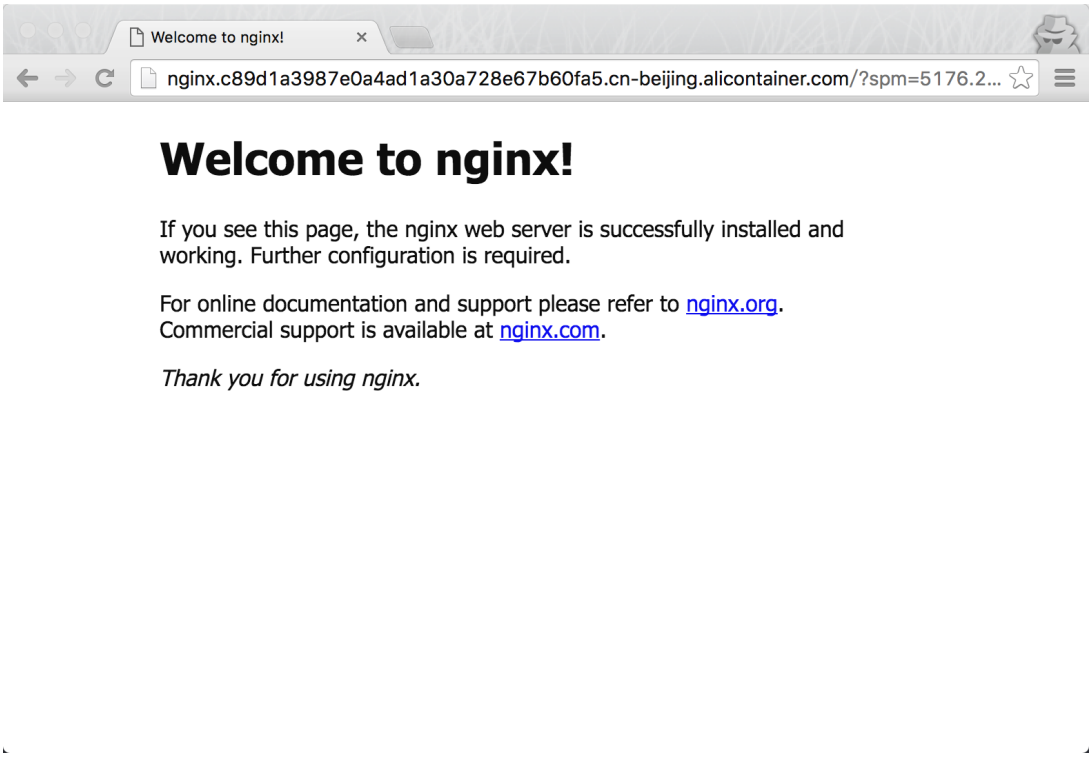
The routing service must be upgraded to the latest version. For more information, see Upgrade system services.

Scenario

In the following example, assume that you perform a blue-green release for an Nginx static page application. The initial application template is as follows:

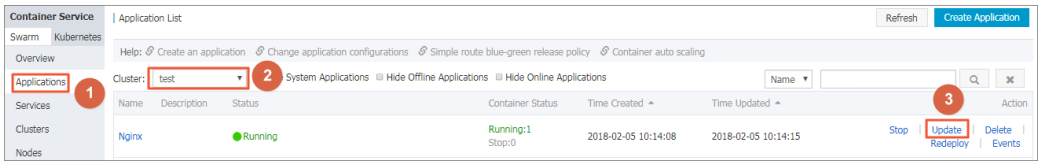
```
nginx-v1:
  image: 'registry.aliyuncs.com/ringtail/nginx:1.0'
  labels:
    aliyun.routing.port_80: nginx
  restart: always
```

After the deployment, the page is as follows.

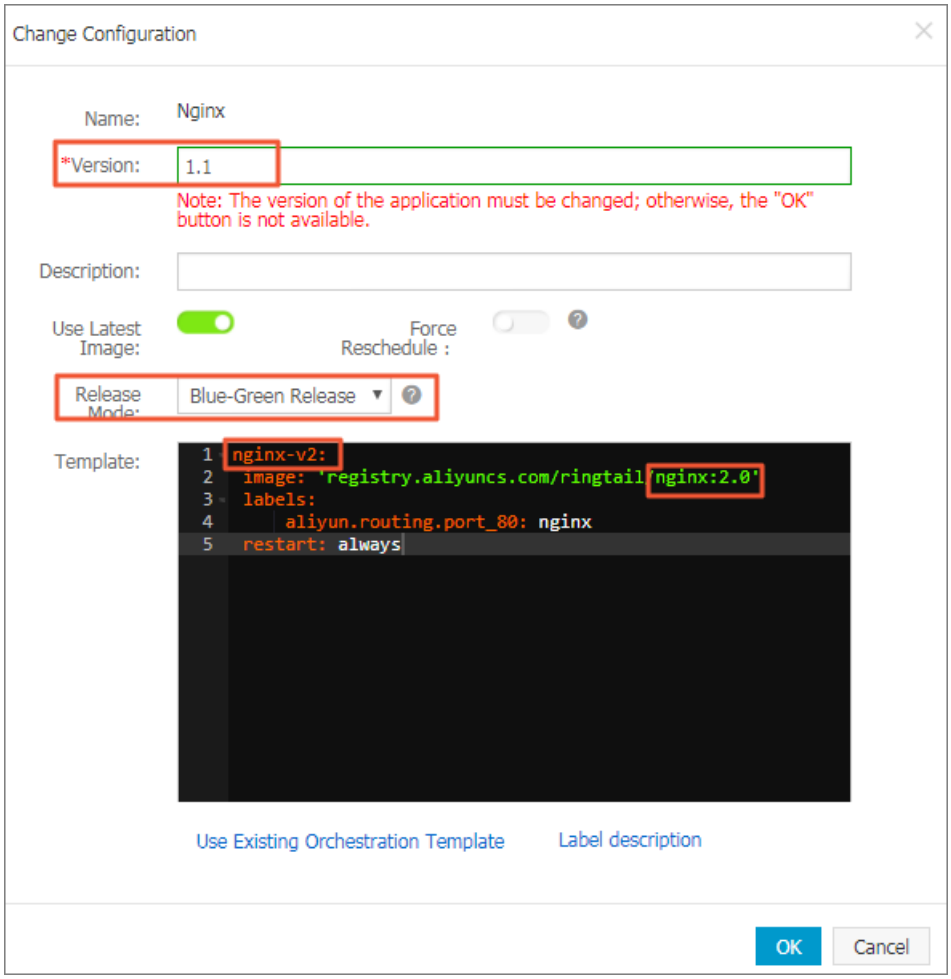


Procedure

1. Log on to the Container Service console.
2. Under Swarm, click **Applications** in the left-side navigation pane.
3. Select the cluster in which the application resides from the Cluster drop-down list.
4. Click **Update** at the right of the application.



5. Set the release mode and the configurations of the new service version.
   - The new and old versions cannot share the same name.
   - To make sure that the application does not experience downtime when switching versions, the weight of the new service version is set to 0 by default. On the route management page, you must adjust the weight to switch traffic to the new version.



The template sample is as follows:

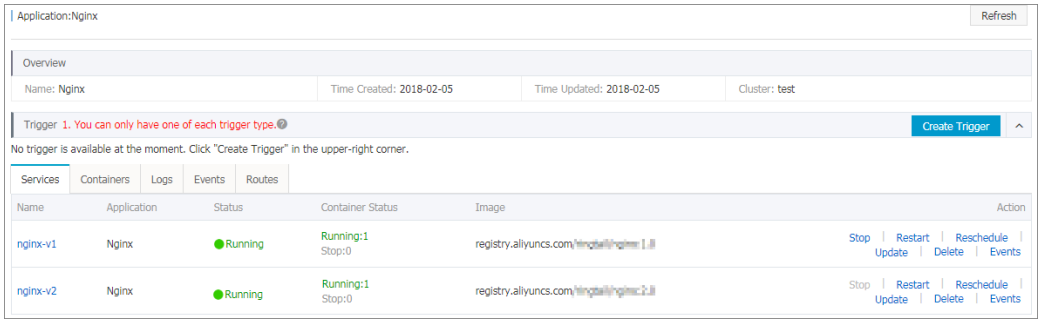
```
nginx-v2:
 image: 'registry.aliyuncs.com/ringtail/nginx:2.0'
 labels:
     aliyun.routing.port_80: nginx
 restart: always
```

6. Click **OK** to release the changed version.

   The release process goes through two statuses:

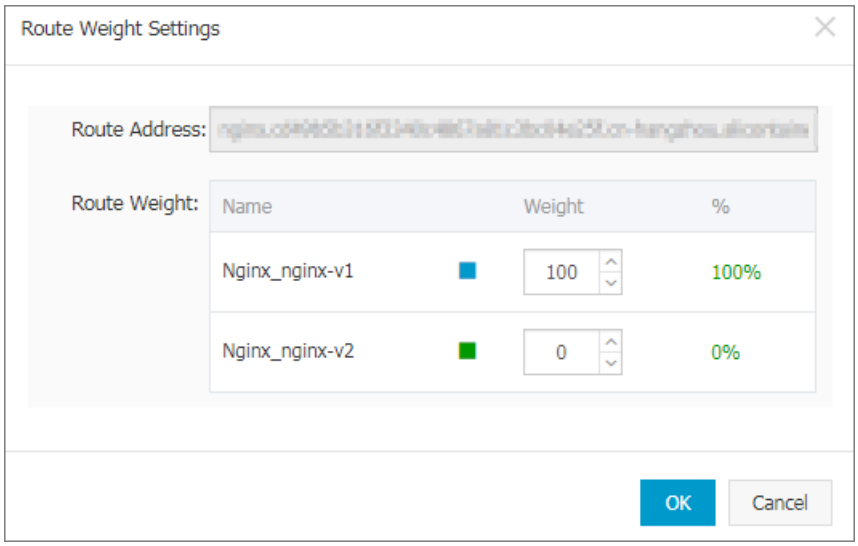
   - Blue-green release in progress: Indicates that the startup of the new service version is not completed.
   - Blue-green release awaiting confirmation: Indicates that the startup of the new service version is completed. Another release can be performed until this release is confirmed or rolled back.

     Click the application name to go to the application details page. You can see that the new and old application versions coexist.



7. Click the **Routes** tab and then click **Set service weight**.

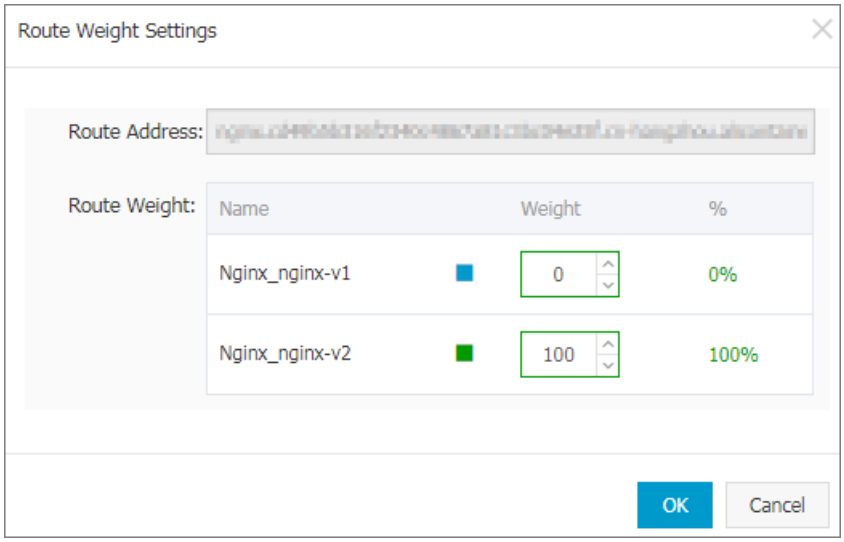
   As shown in the following figure, the old service has a weight of 100 and the new service has a weight of 0.



   To realize zero downtime update, set the weight of the new version to 100. Now the new version and old version account for 50% of the weight respectively. Test if both versions have stable traffic.

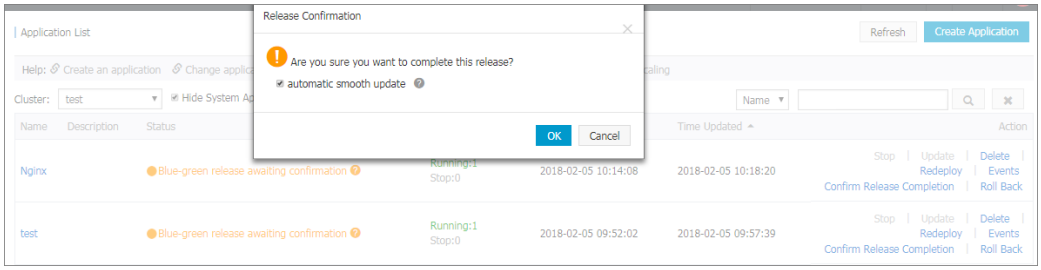
   > **Note** Adjusting the weights of the new version and old version at the same time might result in the failure of some requests. Therefore, adjust the weights in two steps and only adjust the weight of one version in each step. For example, adjust the weight of the new version from 0 to 100 first, and then adjust the weight of the old version from 100 to 0 after the traffic is stable.

   Then, adjust the weight of the old version to 0 and that of the new version to 100.

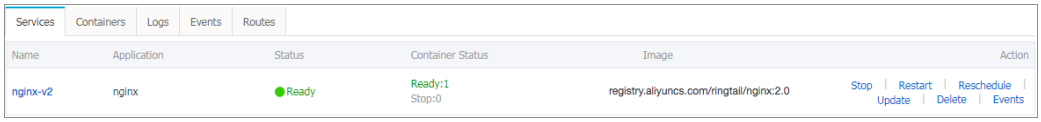


8. The routing service enables the session persistence by default. Therefore, you can open a new browser window to access the new version.

9. After the entire release process has been verified, click **Confirm Release Completion** on the Application List page. Select whether or not to automatically perform the **smooth update** in the displayed dialog box and click **OK** to confirm the release before you can release subsequent versions.



Now the service list of the application has been updated and the old service version has been taken offline and deleted.



17.3. Blue-green release policy with Server Load Balancer routing

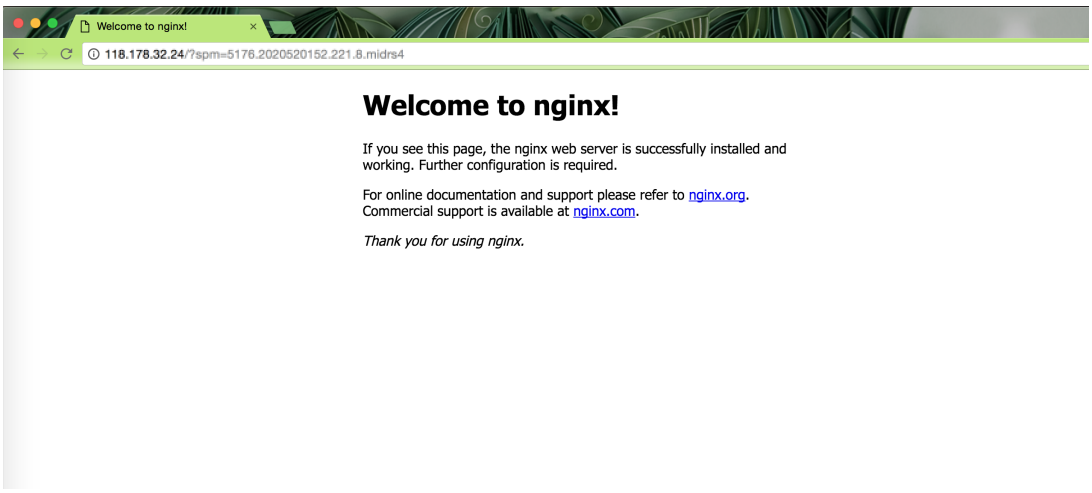
Blue-green release is a zero downtime application update policy. During a blue-green release, the old and new service versions of an application coexist, and also share the Server Load Balancer instance. By adjusting the Server Load Balancer weights, you can switch traffic between different service versions. After the new version passes the verification, you can delete the old service version by confirming the release. If the new version does not pass the verification, the release is rolled back and the new version is deleted.

Scenarios

In the following example, assume that you want to perform a blue-green release for an Nginx static page application. The initial application template is as follows:

```
nginx-v1:
  image: 'registry.aliyuncs.com/ringtail/nginx:1.0'
  ports:
    - 80:80/tcp
  labels:
    aliyun.lb.port_80: tcp://proxy_test:80
  restart: always
```

After the deployment, the page is as follows.



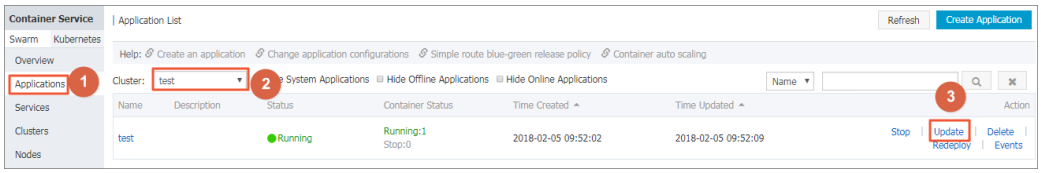
Instructions

Each container needs to expose the host port. Therefore, when performing blue-green release with Server Load Balancer routing, make sure that the number of containers in a service is less than or equal to half the number of machines in the cluster. Otherwise, the port conflict occurs.

We recommend that you contract the number of containers to half the number of machines in the cluster before performing blue-green release with Server Load Balancer routing. Expand the number of containers to the original amount after the blue-green release is completed.

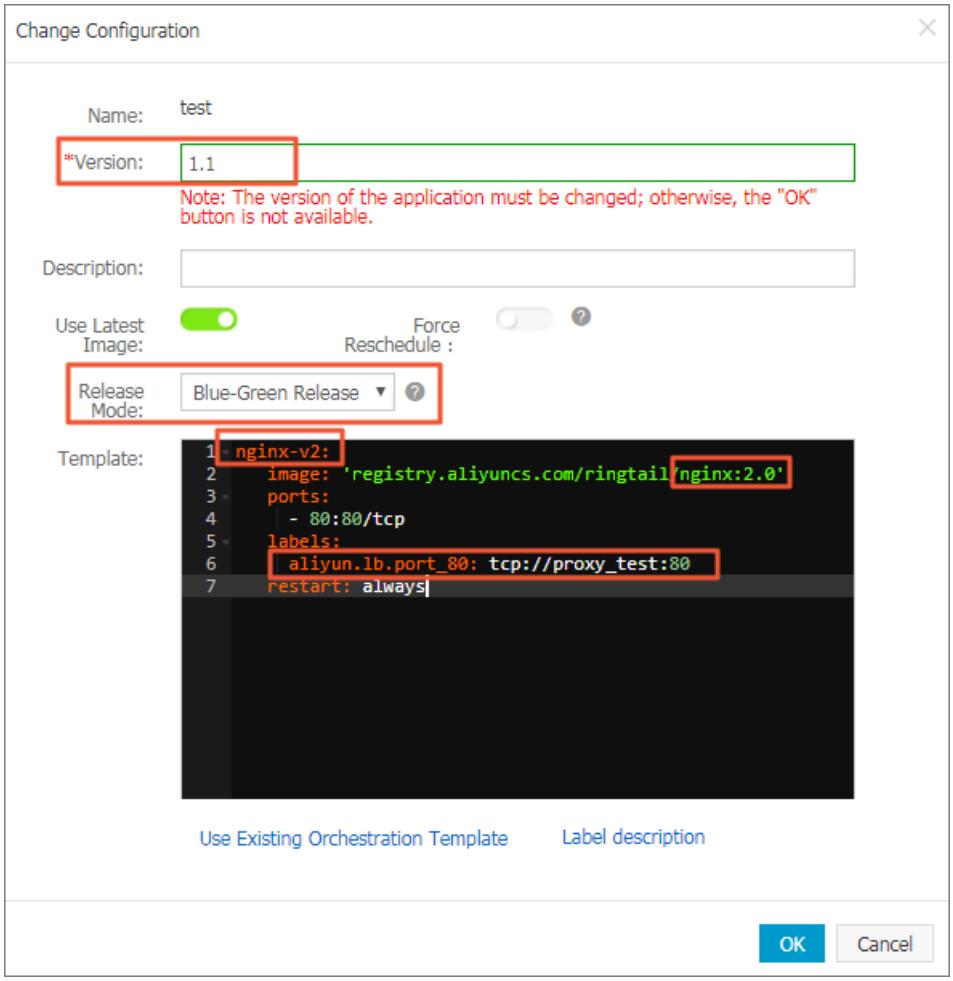
Procedure

1. Log on to the Container Service console.
2. Click Swarm > **Applications** in the left-side navigation pane.
3. Select the cluster in which the application resides from the Cluster drop-down list.
4. Click **Update** at the right of the application.



5. Set the release mode and the configurations of the new service version.

> **Note**
>
> - The new and old versions cannot share the same name.
> - To make sure that the application does not experience downtime when switching versions, the weight of the new service version is set to 0 by default. On the route management page, you must adjust the weight to switch traffic to the new version.



The template sample is as follows:

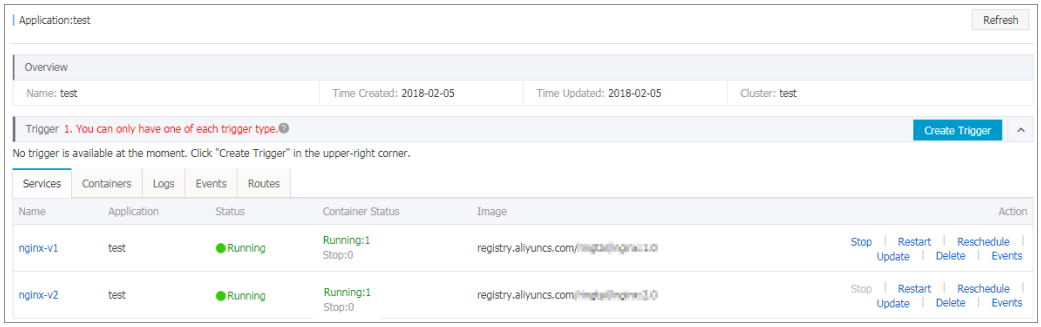
```
nginx-v2:
   image: 'registry.aliyuncs.com/ringtail/nginx:2.0'
   ports:
     -80: 80/tcp
   labels:
     aliyun.lb.port_80: tcp://proxy_test:80
   restart: always
```

6. Click **OK** to release the changed version.

   The release process goes through two statuses:

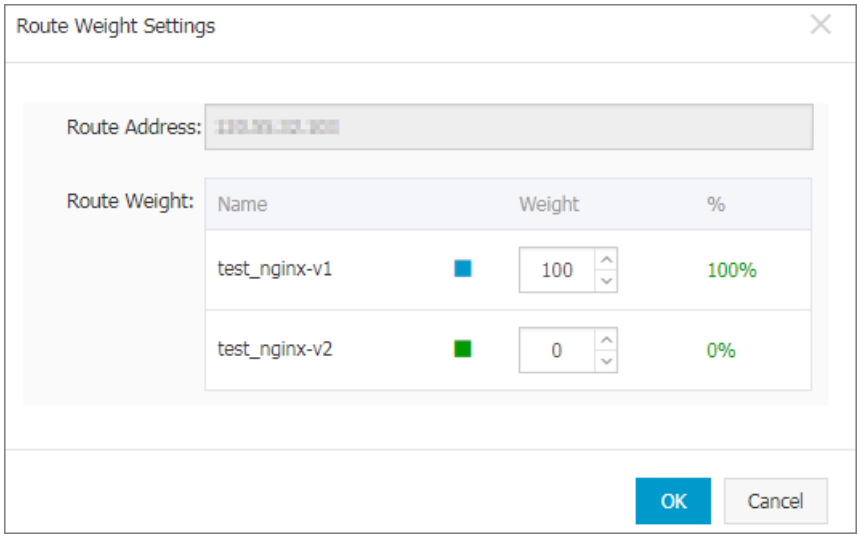
   - Blue-green release in progress: Indicates that the startup of the new service version is not completed.
   - Blue-green release awaiting confirmation: Indicates that the startup of the new service version is completed. Another release can be performed until this release is confirmed or rolled back.

     Click the application name to go to the application details page. You can see that the new and old application versions coexist.



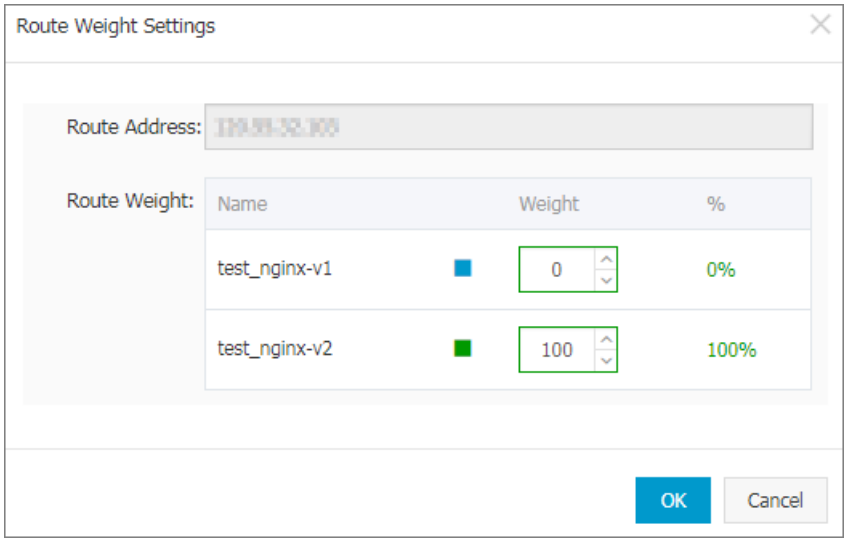
7. Click the **Routes** tab and then click **Set service weight**.

   The Server Load Balancer weight of the old version is 100 and that of the new version is 0.



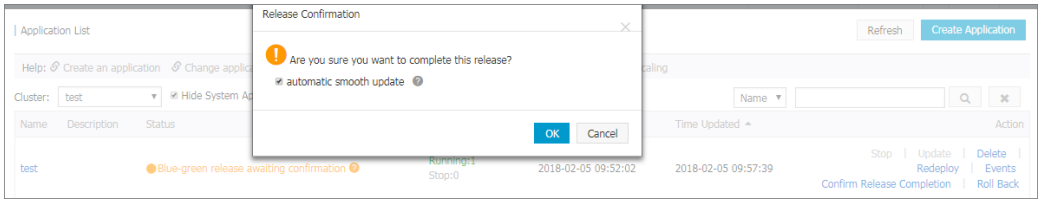
   To realize zero downtime update, set the weight of the new version to 100. Now the new version and old version account for 50% of the weight respectively. Test if both versions have stable traffic.

   > **Note** Adjusting the weights of the new version and old version at the same time might result in the failure of some requests. Therefore, adjust the weights in two steps and only adjust the weight of one version in each step. For example, adjust the weight of the new version from 0 to 100 first, and then adjust the weight of the old version from 100 to 0 after the traffic is stable.
   >
   > Then, adjust the weight of the old version to 0 and that of the new version to 100.



8. You can open a new browser window to access the new version.

9. After the entire release process has been verified, click **Confirm Release Completion** on the Application List page. Select whether or not to automatically perform the **smooth update** in the displayed dialog box and click **OK** to confirm the release before you can release subsequent versions.



Now the service list of the application has been updated and the old service version has been taken offline and deleted.