# Container Service

## User Guide

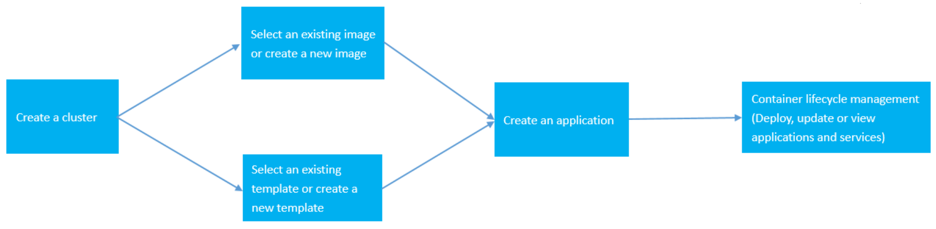
MORE THAN JUST CLOUD | Alibaba Cloud

# User Guide

## Overview

## Workflow

The following flowchart details how to use the Container Service:


Create a cluster
Select an existing image or create a new image
Select an existing template or create a new template
Create an application
Container lifecycle management (Deploy, update or view applications and services)

### Procedure

Step 1: Create a cluster, select the cluster network environment, and set the number of nodes and configuration information of the cluster.

Step 2: Select an image or orchestration template (if your application is composed of services supported by multiple images, you can select a single orchestration template).

Step 3: Create and deploy an application.

Step 4: View the status of the deployed application as well as the relevant service and container information.

## Basic concepts and terms

### Cluster

A cluster describes a collection of cloud resources required to run containers. It can associates with server nodes, Server Load Balancer instances, VPCs, and other cloud resources.

## Node

A node is a server (either a VM instance or a physical server) which is installed with a Docker Engine and is used to deploy and manage clusters. The Agent program of the Container Service is installed in a node and registered to a cluster. The quantity of nodes in a cluster is scalable.

## Container

A container is an instance created using a Docker image. A single node can run multiple containers.

## Image

A Docker image is a standard packaging format of a container application. You can specify an image to deploy container applications. The image may be obtained from the Docker Hub, Alibaba Cloud Container Hub, or your private Registry. An image ID is uniquely identified by the URI of the image repository and the image tag name (the latest tag name is used by default).

## Orchestration template

An orchestration template contains definitions of, and interconnection relationships between a group of container services, and can be used to deploy and manage multiple container applications. The Container Service is compatible with Docker Compose and is scalable.

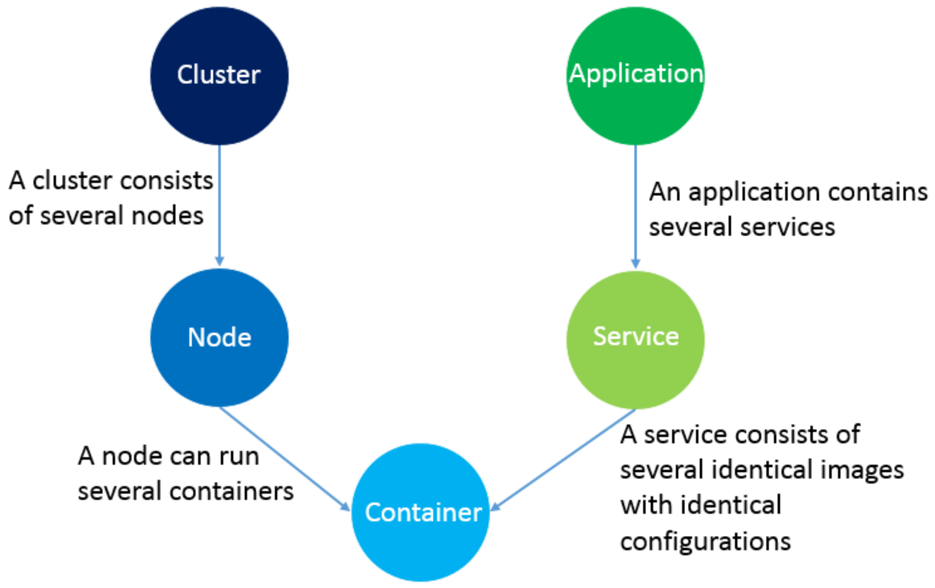
## Application

An application can be created from an image or an orchestration template. Each application can contain one or more container services.

## Service

A service is a group of containers running identical images with identical configurations. It is a scalable microservice.

## Associations


Cluster
A cluster consists of several nodes
Node
A node can run several containers
Container
Application
An application contains several services
Service
A service consists of several identical images with identical configurations

# Cluster management

## Cluster introduction

A cluster refers to a combination of cloud resources that are necessary for the operation of a container. It is associated with a number of ECS nodes, Server Load Balancer, and other cloud resources.

### Create a cluster

You can create a cluster using one of the following two methods:

**Method 1:** Create a cluster and, at the same time, create several ECS instances.

You can directly create a cluster with several new ECS instances through the Container Service. For details, refer to Create a cluster.

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

**Method 2:** Create a zero-node cluster and add an existing ECS instance.

Create a zero-node cluster.

If you have purchased several ECS instances from the ECS service, you can create a zero-node cluster in the Container Service. The operation is similar to **Method 1**, however, you only need to select **Do not add**.

Add an existing ECS instance.

You can add an existing ECS instance to the Container service using one of the following two methods:

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

As this method will reset the image and system disk of the ECS instance, use it with due care. However, the server added using this method is cleaner.

Execute scripts on the ECS instance and manually add it to the cluster.

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

For details, refer to Add an existing ECS instance.

## Manage a cluster

You can search for, expand, connect to, clean up, or delete a cluster.

For more details, refer to:

- Search for a cluster
- Expand a cluster
- Connect to a cluster
- Clean up a cluster disk
- Delete a cluster

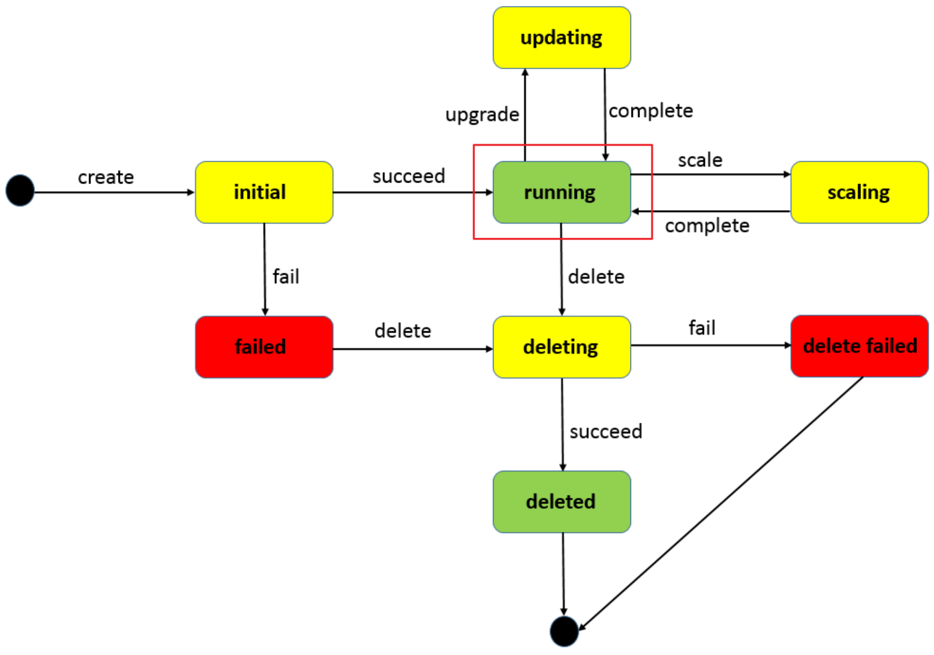
# Lifecycle of a cluster

A complete cluster lifecycle includes the following statuses.

| **Status** | **Description** |
|---|---|
| Inactive | The cluster is waiting to add nodes . |
| 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 nodes of the cluster. |
| Failed | The cluster’ s application for the cloud resources failed. |
| Deleting | The cluster is being deleted. |
| DeleteFailed | Cluster deletion failed. |
| Deleted (invisible to users) | The cluster is successfully deleted. |

## Cluster status flow



You can specify the configuration and the number of ECS instances when creating clusters. You can also create a zero-node cluster, and then bind it with other ECS instances.

**Note:** If you create a zero-node cluster, the cluster will be in the “Inactive” status after it is successfully created. The cluster will be activated (change to the “Running” status) after you add ECS instances to it. For information about how to add ECS instances to the cluster, refer to Add an existing ECS instance.

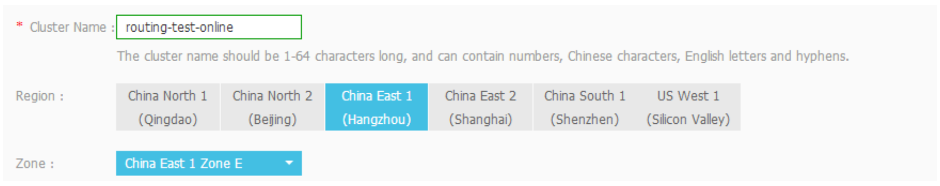
## Constraints

Server Load Balancer instances created with container clusters are only available in Pay-As-You-Go mode.

## Operating procedure

Log on to the Container Service console.

Click **Clusters** in the left navigation pane, and then click **Create Cluster** in the upper-right corner.


* Cluster Name : routing-test-online
The cluster name should be 1-64 characters long, and can contain numbers, Chinese characters, English letters and hyphens.
Region : China North 1 (Qingdao) | China North 2 (Beijing) | China East 1 (Hangzhou) | China East 2 (Shanghai) | China South 1 (Shenzhen) | US West 1 (Silicon Valley)
Zone : China East 1 Zone E

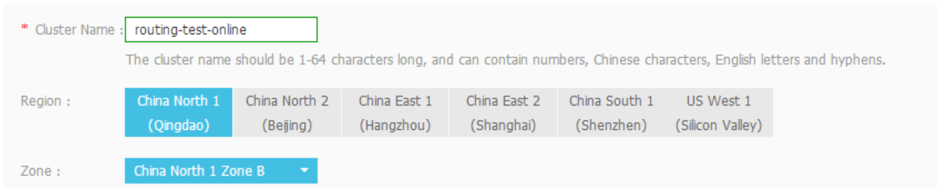
Enter the basic information of the cluster.

- **Cluster Name:** The name of the cluster to be created. It can be 1~64 characters long and be composed of numbers, Chinese characters, English letters and hyphens (-).

  **Note:** The cluster name must be unique under the same user and the same region.

- **Region:** The region which the cluster will be deployed to.
- **Zone:** The zone of the cluster.

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


* Cluster Name : routing-test-online
The cluster name should be 1-64 characters long, and can contain numbers, Chinese characters, English letters and hyphens.
Region : China North 1 (Qingdao) | China North 2 (Beijing) | China East 1 (Hangzhou) | China East 2 (Shanghai) | China South 1 (Shenzhen) | US West 1 (Silicon Valley)
Zone : China North 1 Zone B

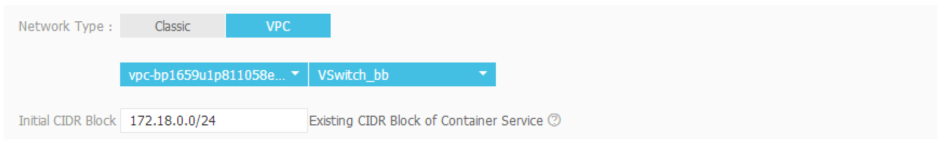
Set the network type of the cluster.

You can set the network types to **Classic** or **VPC**. Corresponding ECS instances and other cloud resources are managed under the corresponding network environment.

If you select **Classic**, no additional configuration is required.

Classic network is a public basic network uniformly planned by Alibaba Cloud. The network address and topology are assigned by Alibaba Cloud and can be used without special configurations.

If you select **VPC**, you need to configure relevant information.

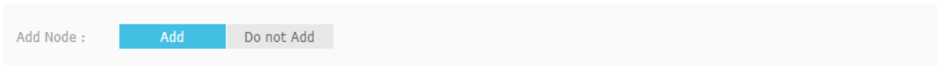


VPC enables you to build an isolated network environment based on Alibaba Cloud. You will have full control over your own virtual network, including a free IP address range, network segment division, route table, gateway configuration, and so on.

You need to specify a VPC, a VSwitchId and the starting network segment of a container (subnet segment to which the Docker container belongs. For the convenience of IP management, the container of each virtual machine belongs to a different network segment, and container subnet segment should not conflict with virtual machine segment).

It is recommended that you build an exclusive VPC/VSwitchId for the container cluster to prevent network conflicts.

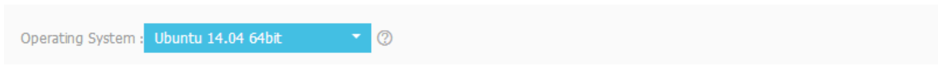
Add nodes.



You can create a cluster with nodes, or create a zero-node cluster and then add existing nodes to the cluster. For information about how to add existing nodes to the cluster, refer to Add an existing ECS instance.

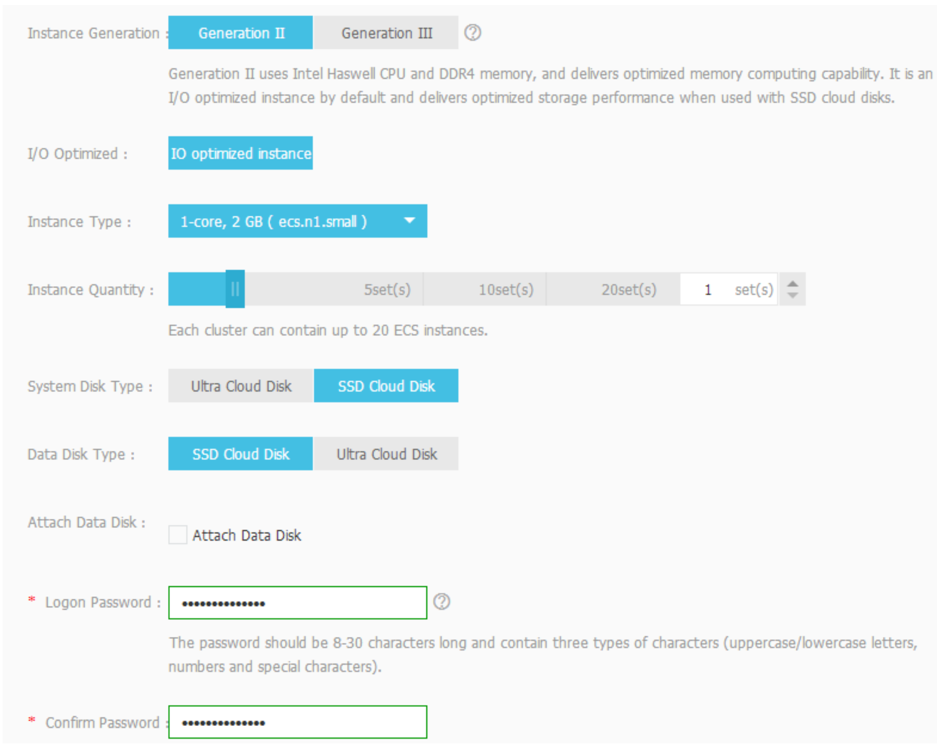
- **Add**

  Set the operating system of the node.



  Operating systems such as 64-bit Ubuntu 14.04 and 64-bit CentOS 7.0 are supported.

  Configure the ECS instance specifications.



Instance Generation : Generation II | Generation III

Generation II uses Intel Haswell CPU and DDR4 memory, and delivers optimized memory computing capability. It is an I/O optimized instance by default and delivers optimized storage performance when used with SSD cloud disks.

I/O Optimized : IO optimized instance

Instance Type : 1-core, 2 GB ( ecs.n1.small )

Instance Quantity : 5set(s) 10set(s) 20set(s) 1 set(s)

Each cluster can contain up to 20 ECS instances.

System Disk Type : Ultra Cloud Disk | SSD Cloud Disk

Data Disk Type : SSD Cloud Disk | Ultra Cloud Disk

Attach Data Disk : Attach Data Disk

* Logon Password : ••••••••••••••

The password should be 8-30 characters long and contain three types of characters (uppercase/lowercase letters, numbers and special characters).

* Confirm Password : ••••••••••••••

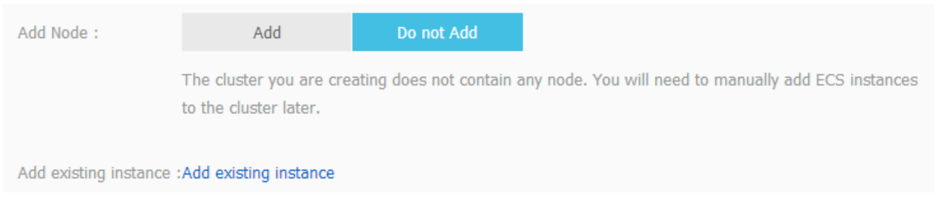
You can specify different instance types and quantities, the capacity of data disk (The ECS instance has a 20GB system disk by default), and logon password.

Note:

i. If you select a data disk, it will be attached to the /var/lib/docker directory, and used for the storage of Docker images and containers.
ii. In terms of performance and management, it is recommended that you attach an independent data disk to the host and manage the persistent data in the container by using Docker volumes.

### Do not Add

You can click **Add existing instance** to add existing ECS instances to the cluster, or you can click **Create Cluster** directly and add existing ECS instances to the cluster after the cluster is created.



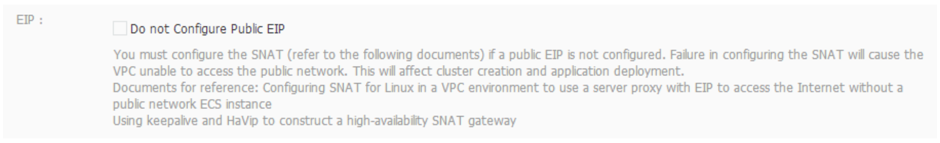
Add Node : Add | Do not Add

The cluster you are creating does not contain any node. You will need to manually add ECS instances to the cluster later.

Add existing instance :Add existing instance

Configure EIP.

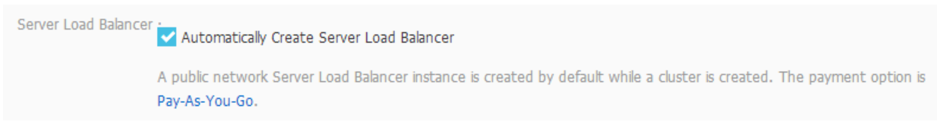
If you set the network type to **VPC**, by default, the Container Service configures an EIP for each ECS instance under the VPC. If this is not required, select **Do not Configure Public EIP**. However, you will then need to configure the SNAT gateway.


EIP :

☐ Do not Configure Public EIP

You must configure the SNAT (refer to the following documents) if a public EIP is not configured. Failure in configuring the SNAT will cause the VPC unable to access the public network. This will affect cluster creation and application deployment.
Documents for reference: Configuring SNAT for Linux in a VPC environment to use a server proxy with EIP to access the Internet without a public network ECS instance
Using keepalive and HaVip to construct a high-availability SNAT gateway

Create a Server Load Balancer instance.


Server Load Balancer :

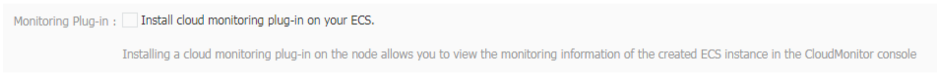
✓ Automatically Create Server Load Balancer

A public network Server Load Balancer instance is created by default while a cluster is created. The payment option is Pay-As-You-Go.

When a cluster is created, a public network Server Load Balancer instance is created by default. You can access the container applications in the cluster through this Server Load Balancer. This is a Pay-As-You-Go Server Load Balancer instance.

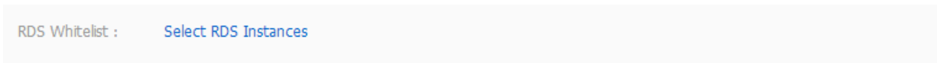
Install cloud monitoring plug-in on your ECS.

Installing a cloud monitoring plug-in on the node allows you to view the monitoring information of the created ECS instance in the CloudMonitor console.


Monitoring Plug-in : ☐ Install cloud monitoring plug-in on your ECS.

Installing a cloud monitoring plug-in on the node allows you to view the monitoring information of the created ECS instance in the CloudMonitor console

Add node IP addresses to RDS instance white lists.


RDS Whitelist : Select RDS Instances

You can add the IP addresses of the created ECS instances to RDS instances so that the ECS instances can access the RDS instances.

Note:

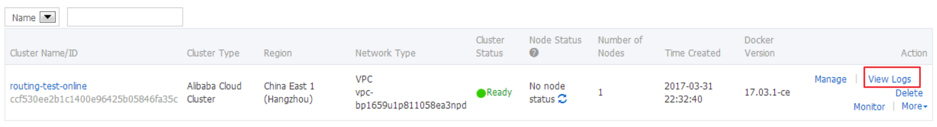
- You can only make this configuration when you select **Add** nodes.
- The ECS instances must be in the same region as the RDS instances.

Click **Create Cluster**.

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

## Subsequent operations

You can locate the cluster created on the **Cluster List** page and click **View Logs** to view the creation process log of the cluster.



| Cluster Name/ID | Cluster Type | Region | Network Type | Cluster Status | Node Status | Number of Nodes | Time Created | Docker Version | Action |
|---|---|---|---|---|---|---|---|---|---|
| routing-test-online ccf530ee2b1c1400e96425b05846fa35c | Alibaba Cloud Cluster | China East 1 (Hangzhou) | VPC vpc-bp1659u1p811058ea3npd | ● Ready | No node status | 1 | 2017-03-31 22:32:40 | 17.03.1-ce | Manage \| View Logs Delete Monitor \| More |

You can create applications in the created cluster. For details, refer to Create an application.

You can add an ECS instance you bought to a specified cluster.

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

- **Add ECS instances automatically:** This method will reset the image and system disk of the ECS instance. You can add one or more instances to the cluster at a time.
- **Manually add:** Manually add the instance to the cluster by executing scripts on the ECS instance. You can only select one ECS at a time.

## Prerequisite

If you have not created a cluster before, you must first create cluster. For information about how to create a cluster, refer to Create a cluster.

## Considerations

The ECS instance to be added must be in the same region and use the same type of network (classic or VPC) as the cluster.

When adding an existing ECS instance, make sure that your ECS instance has a public IP (classic network) or EIP (VPC); otherwise, the ECS instance might not be added successfully.

The Container Service does not support adding ECS instances under a different account.

If you select **Manually Add**, note the following notices:

If you have already installed Docker on your ECS instance, manually adding the ECS instance to a cluster might fail. It is recommended that you use the following commands to uninstall Docker and remove the Docker folders before adding the ECS instance to a cluster.

- **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

The Container Service has special requirement for the operating system of the ECS instance. It is recommended that you use Ubuntu 14.04/16.04 or CentOS 7 operating system.

The Container Service requires that the kernel of the ECS instance should be Linux 3.18 or higher. When adding an ECS instance manually, if your kernel version is not up to the requirement, the ECS instance cannot be added. In this case, you need to upgrade your kernel.

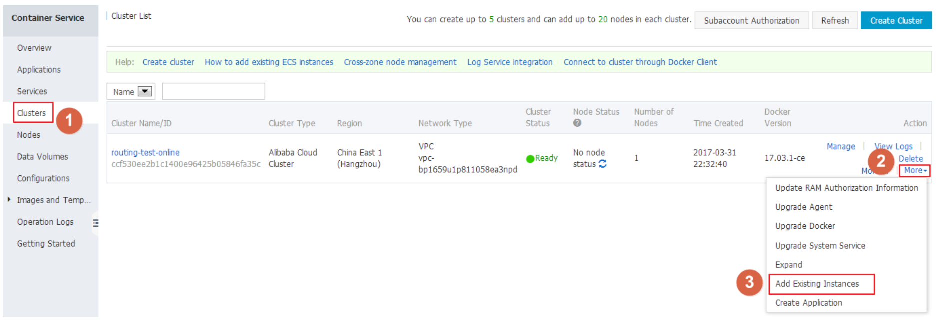
**Note:** To avoid losing your data, you can create a snapshot before upgrading your kernel. For information about how to create a snapshot, refer to Create a snapshot.

## Operating procedure

Log on to the Container Service console.

Click **Clusters** in the left navigation pane.

Locate the target cluster. Click **More** > **Add Existing Instances**.



You can create up to 5 clusters and can add up to 20 nodes in each cluster.

Help: Create cluster How to add existing ECS instances Cross-zone node management Log Service integration Connect to cluster through Docker Client

| Cluster Name/ID | Cluster Type | Region | Network Type | Cluster Status | Node Status | Number of Nodes | Time Created | Docker Version | Action |
|---|---|---|---|---|---|---|---|---|---|
| routing-test-online ccf530ee2b1c1400e96425b05846fa35c | Alibaba Cloud Cluster | China East 1 (Hangzhou) | VPC vpc-bp1659u1p811058ea3npd | ● Ready | No node status | 1 | 2017-03-31 22:32:40 | 17.03.1-ce | Manage \| View Logs \| Delete Mo... \| More |

Add ECS instances.

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

You can add ECS instances using one of the following two methods:

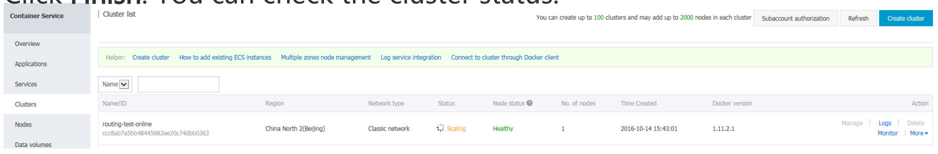
Add ECS instances automatically.

**Note:** This method will reset the image and system disk of the ECS instance; use it with due care. To avoid losing your data, create a snapshot before you add the ECS instance. For information about how to create a snapshot, refer to Create a snapshot.

Select the instances to be added and click **Next**.
You can add one or multiple instances to the cluster at a time.

Set the instance information. Click **Next** and click **Confirm** in the pop-up confirmation dialog box.

Click **Finish**. You can check the cluster status.


Container Service
Overview
Applications
Services
Clusters
Nodes
Data volumes
Cluster list
Subaccount authorization
Refresh
Create cluster
Helper: Create cluster  How to add existing ECS instances  Multiple zones node management  Log service integration  Connect to cluster through Docker client
Name

You can create up to 100 clusters and may add up to 2000 nodes in each cluster

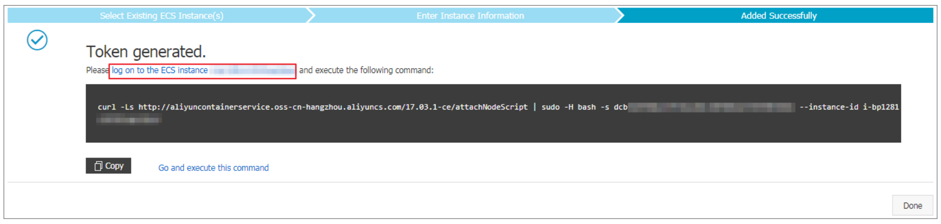
| Name/ID | Region | Network type | Status | Node status | No. of nodes | Time Created | Docker version | Action |
|---|---|---|---|---|---|---|---|---|
| routing-test-online ccc8ab7a5bb48445682ee20c74dbb0363 | China North 2(Beijing) | Classic network | Scaling | Healthy | 1 | 2016-10-14 15:43:01 | 1.11.2.1 | Manage \| Logs \| Delete Monitor \| More |

Manually add the instance to the cluster by executing scripts on the ECS instance.

Select **Manually Add**, select an ECS instance, and click **Next**.
You can only select one ECS at a time.

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

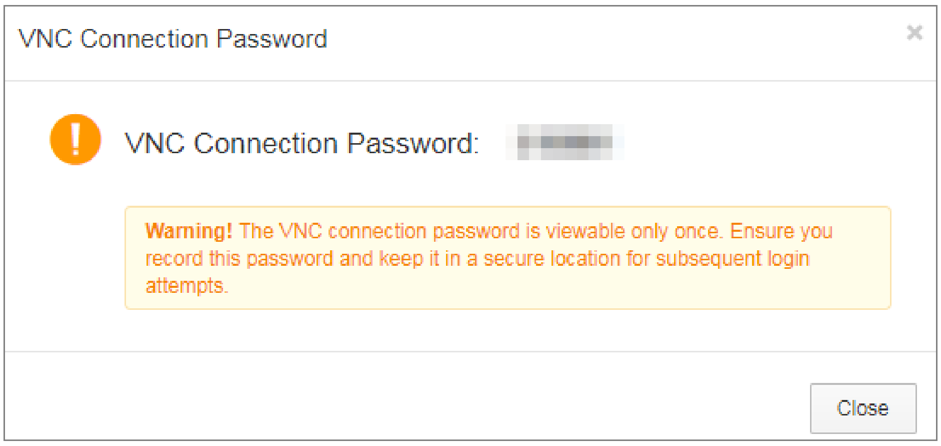
Click **log on to the ECS instance xxxxxx**.


Select Existing ECS Instance(s)
Enter Instance Information
Added Successfully
Copy
Go and execute this command
Done

Token generated.
Please log on to the ECS instance [redacted] and execute the following command:

```
curl -Ls http://aliyuncontainerservice.oss-cn-hangzhou.aliyuncs.com/17.03.1-ce/attachNodeScript | sudo -H bash -s dcb[redacted] --instance-id i-bp1281
[redacted]
```

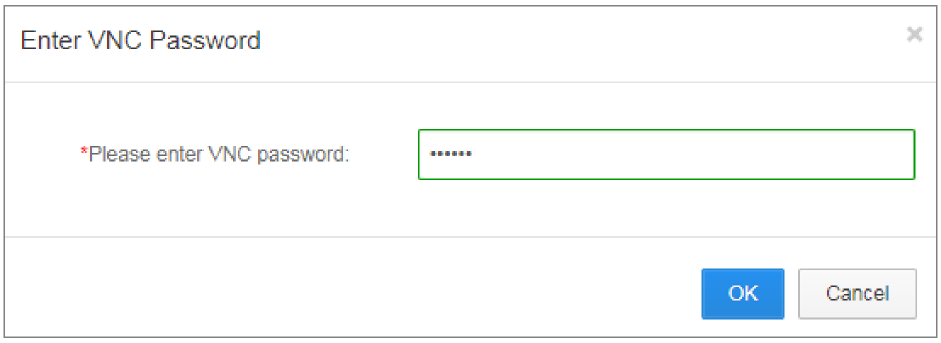
In the pop-up dialog box, copy the VNC password (the password used to connect to the ECS instance through a remote terminal) and click **Close**.


VNC Connection Password

VNC Connection Password: [redacted]

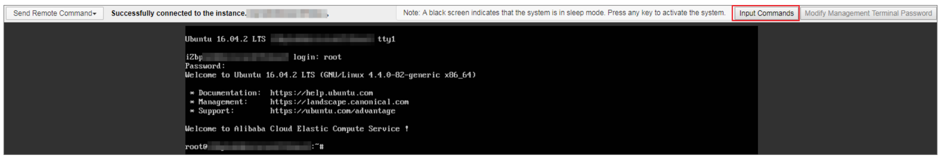
**Warning!** The VNC connection password is viewable only once. Ensure you record this password and keep it in a secure location for subsequent login attempts.

In the pop-up dialog box, enter the VNC password and click **OK**.


Enter VNC Password

*Please enter VNC password: ••••••

Enter the logon account and password of the ECS instance, and press Enter to log on to the ECS instance.

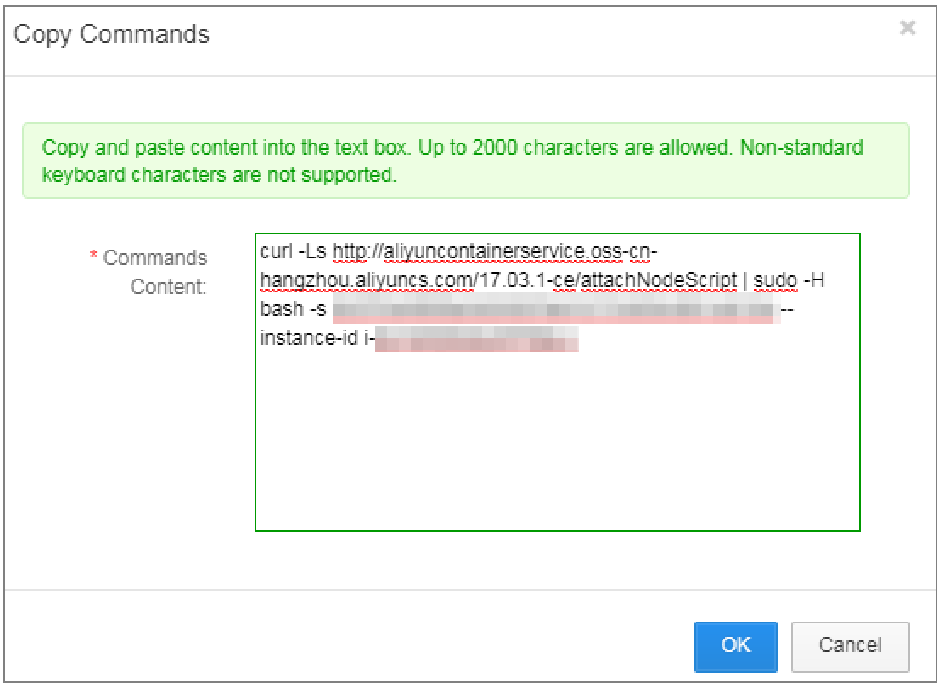


Successfully connected to the instance. [redacted].

Note: A black screen indicates that the system is in sleep mode. Press any key to activate the system.

```
Ubuntu 16.04.2 LTS [redacted] tty1

iZbp[redacted] login: root
Password:
Welcome to Ubuntu 16.04.2 LTS (GNU/Linux 4.4.0-82-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

Welcome to Alibaba Cloud Elastic Compute Service !

root@[redacted]:~#
```

Click **Input Commands**, paste the above script into the pop-up dialog box, click **OK** and press Enter.



Copy Commands

Copy and paste content into the text box. Up to 2000 characters are allowed. Non-standard keyboard characters are not supported.

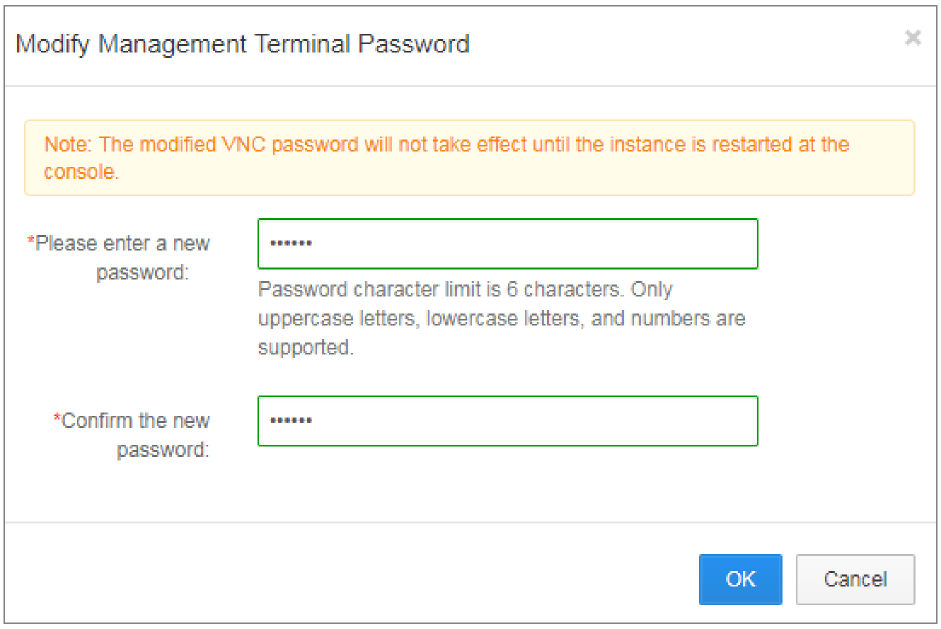
* Commands Content:

```
curl -Ls http://aliyuncontainerservice.oss-cn-
hangzhou.aliyuncs.com/17.03.1-ce/attachNodeScript | sudo -H
bash -s [redacted] --
instance-id i-[redacted]
```

The system executes the script. Wait until the script is successfully executed. A success prompt is displayed. Now, this ECS is successfully added.

## Related operation

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



Modify Management Terminal Password

Note: The modified VNC password will not take effect until the instance is restarted at the console.

*Please enter a new password: ••••••

Password character limit is 6 characters. Only uppercase letters, lowercase letters, and numbers are supported.

*Confirm the new password: ••••••

# Manage cross-zone nodes

To provide high availability of applications, you may select to distribute multiple nodes in different zones when creating a cluster.

First, create a cluster with one node (or directly create a zero-node cluster). After the cluster is created, add the nodes of different zones by expanding the cluster or adding existing ECS instances.

Note:

- Nodes added through expanding 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.

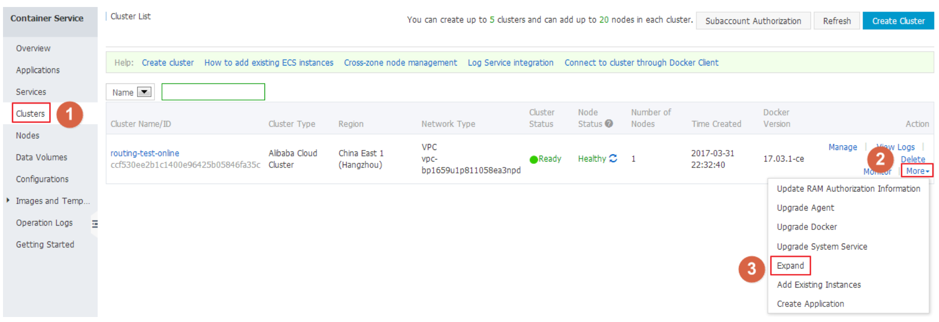
## Add nodes of different zones through expanding

### Operating procedure

Log on to the Container Service console.

Click **Clusters** in the left navigation pane.

Locate the cluster to be expanded, click **More** > **Expand**.

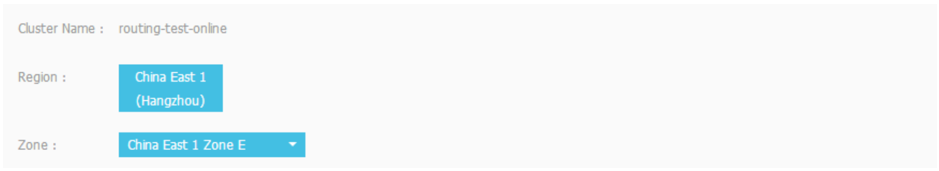


You can create up to 5 clusters and can add up to 20 nodes in each cluster.

| Cluster Name/ID | Cluster Type | Region | Network Type | Cluster Status | Node Status | Number of Nodes | Time Created | Docker Version | Action |
|---|---|---|---|---|---|---|---|---|---|
| routing-test-online ccf530ee2b1c1400e96425b05846fa35c | Alibaba Cloud Cluster | China East 1 (Hangzhou) | VPC vpc-bp1659u1p811058ea3npd | ● Ready | Healthy | 1 | 2017-03-31 22:32:40 | 17.03.1-ce | Manage \| View Logs \| Delete Monitor \| More |

In the pop-up dialog box, configure the specifications of the new node.

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



Cluster Name : routing-test-online

Region : China East 1 (Hangzhou)

Zone : China East 1 Zone E

Click **Expand**. The new node is added to the cluster.

Repeat the steps above to create and add nodes of different zones to the cluster.

## Add nodes of different zones through adding existing ECS instances

### Prerequisite

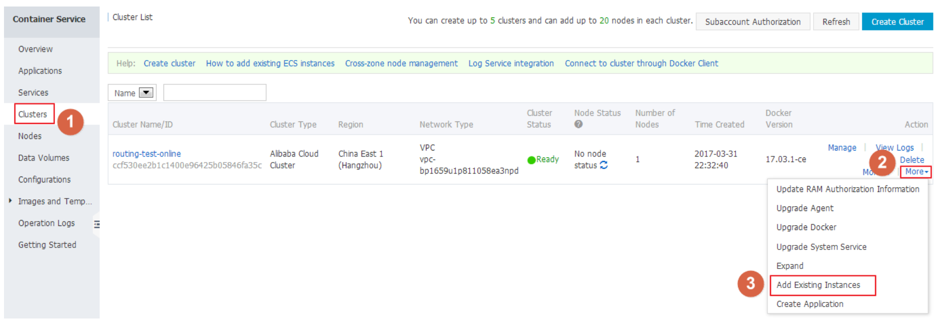
To add nodes by using this method, purchase ECS instances from the ECS purchase page first, and select different zones for them during purchase.

### Operating procedure

Log on to the Container Service console.

Click **Clusters** in the left navigation pane.

Locate the target cluster, click **More** > **Add Existing Instance**.



You can create up to 5 clusters and can add up to 20 nodes in each cluster.

Help: Create cluster How to add existing ECS instances Cross-zone node management Log Service integration Connect to cluster through Docker Client

| Cluster Name/ID | Cluster Type | Region | Network Type | Cluster Status | Node Status | Number of Nodes | Time Created | Docker Version | Action |
|---|---|---|---|---|---|---|---|---|---|
| routing-test-online ccf530ee2b1c1400e96425b05846fa35c | Alibaba Cloud Cluster | China East 1 (Hangzhou) | VPC vpc-bp1659u1p811058ea3npd | ● Ready | No node status | 1 | 2017-03-31 22:32:40 | 17.03.1-ce | Manage \| View Logs \| Delete \| More |

Select ECS instances of different zones and add them manually or automatically.

For the details, refer to Add an existing ECS instance.

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

# Set the root domain name of a cluster

In Create a Nginx Web server from an image, when you set the web routing rules, you only need to enter the prefix nginx of the domain name. Then, you obtain the domain name nginx.$cluster_id.$region_id.alicontainer.com. You can replace this domain name by setting a root domain name (example.com is used in this example) of the cluster. When you redeploy the service nginx, the domain name changes from nginx.c2818a77aac20428488694c0cd1600e6e.cn-shenzhen.alicontainer.com to nginx.example.com, which makes it convenient for you to access the cluster applications with your own root domain name.

The following example details the complete process.

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

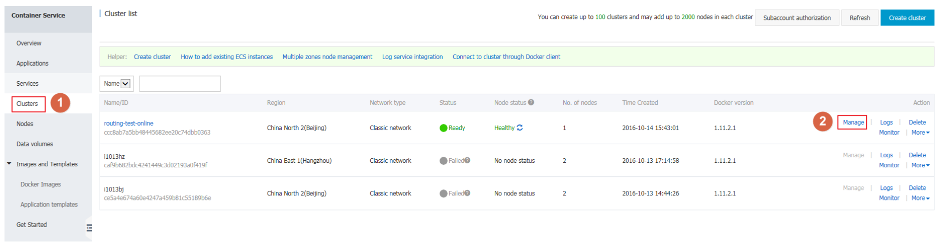
## Operating procedure

Bind a Server Load Balancer instance.

Log on to the Container Service console.

Click **Clusters** in the left navigation pane.

Locate the cluster to be configured (**routing-test-online** is used in this example), and click **Manage**.

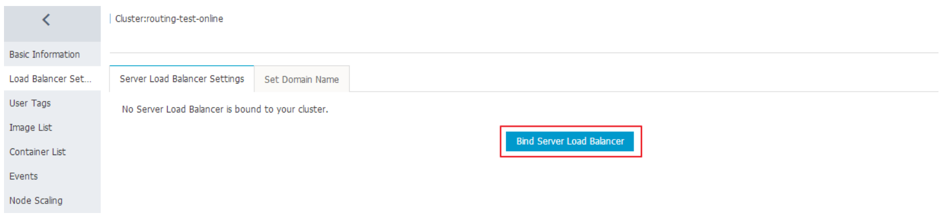


You can create up to 100 clusters and may add up to 2000 nodes in each cluster

| Name/ID | Region | Network type | Status | Node status | No. of nodes | Time Created | Docker version | Action |
|---|---|---|---|---|---|---|---|---|
| routing-test-online ccc8ab7a5bb48445682ee20c74dbb0363 | China North 2(Beijing) | Classic network | Ready | Healthy | 1 | 2016-10-14 15:43:01 | 1.11.2.1 | Manage \| Logs \| Delete Monitor \| More |
| i1013hz caf9b682bdc4241449c3d02193a0f419f | China East 1(Hangzhou) | Classic network | Failed | No node status | 2 | 2016-10-13 17:14:58 | 1.11.2.1 | Manage \| Logs \| Delete Monitor \| More |
| i1013bj ce5a4e674a60e4247a459b81c55189b6e | China North 2(Beijing) | Classic network | Failed | No node status | 2 | 2016-10-13 14:44:26 | 1.11.2.1 | Manage \| Logs \| Delete Monitor \| More |

Click **Load Balancer Settings** in the left navigation pane.
If no Server Load Balancer instance is bound, log on to the Server Load Balancer console and create a Server Load Balancer instance; and then, return to this page and bind it.

**Note:** Server Load Balancer instance bound with an intranet is supported.



Cluster:routing-test-online

Basic Information
Load Balancer Set...
User Tags
Image List
Container List
Events
Node Scaling

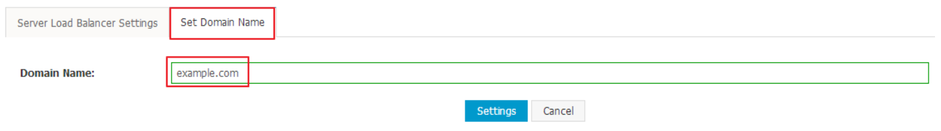
Server Load Balancer Settings | Set Domain Name

No Server Load Balancer is bound to your cluster.

Bind Server Load Balancer

Set the domain name.

Click **Set Domain Name** and enter the root domain name you bought. In this example, example.com is used.



Server Load Balancer Settings | Set Domain Name

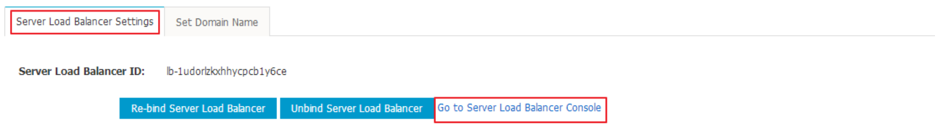
**Domain Name:** example.com

Settings | Cancel

Click **Settings**.

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

Click **Server Load Balancer Settings** and click **Go to Server Load Balancer Console**.

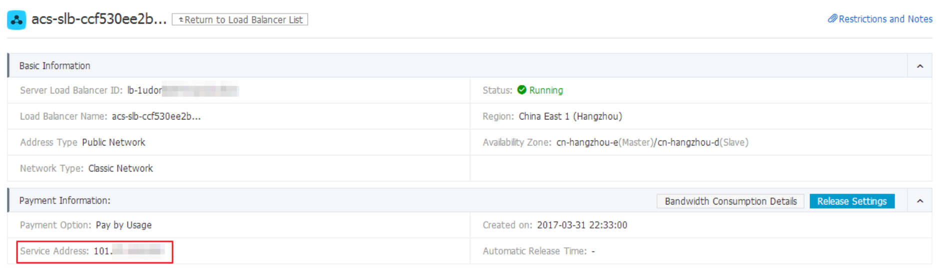


Server Load Balancer Settings | Set Domain Name

**Server Load Balancer ID:** lb-1udorlzkxhhycpcb1y6ce

Re-bind Server Load Balancer | Unbind Server Load Balancer | Go to Server Load Balancer Console

Find the VIP address of the bound Server Load Balancer instance.



acs-slb-ccf530ee2b... Return to Load Balancer List | Restrictions and Notes

Basic Information

| | |
|---|---|
| Server Load Balancer ID: lb-1udor[redacted] | Status: Running |
| Load Balancer Name: acs-slb-ccf530ee2b... | Region: China East 1 (Hangzhou) |
| Address Type Public Network | Availability Zone: cn-hangzhou-e(Master)/cn-hangzhou-d(Slave) |
| Network Type: Classic Network | |

Payment Information: Bandwidth Consumption Details | Release Settings

| | |
|---|---|
| Payment Option: Pay by Usage | Created on: 2017-03-31 22:33:00 |
| Service Address: 101.[redacted] | Automatic Release Time: - |

Ask your DNS resolver service provider to resolve your domain (*.example.com in this example) to the Server Load Balancer VIP address.

Redeploy the web service.

Redeploy your application. The web service access endpoint under the application will change to $your_domain_prefix.example.com.

Access the latest access endpoint http://wordpress.example.com, you will see the "Hello World" page of WordPress.

# Download cluster certificate

With the downloaded certificate, you can connect to the endpoint exposed from the cluster through Docker Swarm API or Docker Client.

## Operating procedure

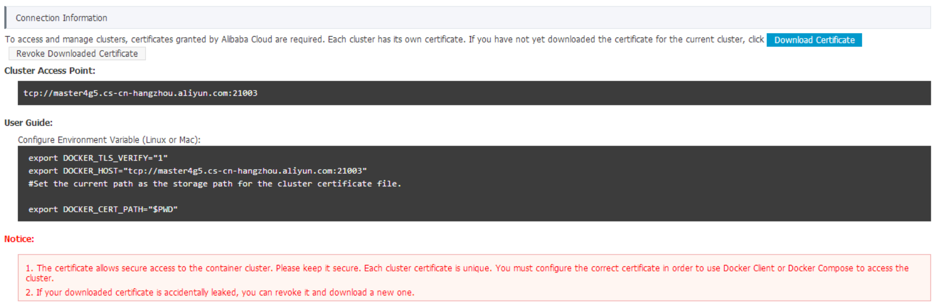
Obtain the access address.

i. Log on to the Container Service console.

Click **Clusters** in the left navigation pane.

Select a cluster in the cluster list and click **Manage**.

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



Connection Information

To access and manage clusters, certificates granted by Alibaba Cloud are required. Each cluster has its own certificate. If you have not yet downloaded the certificate for the current cluster, click Download Certificate

Revoke Downloaded Certificate

**Cluster Access Point:**

```
tcp://master4g5.cs-cn-hangzhou.aliyun.com:21003
```

**User Guide:**

Configure Environment Variable (Linux or Mac):

```
export DOCKER_TLS_VERIFY="1"
export DOCKER_HOST="tcp://master4g5.cs-cn-hangzhou.aliyun.com:21003"
#Set the current path as the storage path for the cluster certificate file.

export DOCKER_CERT_PATH="$PWD"
```

**Notice:**

1. The certificate allows secure access to the container cluster. Please keep it secure. Each cluster certificate is unique. You must configure the correct certificate in order to use Docker Client or Docker Compose to access the cluster.
2. If your downloaded certificate is accidentally leaked, you can revoke it and download a new one.

Download and save the certificate.

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

Click **Download Certificate** in the cluster details page to download the certificate which is contained in the certFile.zip file. 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 the ~/.acs/certs/ClusterName/ directory is recommended for easy management.

```
mkdir ~/.acs/certs/ClusterName/  #Replace ClusterName with your cluster name
cd ~/.acs/certs/ClusterName/
```

```
cp /path/to/certFile.zip .
unzip certFile.zip
```

The certFile.zip file contains ca.pem, cert.pem, and key.pem files.

You can scale out your cluster according to your business needs.

**Note:** Instances added to the cluster are Pay-As-You-Go instances.

## Limitation

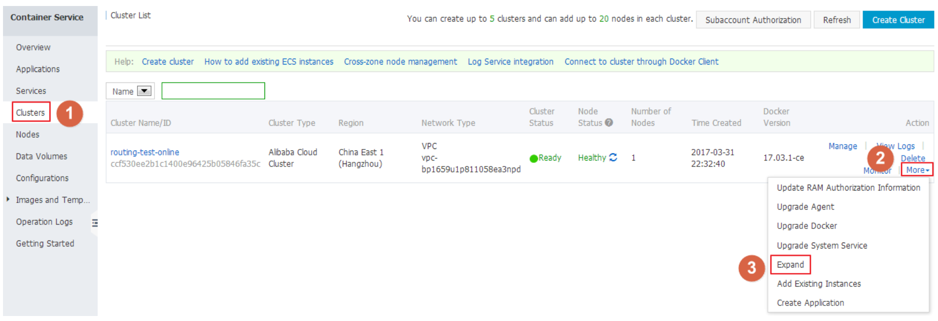
A cluster can contain up to 20 nodes.

## Operating procedure

Log on to the Container Service console.

Click **Clusters** in the left navigation pane.

Locate the cluster to be expanded. Click **More** > **Expand**.



You can create up to 5 clusters and can add up to 20 nodes in each cluster.

| Cluster Name/ID | Cluster Type | Region | Network Type | Cluster Status | Node Status | Number of Nodes | Time Created | Docker Version | Action |
|---|---|---|---|---|---|---|---|---|---|
| routing-test-online ccf530ee2b1c1400e96425b05846fa35c | Alibaba Cloud Cluster | China East 1 (Hangzhou) | VPC vpc-bp1659u1p811058ea3npd | ● Ready | Healthy | 1 | 2017-03-31 22:32:40 | 17.03.1-ce | Manage \| View Logs \| Delete Monitor \| More |

In the pop-up dialog box, configure the specifications of the new node.

You can select the number of server nodes to be added and the corresponding specifications.

Click **Expand**.

# Search for a cluster

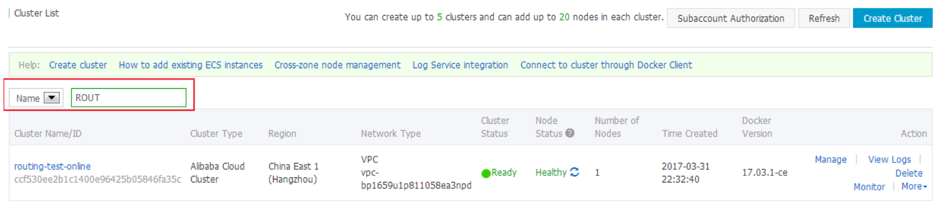
## Operating procedure

Log on to the Container Service console.

Click **Clusters** in the left navigation pane.

Enter the cluster name or keywords of the cluster name in the search box. Clusters with the keywords in their names are displayed.

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


Cluster List

You can create up to 5 clusters and can add up to 20 nodes in each cluster.

Subaccount Authorization
Refresh
Create Cluster
Help: Create cluster
How to add existing ECS instances
Cross-zone node management
Log Service integration
Connect to cluster through Docker Client

Name: ROUT

| Cluster Name/ID | Cluster Type | Region | Network Type | Cluster Status | Node Status | Number of Nodes | Time Created | Docker Version | Action |
|---|---|---|---|---|---|---|---|---|---|
| routing-test-online ccf530ee2b1c1400e96425b05846fa35c | Alibaba Cloud Cluster | China East 1 (Hangzhou) | VPC vpc-bp1659u1p811058ea3npd | Ready | Healthy | 1 | 2017-03-31 22:32:40 | 17.03.1-ce | Manage \| View Logs \| Delete Monitor \| More |

# Delete a cluster

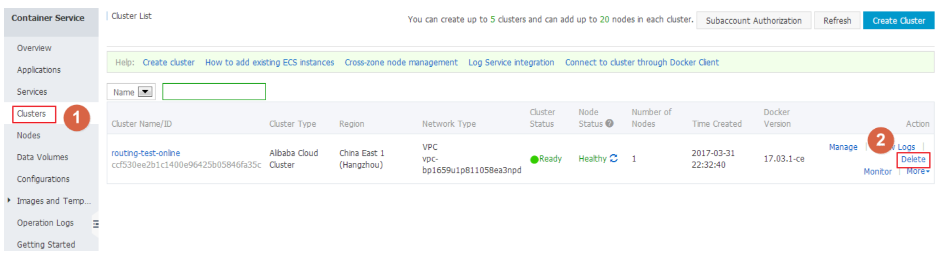
You can delete clusters from the Container Service. When you delete a cluster, the associated cloud resources such as ECS instances and Server Load Balancer instances as well as all the applications and services running on the instances will also be deleted. Use this operation with due care.

## Operating procedure

Log on to the Container Service console.

Click **Clusters** in the left navigation pane.

Locate the cluster to be deleted and click **Delete**.



You can create up to 5 clusters and can add up to 20 nodes in each cluster.

| Cluster Name/ID | Cluster Type | Region | Network Type | Cluster Status | Node Status | Number of Nodes | Time Created | Docker Version | Action |
|---|---|---|---|---|---|---|---|---|---|
| routing-test-online ccf530ee2b1c1400e96425b05846fa35c | Alibaba Cloud Cluster | China East 1 (Hangzhou) | VPC vpc-bp1659u1p811058ea3npd | ● Ready | Healthy | 1 | 2017-03-31 22:32:40 | 17.03.1-ce | Manage Logs Delete Monitor More |

In the pop-up dialog box, select whether to keep the Server Load Balancer instances and click **OK**.

# Clean up a cluster disk

Disk cleanup operation cleans up dirty data on each server in the cluster of users. Dirty data is limited to:

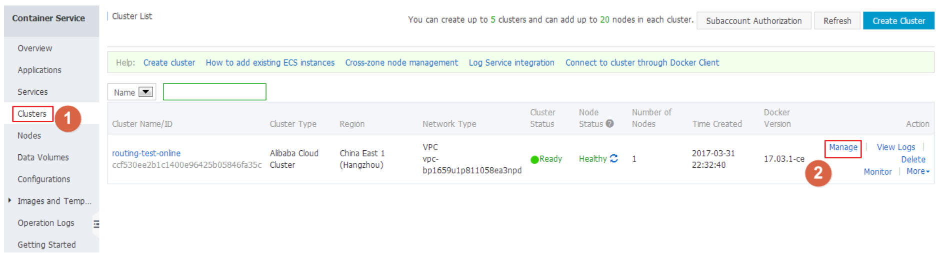
- Docker images downloaded to a local position but not used
- Volume directory once attached to a container but not cleaned up after the destruction of the container

## Operating procedure

Log on to the Container Service console.

Click **Clusters** in the left navigation pane.

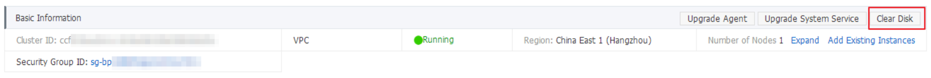
Locate the cluster to be cleaned up and click **Manage**.



You can create up to 5 clusters and can add up to 20 nodes in each cluster.

| Cluster Name/ID | Cluster Type | Region | Network Type | Cluster Status | Node Status | Number of Nodes | Time Created | Docker Version | Action |
|---|---|---|---|---|---|---|---|---|---|
| routing-test-online ccf530ee2b1c1400e96425b05846fa35c | Alibaba Cloud Cluster | China East 1 (Hangzhou) | VPC vpc-bp1659u1p811058ea3npd | ● Ready | Healthy | 1 | 2017-03-31 22:32:40 | 17.03.1-ce | Manage View Logs Delete Monitor More |

Click **Clear Disk**.



| | | | | |
|---|---|---|---|---|
| Cluster ID: ccf[redacted] | VPC | ● Running | Region: China East 1 (Hangzhou) | Number of Nodes 1 Expand Add Existing Instances |
| Security Group ID: sg-bp[redacted] | | | | |

# Upgrade Agent

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

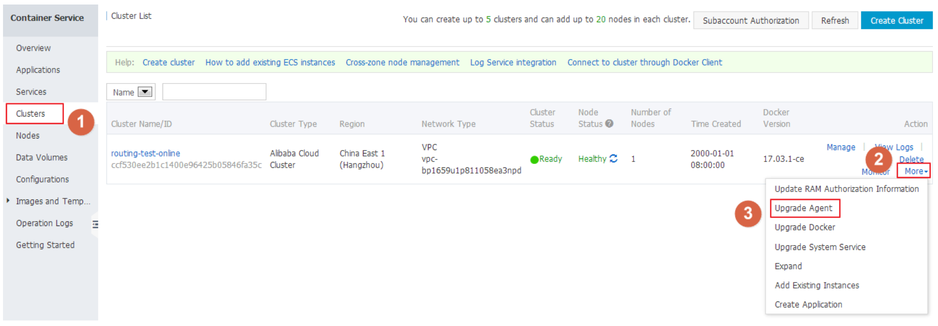
New functions are regularly added to the Container Service. If you need the latest functions, update the Agent of the cluster.

## Operating procedure

Log on to the Container Service console.

Click **Clusters** in the left navigation pane.

Locate the cluster whose Agent is to be updated, click **More** > **Upgrade Agent**.



You can create up to 5 clusters and can add up to 20 nodes in each cluster.

| Cluster Name/ID | Cluster Type | Region | Network Type | Cluster Status | Node Status | Number of Nodes | Time Created | Docker Version | Action |
|---|---|---|---|---|---|---|---|---|---|
| routing-test-online ccf530ee2b1c1400e96425b05846fa35c | Alibaba Cloud Cluster | China East 1 (Hangzhou) | VPC vpc-bp1659u1p811058ea3npd | ● Ready | Healthy | 1 | 2000-01-01 08:00:00 | 17.03.1-ce | Manage \| View Logs \| Delete Monitor \| More |

In the pop-up dialog box, click **OK**.

# Upgrade Docker Daemon

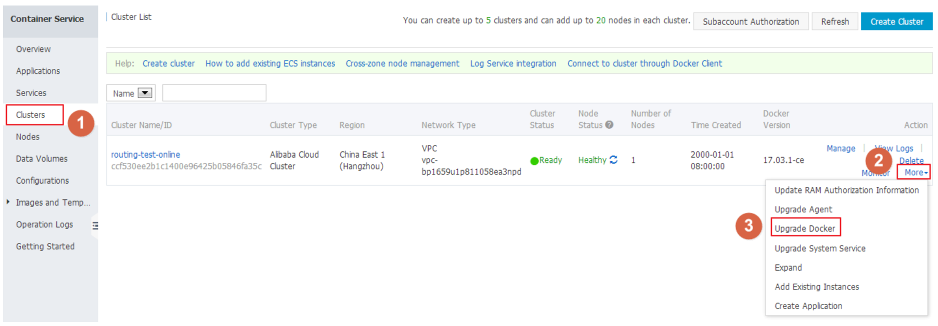
Standard Docker Daemon is installed on each server in the cluster for container management.

## Operating procedure

Log on to the Container Service console.

Click **Clusters** in the left navigation pane.

Locate the cluster whose Docker Deamon you want to update, click **Upgrade** in the Docker version column or click **More** > **Upgrade Docker**.



You can create up to 5 clusters and can add up to 20 nodes in each cluster.

Help: Create cluster How to add existing ECS instances Cross-zone node management Log Service integration Connect to cluster through Docker Client

| Cluster Name/ID | Cluster Type | Region | Network Type | Cluster Status | Node Status | Number of Nodes | Time Created | Docker Version | Action |
|---|---|---|---|---|---|---|---|---|---|
| routing-test-online ccf530ee2b1c1400e96425b05846fa35c | Alibaba Cloud Cluster | China East 1 (Hangzhou) | VPC vpc-bp1659u1p811058ea3npd | ●Ready | Healthy | 1 | 2000-01-01 08:00:00 | 17.03.1-ce | Manage \| View Logs \| Delete Monitor More |

If your Agent version is too old, you need to first upgrade your Agent. Click **Upgrade Agent** and follow the instructions.

If your Agent is the latest version, upgrade Docker directly.

You can upgrade Docker using one of the following methods:

### Upgrade Directly

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

### Back up Snapshot before Upgrade

It is recommended that you back up the snapshot before upgrading Docker, so that you can recover Docker if a fault occurs during the upgrade process.

Click **Back up Snapshot before Upgrade**, and then the system will call the ECS OpenApI to take snapshots of the nodes in the cluster.

Taking snapshots may take some time. After all snapshots are taken, the system enters the Docker Engine upgrading process automatically.

If the system fails to take the snapshot, the **Continue** and **Quit** buttons become available. You can click **Continue** to enter the Docker Engine upgrading process, or click **Quit** to give up the upgrade.

At this point, return to the **Cluster list** page and you can see that the cluster operated just now 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.

# Upgrade system services

The system services of a cluster are used to address general services necessary for applications. The system services include log service acslogging, routing service acsrouting, monitor service acsmonitoring and volume service acsvolumedriver.

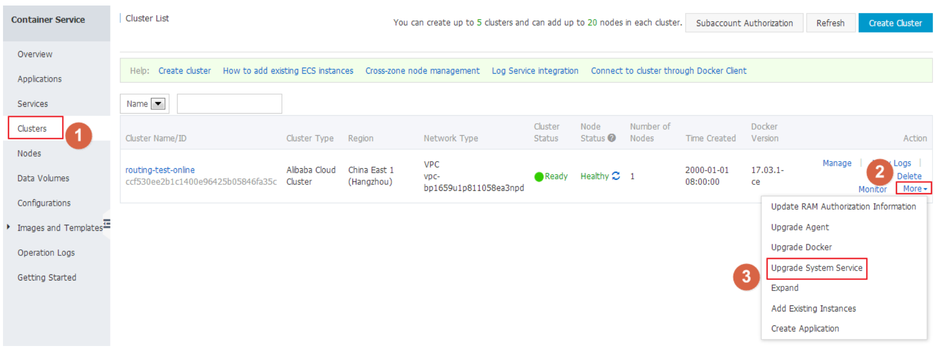
**Note:** During the upgrade of the system services of a cluster, your application or service might be temporarily inaccessible or abnormal. Be careful about the upgrade operation. It is recommended that you upgrade the system services when the access traffic is low or at the maintenance time.

## Operating procedure

Log on to the Container Service console.

Click **Clusters** in the left navigation pane.

Locate the cluster whose system services you want to upgrade, click **More** > **Upgrade System Service**.



You can create up to 5 clusters and can add up to 20 nodes in each cluster.

| Cluster Name/ID | Cluster Type | Region | Network Type | Cluster Status | Node Status | Number of Nodes | Time Created | Docker Version | Action |
|---|---|---|---|---|---|---|---|---|---|
| routing-test-online ccf530ee2b1c1400e96425b05846fa35c | Alibaba Cloud Cluster | China East 1 (Hangzhou) | VPC vpc-bp1659u1p811058ea3npd | Ready | Healthy | 1 | 2000-01-01 08:00:00 | 17.03.1-ce | Manage Logs Delete Monitor More |

In the pop-up dialog box, select the system service to be upgraded and click **Upgrade**.

For example, select **Routing Service** (corresponding to acsrouting; note that upgrade will temporarily impact your access to applications), **Volume Service** (corresponding to acsvolumedriver; note that upgrade will temporarily impact your associated application functions).

Click **Applications** in the left navigation pane and select the target cluster.

You will find that the system services are being upgraded.

Upon completion of the upgrade, the affected services resume normal functioning.

## Node management

You can remove nodes from a cluster. The machine information of the removed node is no longer displayed in the node list.

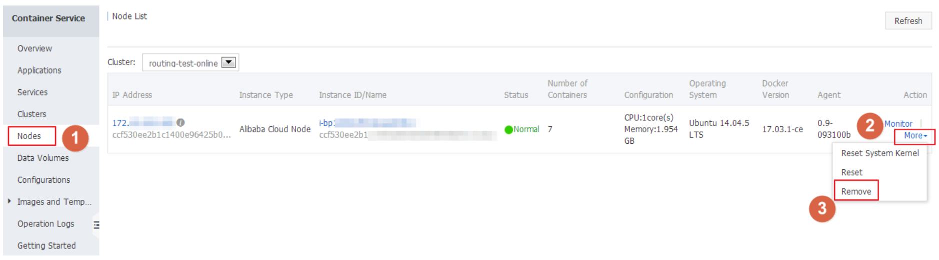
### Operating procedure

Log on to the Container Service console.

Click **Nodes** in the left navigation bar.

Select the cluster of the node to be removed.

Locate the node you want to remove, click **More** > **Remove**.



Cluster: routing-test-online

| IP Address | Instance Type | Instance ID/Name | Status | Number of Containers | Configuration | Operating System | Docker Version | Agent | Action |
|---|---|---|---|---|---|---|---|---|---|
| 172.[redacted] ccf530ee2b1c1400e96425b0... | Alibaba Cloud Node | i-bp[redacted] ccf530ee2b1[redacted] | Normal | 7 | CPU:1core(s) Memory:1.954 GB | Ubuntu 14.04.5 LTS | 17.03.1-ce | 0.9-093100b | Monitor More |

In the pop-up confirmation dialog box, select **Migrate Container** based on your need and click **Confirm**.

## Container Migration

- You can select **Migrate Containers** when you remove or reset a node.
- If you select **Migrate Containers**, all containers on the node are migrated to other machines in the cluster.
- Container migration causes loss of the data in local volumes. Back up the data before container migration. The node reset process stops if container migration fails. In this case,

you must manually reset the node again. During the second reset, you can deselect **Migrate Containers** to forcibly reset the node.
- You can click **Clusters** > **Manage** > **Events** to view the container migration log.

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 are lost. The reset machine is re-added to the cluster.

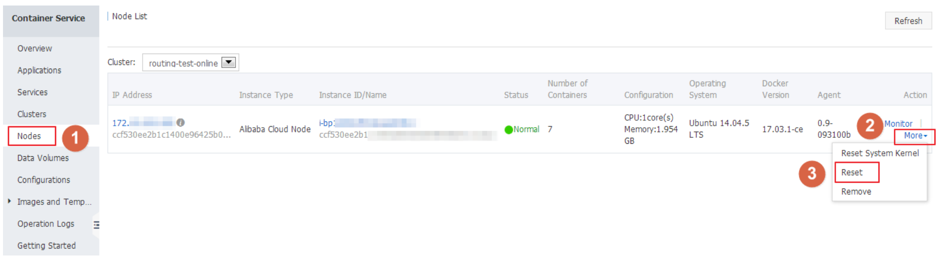
## Operating procedure

Log on to the Container Service console.

Click **Nodes** in the left navigation bar.

Select the cluster of the node to be reset.

Locate the node you want to reset, click **More** > **Reset**.

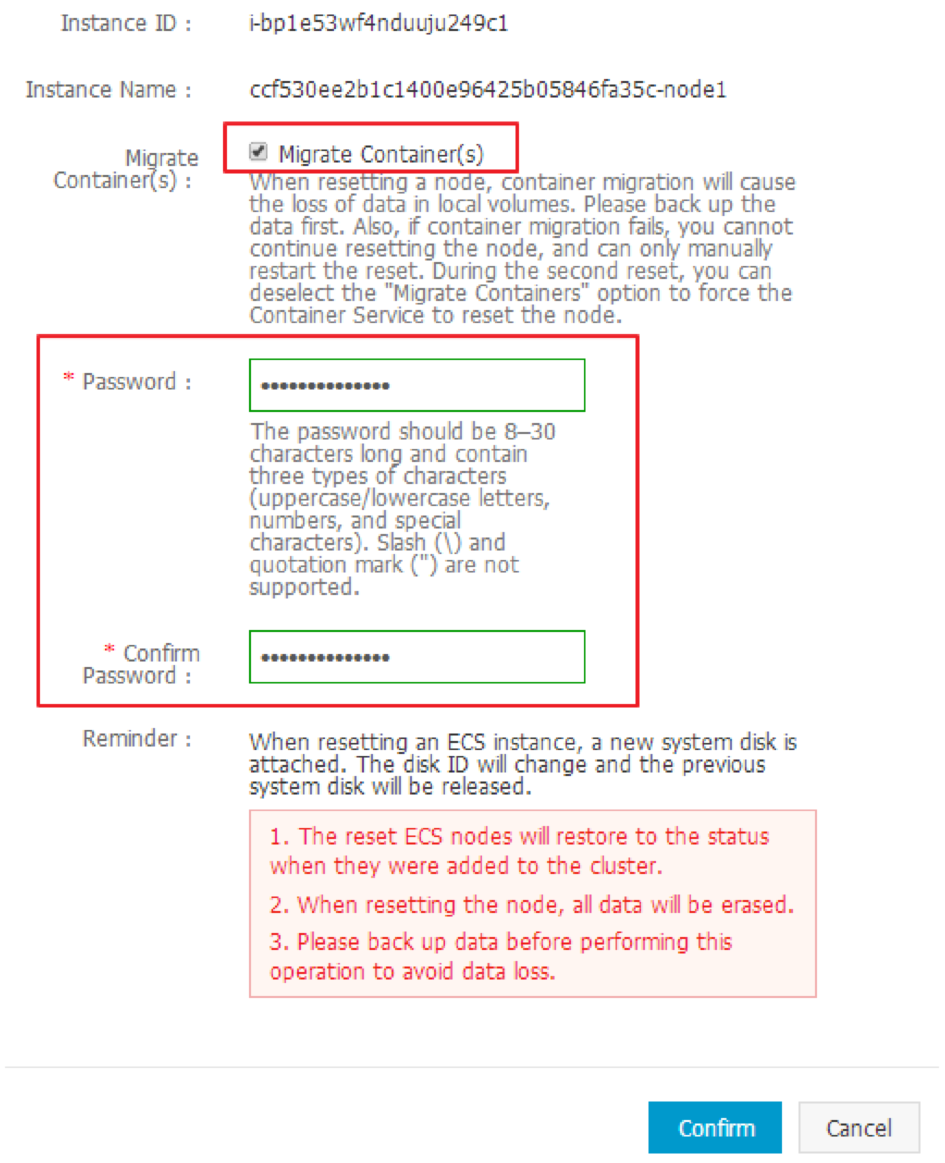


Cluster: routing-test-online

| IP Address | Instance Type | Instance ID/Name | Status | Number of Containers | Configuration | Operating System | Docker Version | Agent | Action |
|---|---|---|---|---|---|---|---|---|---|
| 172.[redacted] ccf530ee2b1c1400e96425b0... | Alibaba Cloud Node | i-bp[redacted] ccf530ee2b1[redacted] | ●Normal | 7 | CPU:1core(s) Memory:1.954 GB | Ubuntu 14.04.5 LTS | 17.03.1-ce | 0.9-093100b | Monitor More |

In the pop-up confirmation dialog box, select **Migrate Containers** based on your need, enter the instance logon password, and click **Confirm**.

For notes about container migration, refer to **Container Migration** below.



Instance ID : i-bp1e53wf4nduuju249c1

Instance Name : ccf530ee2b1c1400e96425b05846fa35c-node1

Migrate Container(s) : ✓ Migrate Container(s)
When resetting a node, container migration will cause the loss of data in local volumes. Please back up the data first. Also, if container migration fails, you cannot continue resetting the node, and can only manually restart the reset. During the second reset, you can deselect the "Migrate Containers" option to force the Container Service to reset the node.

* Password : ••••••••••••••
The password should be 8–30 characters long and contain three types of characters (uppercase/lowercase letters, numbers, and special characters). Slash (\) and quotation mark (") are not supported.

* Confirm Password : ••••••••••••••

Reminder : When resetting an ECS instance, a new system disk is attached. The disk ID will change and the previous system disk will be released.

1. The reset ECS nodes will restore to the status when they were added to the cluster.
2. When resetting the node, all data will be erased.
3. Please back up data before performing this operation to avoid data loss.

## Container Migration

- You can select **Migrate Containers** when you remove or reset a node.
- If you select **Migrate Containers**, all containers on the node are migrated to other machines in the cluster.
- Container migration causes loss of the data in local volumes. Back up the data before container migration. The node reset process stops if container migration fails. In this case, you must manually reset the node again. During the second reset, you can deselect **Migrate Containers** to forcibly reset the node.
- You can click **Clusters** > **Manage** > **Events** to view the container migration log.

# Security Group

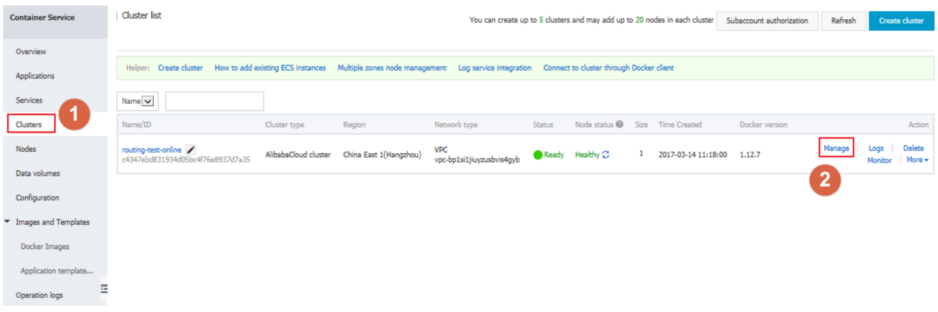
## View security group rules

### Procedures

Log on to the Container Service console.

Click **Clusters** in the left navigation pane.

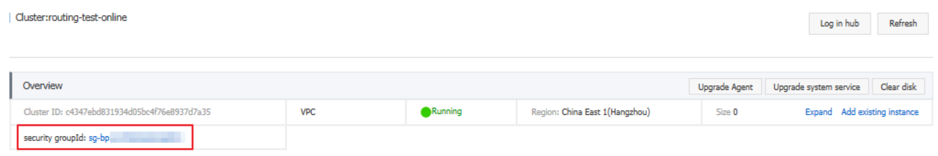
Locate the desired cluster and click **Manage** on the right side.



You can create up to 5 clusters and may add up to 20 nodes in each cluster

| Name/ID | Cluster type | Region | Network type | Status | Node status | Size | Time Created | Docker version | Action |
|---|---|---|---|---|---|---|---|---|---|
| routing-test-online c4347ebd831934d05bc4f76e8937d7a35 | AlibabaCloud cluster | China East 1(Hangzhou) | VPC vpc-bp1si1jiuyzusbvis4gyb | ● Ready | Healthy | 1 | 2017-03-14 11:18:00 | 1.12.7 | Manage \| Logs \| Delete Monitor \| More |

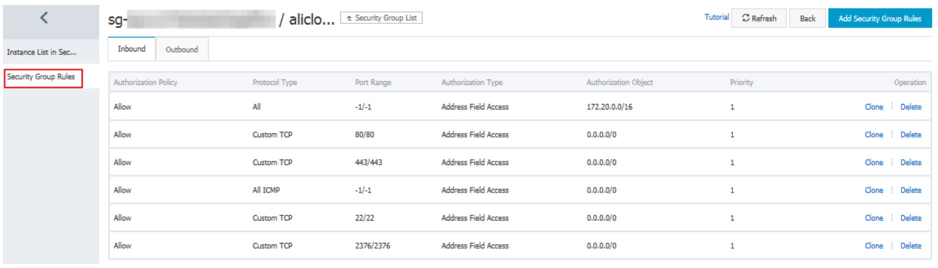
Click the security group ID to jump to the details page of this security group on the ECS Management Console.



Cluster ID: c4347ebd831934d05bc4f76e8937d7a35 | VPC | ● Running | Region: China East 1(Hangzhou) | Size 0

security groupId: sg-bp[redacted]

Click **Security Group Rules** on the left navigation pane. You can view the security group rules.

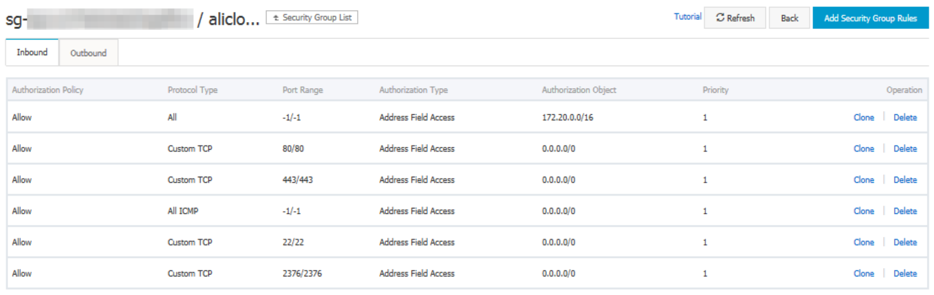


| Authorization Policy | Protocol Type | Port Range | Authorization Type | Authorization Object | Priority | Operation |
|---|---|---|---|---|---|---|
| Allow | All | -1/-1 | Address Field Access | 172.20.0.0/16 | 1 | Clone \| Delete |
| Allow | Custom TCP | 80/80 | Address Field Access | 0.0.0.0/0 | 1 | Clone \| Delete |
| Allow | Custom TCP | 443/443 | Address Field Access | 0.0.0.0/0 | 1 | Clone \| Delete |
| Allow | All ICMP | -1/-1 | Address Field Access | 0.0.0.0/0 | 1 | Clone \| Delete |
| Allow | Custom TCP | 22/22 | Address Field Access | 0.0.0.0/0 | 1 | Clone \| Delete |
| Allow | Custom TCP | 2376/2376 | Address Field Access | 0.0.0.0/0 | 1 | Clone \| Delete |

# Security group rules

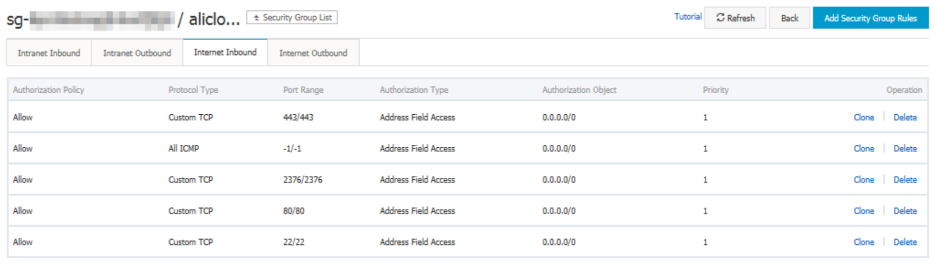
The security groups created by default for Container Service clusters created after February 28 have been reinforced. The opening rules are as follows.

### VPC security group:

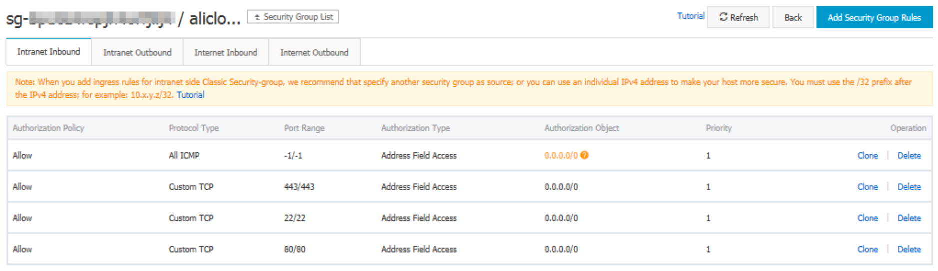

sg-[redacted] / aliclo... Security Group List

| Authorization Policy | Protocol Type | Port Range | Authorization Type | Authorization Object | Priority | Operation |
|---|---|---|---|---|---|---|
| Allow | All | -1/-1 | Address Field Access | 172.20.0.0/16 | 1 | Clone \| Delete |
| Allow | Custom TCP | 80/80 | Address Field Access | 0.0.0.0/0 | 1 | Clone \| Delete |
| Allow | Custom TCP | 443/443 | Address Field Access | 0.0.0.0/0 | 1 | Clone \| Delete |
| Allow | All ICMP | -1/-1 | Address Field Access | 0.0.0.0/0 | 1 | Clone \| Delete |
| Allow | Custom TCP | 22/22 | Address Field Access | 0.0.0.0/0 | 1 | Clone \| Delete |
| Allow | Custom TCP | 2376/2376 | Address Field Access | 0.0.0.0/0 | 1 | Clone \| Delete |

### Classic network security group (internet inbound and intranet inbound):


sg-[redacted] / aliclo... Security Group List

| Authorization Policy | Protocol Type | Port Range | Authorization Type | Authorization Object | Priority | Operation |
|---|---|---|---|---|---|---|
| Allow | Custom TCP | 443/443 | Address Field Access | 0.0.0.0/0 | 1 | Clone \| Delete |
| Allow | All ICMP | -1/-1 | Address Field Access | 0.0.0.0/0 | 1 | Clone \| Delete |
| Allow | Custom TCP | 2376/2376 | Address Field Access | 0.0.0.0/0 | 1 | Clone \| Delete |
| Allow | Custom TCP | 80/80 | Address Field Access | 0.0.0.0/0 | 1 | Clone \| Delete |
| Allow | Custom TCP | 22/22 | Address Field Access | 0.0.0.0/0 | 1 | Clone \| Delete |


sg-[redacted] / aliclo... Security Group List

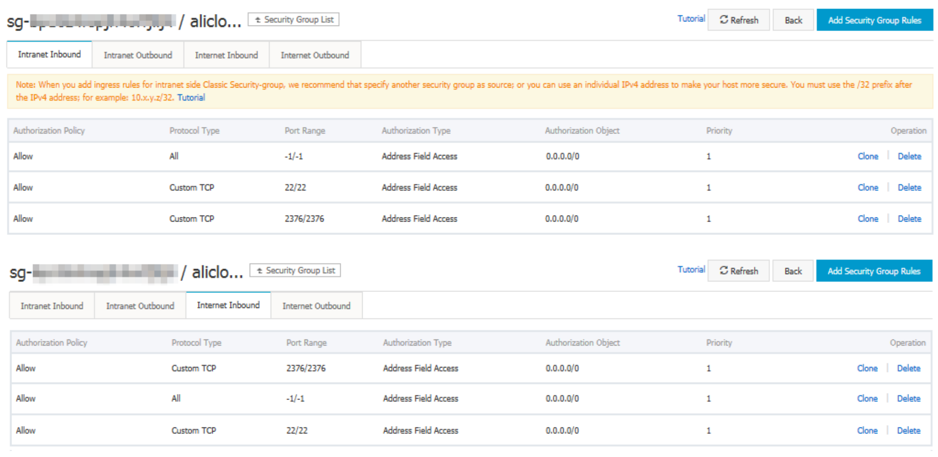
Note: When you add ingress rules for intranet side Classic Security-group, we recommend that specify another security group as source; or you can use an individual IPv4 address to make your host more secure. You must use the /32 prefix after the IPv4 address; for example: 10.x.y.z/32. Tutorial

| Authorization Policy | Protocol Type | Port Range | Authorization Type | Authorization Object | Priority | Operation |
|---|---|---|---|---|---|---|
| Allow | All ICMP | -1/-1 | Address Field Access | 0.0.0.0/0 | 1 | Clone \| Delete |
| Allow | Custom TCP | 443/443 | Address Field Access | 0.0.0.0/0 | 1 | Clone \| Delete |
| Allow | Custom TCP | 22/22 | Address Field Access | 0.0.0.0/0 | 1 | Clone \| Delete |
| Allow | Custom TCP | 80/80 | Address Field Access | 0.0.0.0/0 | 1 | Clone \| Delete |

**Note:**

- Port 443 and Port 80 can be opened or closed based on your own needs.
- ICMP rules are recommended to be retained for the convenience of troubleshooting. Some tools are also dependent on ICMP.
- The Container Service depends on Port 22 and Port 2376 to initialize the machine. Make sure to keep the two rules.

The security group rules for clusters created before February 28 are loose. Take the classic network security group rules for example.



Note: When you add ingress rules for intranet side Classic Security-group, we recommend that specify another security group as source; or you can use an individual IPv4 address to make your host more secure. You must use the /32 prefix after the IPv4 address; for example: 10.x.y.z/32. Tutorial

| Authorization Policy | Protocol Type | Port Range | Authorization Type | Authorization Object | Priority | Operation |
|---|---|---|---|---|---|---|
| Allow | All | -1/-1 | Address Field Access | 0.0.0.0/0 | 1 | Clone \| Delete |
| Allow | Custom TCP | 22/22 | Address Field Access | 0.0.0.0/0 | 1 | Clone \| Delete |
| Allow | Custom TCP | 2376/2376 | Address Field Access | 0.0.0.0/0 | 1 | Clone \| Delete |

| Authorization Policy | Protocol Type | Port Range | Authorization Type | Authorization Object | Priority | Operation |
|---|---|---|---|---|---|---|
| Allow | Custom TCP | 2376/2376 | Address Field Access | 0.0.0.0/0 | 1 | Clone \| Delete |
| Allow | All | -1/-1 | Address Field Access | 0.0.0.0/0 | 1 | Clone \| Delete |
| Allow | Custom TCP | 22/22 | Address Field Access | 0.0.0.0/0 | 1 | Clone \| Delete |

If you want to tighten the rules, refer to the configurations of the security groups created after February 28 and make the following changes (use **Add Security Group Rules** and **Delete** in the figure above).

Add **Allow** + **ICMP** rule in intranet inbound and internet inbound.

If you want to access Port 80 and Port 443 or other ports of the VM, add intranet and internet rules to open these ports.

**Note:** Make sure you open all the ports you need. Otherwise some services may become unavailable. Do not open ports accessed via Server Load Balancers.

Delete the internet inbound rules and intranet inbound rules with the port range of -1/-1 in the address range of 0.0.0.0.

## Security configuration principles

Each cluster should have one security group.

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

Minimal permission principle.

The security group should open the minimal permission required for external access.

For classic network security groups, the internet and intranet rules need to be configured separately.

According to the minimal permission principle, only add rules for the desired NIC type. By default, ECS instances within a security group can communicate with one another. As a result, if you want to add intranet inbound rules, it is recommended that you confirm the reason for the addition, whether access from ECS instances outside the security group is required.

The Container Service security group adds some default rules.

For easier operations on ECS instances, the Container Service security group adds some default rules, for example, ports 80/443 are opened. You can delete the rules if you don’t need them.

**Note:** Do not block ports 22 and 2376. The Container Service needs the two ports to initialize the ECS instances.

Try to communicate over the intranet within the container and do not expose communications to the host machine.

To authorize ECS instances outside the security group for access to the security group, authorize a security group, instead of an individual IP address.

If you want to authorize ECS instances outside the security group to access the security group, you should first 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.

Prioritize the use of the VPC network. Do not bind an EIP address to a node unless necessary. The VPC network has a better isolation performance.

The container network segment should be opened at the VPC intranet outbound/inbound.

Otherwise, the network between containers might be disconnected.

# Image and template management

# View the image list

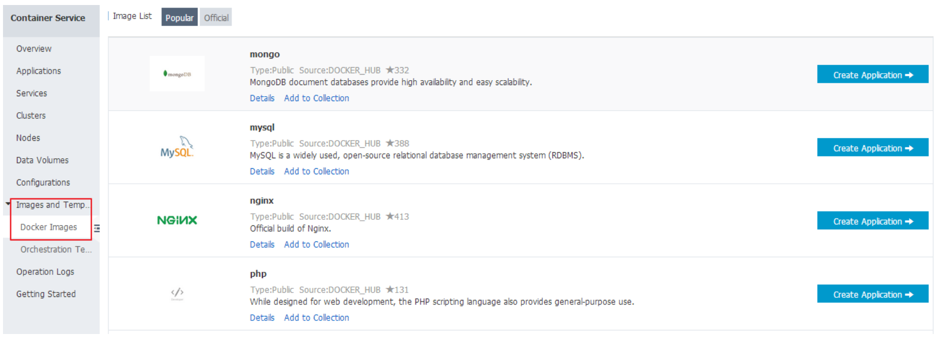
## Operating procedure

1. Log on to the Container Service console.

   Click **Images and Templates** in the left navigation pane and click **Docker Images**.

   You can view the image category.

   - **Popular:** Some common images recommended by the Container Service.
   - **Official:** Official images provided by the Docker hub.



mongo
Type:Public Source:DOCKER_HUB ★332
MongoDB document databases provide high availability and easy scalability.

mysql
Type:Public Source:DOCKER_HUB ★388
MySQL is a widely used, open-source relational database management system (RDBMS).

nginx
Type:Public Source:DOCKER_HUB ★413
Official build of Nginx.

php
Type:Public Source:DOCKER_HUB ★131
While designed for web development, the PHP scripting language also provides general-purpose use.

# View the orchestration template list

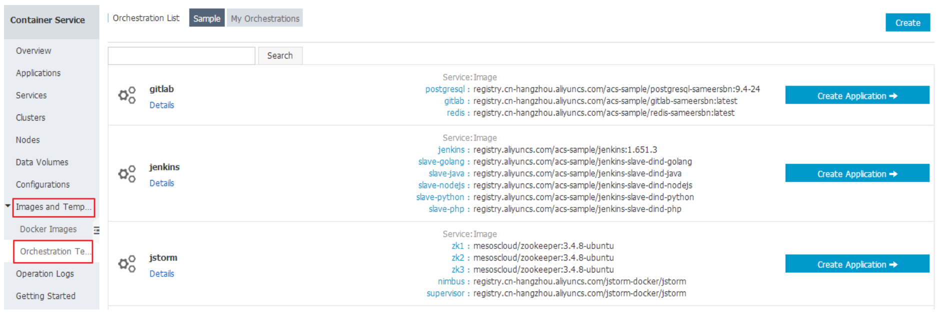
## Operating procedure

Log on to the Container Service console.

Click **Images and Templates** in the left navigation pane and click **Orchestration Templates**.

You can view the template category or click **Search** to search the templates by their name prefix.

- **Sample:** Common orchestration templates recommended by the Container Service.
- **My Templates:** The orchestration templates you create.



gitlab

```
  Service: Image
postgresql : registry.cn-hangzhou.aliyuncs.com/acs-sample/postgresql-sameersbn:9.4-24
    gitlab : registry.cn-hangzhou.aliyuncs.com/acs-sample/gitlab-sameersbn:latest
     redis : registry.cn-hangzhou.aliyuncs.com/acs-sample/redis-sameersbn:latest
```

jenkins

```
  Service: Image
     jenkins : registry.aliyuncs.com/acs-sample/jenkins:1.651.3
 slave-golang : registry.aliyuncs.com/acs-sample/jenkins-slave-dind-golang
   slave-java : registry.aliyuncs.com/acs-sample/jenkins-slave-dind-java
 slave-nodejs : registry.aliyuncs.com/acs-sample/jenkins-slave-dind-nodejs
 slave-python : registry.aliyuncs.com/acs-sample/jenkins-slave-dind-python
    slave-php : registry.aliyuncs.com/acs-sample/jenkins-slave-dind-php
```

jstorm

```
  Service: Image
       zk1 : mesoscloud/zookeeper:3.4.8-ubuntu
       zk2 : mesoscloud/zookeeper:3.4.8-ubuntu
       zk3 : mesoscloud/zookeeper:3.4.8-ubuntu
    nimbus : registry.cn-hangzhou.aliyuncs.com/jstorm-docker/jstorm
supervisor : registry.cn-hangzhou.aliyuncs.com/jstorm-docker/jstorm
```

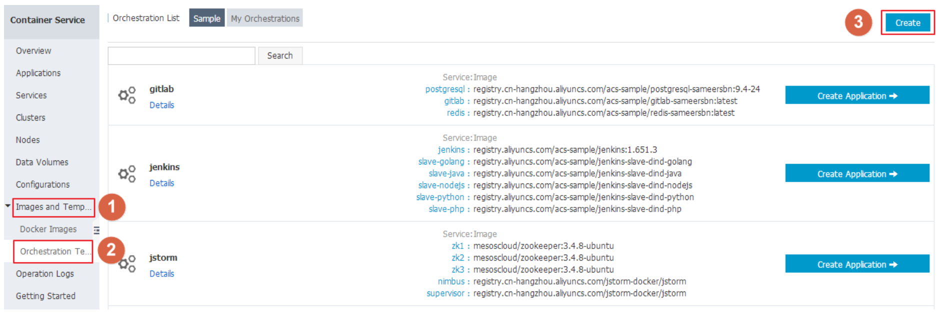
# Create an orchestration template

## Operating procedure

Log on to the Container Service console.

Click **Images and Templates** in the left navigation pane and click **Orchestration Templates**.

Click **Create**.



gitlab

```
  Service: Image
postgresql : registry.cn-hangzhou.aliyuncs.com/acs-sample/postgresql-sameersbn:9.4-24
    gitlab : registry.cn-hangzhou.aliyuncs.com/acs-sample/gitlab-sameersbn:latest
     redis : registry.cn-hangzhou.aliyuncs.com/acs-sample/redis-sameersbn:latest
```

jenkins

```
  Service: Image
     jenkins : registry.aliyuncs.com/acs-sample/jenkins:1.651.3
 slave-golang : registry.aliyuncs.com/acs-sample/jenkins-slave-dind-golang
   slave-java : registry.aliyuncs.com/acs-sample/jenkins-slave-dind-java
 slave-nodejs : registry.aliyuncs.com/acs-sample/jenkins-slave-dind-nodejs
 slave-python : registry.aliyuncs.com/acs-sample/jenkins-slave-dind-python
    slave-php : registry.aliyuncs.com/acs-sample/jenkins-slave-dind-php
```

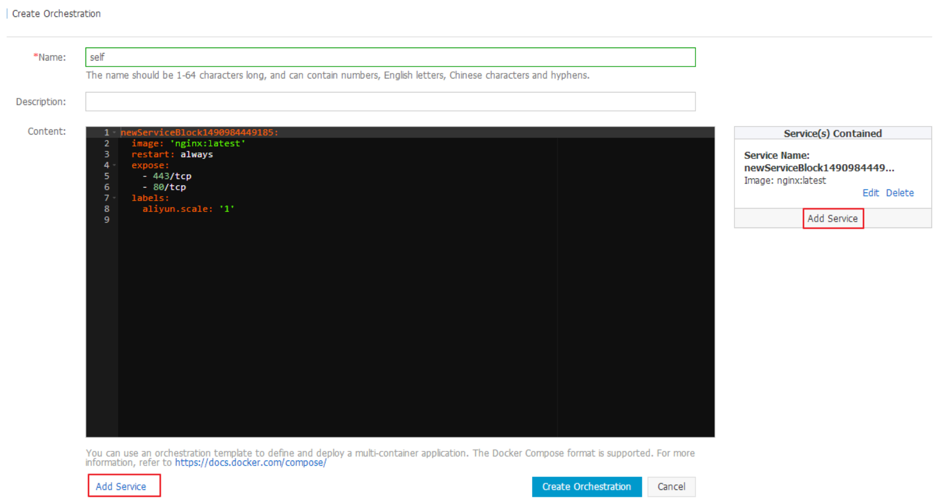
jstorm

```
  Service: Image
       zk1 : mesoscloud/zookeeper:3.4.8-ubuntu
       zk2 : mesoscloud/zookeeper:3.4.8-ubuntu
       zk3 : mesoscloud/zookeeper:3.4.8-ubuntu
    nimbus : registry.cn-hangzhou.aliyuncs.com/jstorm-docker/jstorm
supervisor : registry.cn-hangzhou.aliyuncs.com/jstorm-docker/jstorm
```

In the **Create Template** page, enter the template information.

- **Name:** The template name.
- **Description:** Information about the template.
- **Content:** The yml file of Docker Compose. For more details, refer to Compose File details.

The services contained in the orchestration template are displayed on the right side. You can click **Edit** to modify the template in **Services Contained** or click **Delete** to delete the selected service.



Create Orchestration

*Name: self

The name should be 1-64 characters long, and can contain numbers, English letters, Chinese characters and hyphens.

Description:

Content:

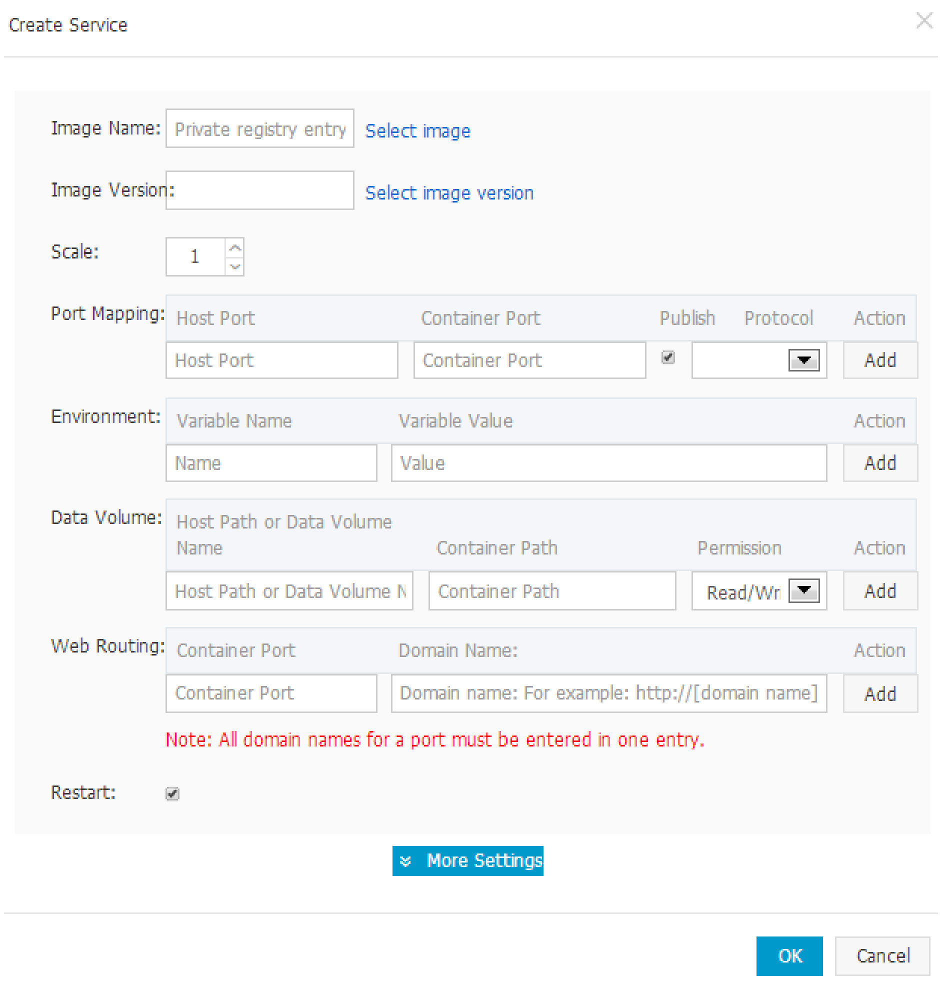
```
newServiceBlock1490984449185:
  image: 'nginx:latest'
  restart: always
  expose:
    - 443/tcp
    - 80/tcp
  labels:
    aliyun.scale: '1'

```

Service(s) Contained

Service Name: newServiceBlock1490984449...
Image: nginx:latest

You can use an orchestration template to define and deploy a multi-container application. The Docker Compose format is supported. For more information, refer to https://docs.docker.com/compose/

In addition, you can click **Add Service** and select the desired image to add services to the orchestration template.



Create Service

Image Name: Private registry entry

Image Version:

Scale: 1

Port Mapping:

| Host Port | Container Port | Publish | Protocol | Action |
|---|---|---|---|---|
| Host Port | Container Port | ✓ | | Add |

Environment:

| Variable Name | Variable Value | Action |
|---|---|---|
| Name | Value | Add |

Data Volume:

| Host Path or Data Volume Name | Container Path | Permission | Action |
|---|---|---|---|
| Host Path or Data Volume N | Container Path | Read/Wr | Add |

Web Routing:

| Container Port | Domain Name: | Action |
|---|---|---|
| Container Port | Domain name: For example: http://[domain name] | Add |

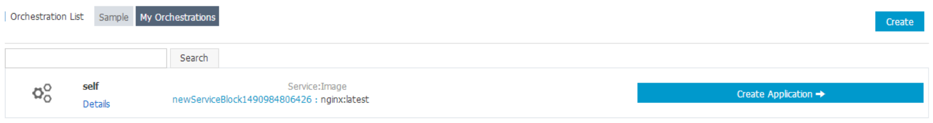
Note: All domain names for a port must be entered in one entry.

Restart: ✓

Click **Create Orchestration**.

## Subsequent operations

You can view the orchestration template created under **My Orchestration** in the **Template List** page.



You can click **Details** to view the detailed information of the orchestration template or you can click **Create Application** to use the orchestration template to create an application.

## Update an orchestration template

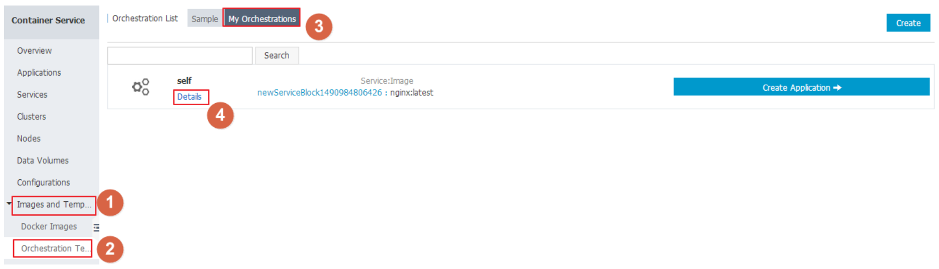
You can only edit orchestration templates that are displayed under **My Orchestrations** in the **Template List** page. If you want to edit **Sample** templates, you can first save the sample template as your own templates and then edit them.

### Operating procedure

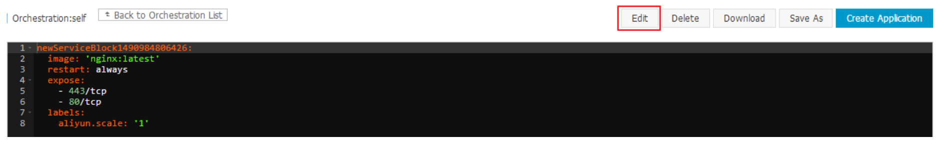
Log on to the Container Service console.

Click **Images and Templates** in the left navigation pane and click **Orchestration Templates**.

Click **My Orchestrations**, locate the template to be updated and then click **Details**.



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

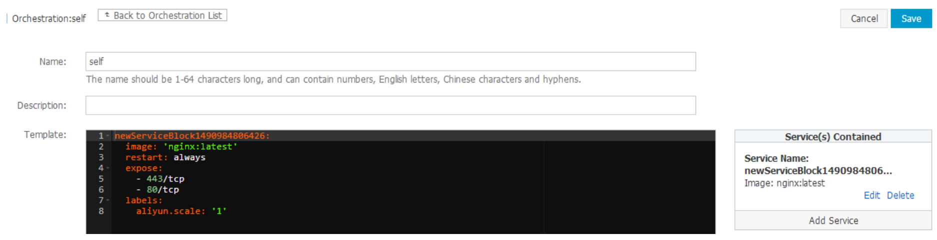


```
newServiceBlock1490984806426:
  image: 'nginx:latest'
  restart: always
  expose:
    - 443/tcp
    - 80/tcp
  labels:
    aliyun.scale: '1'
```

Edit the template content.

You can directly make modifications in the template. Or you can select a service at the right side and click **Edit** to modify the service or click **Delete** to delete the service.

In addition, you can click **Add Service** and select the desired image in the pop-up dialog box to add the service to the orchestration template.



Name: self

The name should be 1-64 characters long, and can contain numbers, English letters, Chinese characters and hyphens.

Description:

Template:

```
newServiceBlock1490984806426:
  image: 'nginx:latest'
  restart: always
  expose:
    - 443/tcp
    - 80/tcp
  labels:
    aliyun.scale: '1'
```

Service(s) Contained

Service Name: newServiceBlock1490984806...
Image: nginx:latest

Click **Save** in the upper-right corner to save the template content.

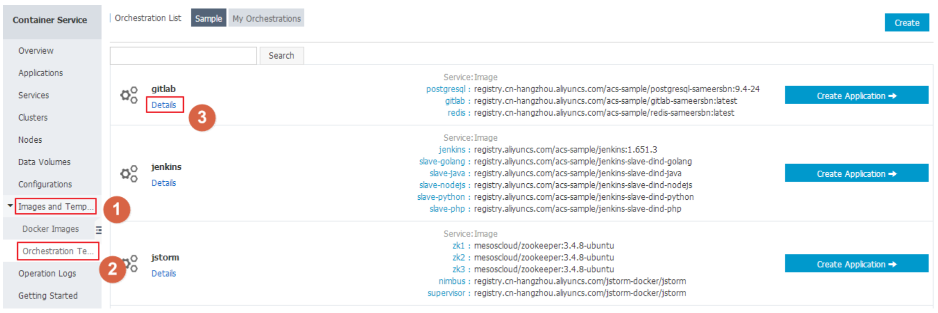
# Download an orchestration template

## Operating procedure

1. Log on to the Container Service console.

Click **Images and Templates** in the left navigation pane and click **Orchestration Templates**.

Select a template and click **Details**.



gitlab

Service: Image
postgresql : registry.cn-hangzhou.aliyuncs.com/acs-sample/postgresql-sameersbn:9.4-24
gitlab : registry.cn-hangzhou.aliyuncs.com/acs-sample/gitlab-sameersbn:latest
redis : registry.cn-hangzhou.aliyuncs.com/acs-sample/redis-sameersbn:latest

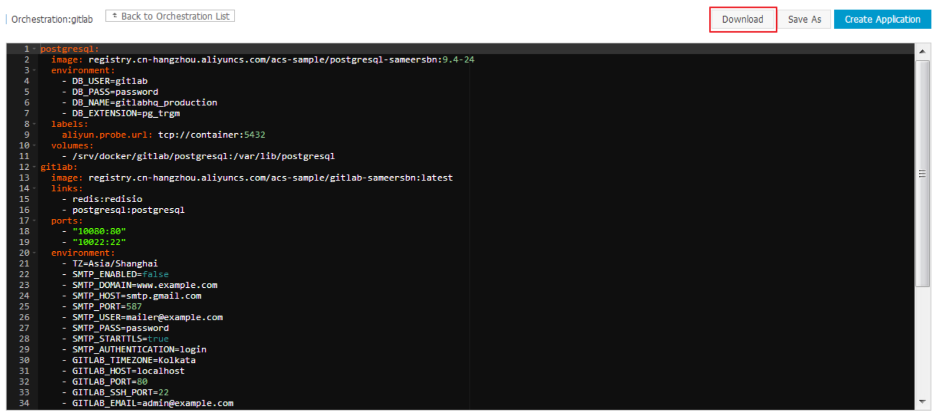
jenkins

Service: Image
jenkins : registry.aliyuncs.com/acs-sample/jenkins:1.651.3
slave-golang : registry.aliyuncs.com/acs-sample/jenkins-slave-dind-golang
slave-java : registry.aliyuncs.com/acs-sample/jenkins-slave-dind-java
slave-nodejs : registry.aliyuncs.com/acs-sample/jenkins-slave-dind-nodejs
slave-python : registry.aliyuncs.com/acs-sample/jenkins-slave-dind-python
slave-php : registry.aliyuncs.com/acs-sample/jenkins-slave-dind-php

jstorm

Service: Image
zk1 : mesoscloud/zookeeper:3.4.8-ubuntu
zk2 : mesoscloud/zookeeper:3.4.8-ubuntu
zk3 : mesoscloud/zookeeper:3.4.8-ubuntu
nimbus : registry.cn-hangzhou.aliyuncs.com/jstorm-docker/jstorm
supervisor : registry.cn-hangzhou.aliyuncs.com/jstorm-docker/jstorm

Click **Download** in the upper-right corner to immediately download a template file with the suffix of yml.



```
postgresql:
  image: registry.cn-hangzhou.aliyuncs.com/acs-sample/postgresql-sameersbn:9.4-24
  environment:
    - DB_USER=gitlab
    - DB_PASS=password
    - DB_NAME=gitlabhq_production
    - DB_EXTENSION=pg_trgm
  labels:
    aliyun.probe.url: tcp://container:5432
  volumes:
    - /srv/docker/gitlab/postgresql:/var/lib/postgresql
gitlab:
  image: registry.cn-hangzhou.aliyuncs.com/acs-sample/gitlab-sameersbn:latest
  links:
    - redis:redisio
    - postgresql:postgresql
  ports:
    - "10080:80"
    - "10022:22"
  environment:
    - TZ=Asia/Shanghai
    - SMTP_ENABLED=false
    - SMTP_DOMAIN=www.example.com
    - SMTP_HOST=smtp.gmail.com
    - SMTP_PORT=587
    - SMTP_USER=mailer@example.com
    - SMTP_PASS=password
    - SMTP_STARTTLS=true
    - SMTP_AUTHENTICATION=login
    - GITLAB_TIMEZONE=Kolkata
    - GITLAB_HOST=localhost
    - GITLAB_PORT=80
    - GITLAB_SSH_PORT=22
    - GITLAB_EMAIL=admin@example.com
```

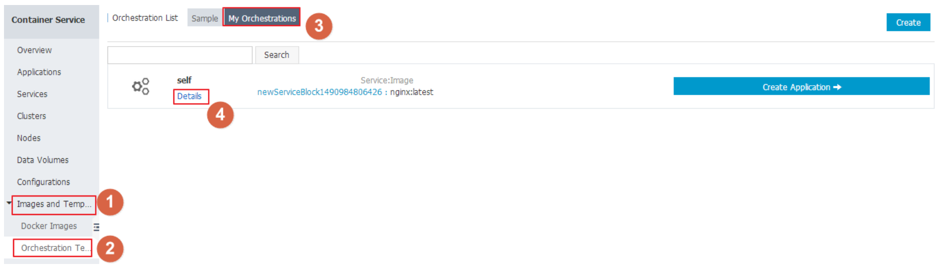
# Delete an orchestration template

## Operating procedure

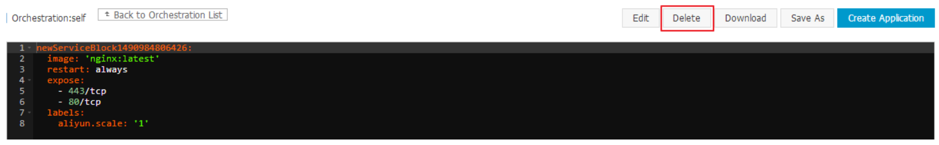
Log on to the Container Service console.

Click **Images and Templates** in the left navigation pane and click **Orchestration Templates**.

Click **My Orchestrations**, locate the template to be deleted and click **Details**.



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



```
newServiceBlock1490984806426:
  image: 'nginx:latest'
  restart: always
  expose:
    - 443/tcp
    - 80/tcp
  labels:
    aliyun.scale: '1'
```

Click **OK** in the pop-up dialog box.

# Service orchestration

## Overview

The Container Service supports the Docker Compose orchestration template to describe multi-container applications.

The orchestration template allows the description an integrated application that can be composed of several services. For example, a portal application can comprise of a Nginx service, a Web service and a database service.

One service may have several container instances and the configurations of all the container instances have to be consistent. For example, the Web service in the above application can activate two or more containers based on the traffic.

## Capacity

The Container Service supports automatic deployment and management of a container application using the orchestration template file.

The labels used by the orchestration template file are compatible with most of the labels described in Docker Compose version 1.5x to 1.7x. For information about specific compatible labels, refer to Label description.

The orchestration template file also supports the version 1 and version 2 Compose file formats. For details, refer to Compose file format versioning.

The Container Service also provides many scale capabilities beyond the community version:

- Unlike the community versions of Docker Compose and Swarm, Alibaba Cloud Container Service supports cross-node container link, and thus you can directly deploy the application described by Docker Compose to the distributed cluster to provide high availability and scalability.
- The Container Service, based on the description in the community Compose template, also provides capabilities to simplify the deployment, maintenance and operations of Web and microservice applications. For details, refer to 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
```

## Label description

The labels used by the Container Service orchestration templates are compatible with most of Docker Compose 1.5.x to 1.7.x labels and provides lots of extended capabilities on the basis of the community version.

### Scale capability labels

The Container Service extends the deployment and lifecycle management capabilities for orchestration templates, and all the scale capabilities are described under labels and deployed for use as sub-labels.

| Label | Description |
|---|---|
| probe | Sets service health check. |
| rolling_updates | Sets the rolling update of services. |
| parallelism | Defines the number of containers to be |

| | |
|---|---|
| | updated at a time.<br>**Note:** This label must be used with the rolling_updates label. |
| depends | Sets service dependencies. |
| scale | Sets the number of containers for this service to scale horizontally. |
| routing | Sets the access domain name of this service. |
| routing.session_sticky | Sets whether routing maintains session sticky (that is session persistence) during the routing request. |
| lb | Exposes the service port to the public network or intranet by customizing Alibaba Cloud Server Load Balancer nat mapping. |
| global | Sets this service as a global service. |

## Function enhancement labels

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

## Other supported labels

| Label | Description |
|---|---|
| external | Sets this service to forcibly link an external address. |
| dns_options | Sets DNS options. The function of this label is the same with that of --dns-opt in the docker run command. |
| oom_kill_disable | Determines whether OOM Killer will be prohibited. The function of this label is the same with that of --oom-kill-disable in the docker run command. |

## Substitute the variable

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

For details, refer to Substitute the variable.

## Container rescheduling

The Container Service supports the rescheduling of Docker container: when a node is invalid, the container can be automatically rescheduled to other available node for operation.

For details, refer to Container rescheduling.

## High availability scheduling

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

For details, refer to High availability scheduling.

## Docker Compose labels that are not supported

Currently, some Docker Compose labels are not supported by the Container Service. For these labels, refer to Docker Compose labels that are not supported.

# probe

Set service health check.

- URLs can be used to perform the check. HTTP and TCP protocols are supported.
- Shell scripts can be used to perform the check.

The health check is initiated from the container host at regular intervals (two seconds by default) to the container or the shell script command is executed 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.

Description of the fields used for check:

aliyun.probe.url: URL requested by HTTP and TCP. You do not have to fill in your own domain name or IP address. You only need to add the word container, which will be resolved into a corresponding container IP address for health check. The service passes the health check when 2XX or 3XX is returned.

- For example, if the container uses Port 8080 to provide the HTTP service and /ping as the URL for health check, the URL format for the probe is http://container:8080/ping. The Container Service automatically uses HTTP GET to request to check the URL for the return results, and if 2XX or 3XX is returned, the health check is successful.
- For example, MySQL container monitors Port 3306, and the URL format for the probe is tcp://container:3306. The service will check whether Port 3306 is enabled. If yes, the health check is successful.

aliyun.probe.cmd: The Shell command, /check.sh, is executed during the health check; the Container Service regularly executes this command within the container. If the shell scripts return the value 0, the health check is successful.

aliyun.probe.timeout_seconds: 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:

- One service can only contain either aliyun.probe.url or aliyun.probe.cmd.
- If aliyun.probe.url or aliyun.probe.cmd is not included in the service, by default, the status of the container is healthy, and other aliyun.probe.xxx labels will be ignored.

**Example:**

Use URL to check the health status of the container.

```
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 the health status of the container.

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

## rolling_updates

During a service update, if this service includes more than one container (by using the scale label), you can define the Container Service to update the (n+1)th container after the nth container has been successfully updated, so as to minimize the service downtime.

**Example:**

Deploy the WordPress service, use the scale label to specify two containers to be deployed, and 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 the number of containers to be updated at a time.

**Note:** This label must be used with the rolling_updates label.

**Label values:**

- The default value is 1, namely updating one container at a time.
- When this value is greater than 1, during rolling_updates, the Container Service updates a certain number of containers at a time as defined by the parallelism label, thus realizing batch update.
- When this value is invalid, the default value will be used.

**Note:** To ensure that there is always at least one container that is providing service, it is recommended that you define the parallelism label to a value less than the number of containers in the service.

**Sample:**

The following example deploys a Nginx service which contains 3 containers as defined by the scale label and the Container Service will update 2 containers at a time as defined by the rolling_updates label and the parallelism label.

```
web:
image: nginx:latest
restart: always
environment:
- "reschedule:on-node-failure"
ports:
- 80
labels:
aliyun.scale: "3"
aliyun.rolling_updates: 'true'
aliyun.rolling_updates.parallelism: "2"
```

## depends

Set service dependencies.

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

**Example:**

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

## scale

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

Currently, the Docker Compose can only start one container in each service. To expand the number of containers, you need to manually set the number after the container starts.

You can now use the scale label to scale as the container starts.

Moreover, after a container is deleted, you can redeploy the application in the Container Service console.(click **Applications** in the left navigation pane > locate the target application > click **Redeploy** ) to restart the container or create a new container to restore the number of containers to the specified quantity.

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

## routing

Set the access domain name of this service.

**Format:**

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

**Explanation:**

- $container_port: The container port. Note that this is not the host port.
- $domain: The domain name. You need to enter your own domain name.
- $domain_prefix: The domain name prefix. If you enter the domain name prefix, the Container Service will provide you a domain name for testing and the domain name suffix is wordpress.<cluster_id>.<region_id>.alicontainer.com.
- $context_path: The request path. You can select and distinguish different backend containers according to the request path.

**Domain name selection:**

- If HTTP protocol is used to expose the service, you can use the Container Service to provide the internal domain name (the top-level domain is alicontainer.com) for test or you can use your own domain name.
- If HTTPS protocol is used, it only supports the domain name you provide, such as www.example.com. The DNS settings need to be modified to specify the domain name to be connected to the Server Load Balancer service provided by the container cluster.

**Format requirements of the label statement:**

- The Container Service allocates subdomain names for each cluster, and you only need to provide the domain name prefix to bind the internal domain name. The domain name prefix only indicates the domain level and cannot be divided with periods (.).
- If you don’t specify scheme, the HTTP protocol will be used by default.
- Both the length of the domain name and the context root should not exceed 128 characters.
- When you bind several domain names to the service, use semicolons (;) to separate them.
- A backend service can have several ports. Such a port refers to the port that the container exposes. One port can only use one label for statement and a service with several ports need to have several labels.

**Example:**

Bind the internal domain name wordpress.<cluster_id>.<region_id>.alicontainer.com provided by the Container Service to Port 80 of the Web service. And bind 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
```

You will then obtain an internal domain name wordpress.cd3dfe269056e4543acbec5e19b01c074.cn-beijing.alicontainer.com. After starting the Web service, you can access corresponding Web services with the URL http://wordpress.cd3dfe269056e4543acbec5e19b01c074.cn-beijing.alicontainer.com or http://wp.sample.com/context.

If the HTTPS service needs to be supported, upload the HTTPS certificate using the Server Load Balancer management console on the Alibaba Cloud website. Then, bind the corresponding cluster to access the Server Load Balancer terminal.

## routing.session_sticky

Set whether routing maintains session sticky (that is session persistence) during the routing request. With session persistence, during the session, the request is routed to the same backend container instead of being randomly routed to different containers for each request.

**Note:** The setting can only work after aliyun.routing.port_$contaienr_port has been set.

The setting method is as follows:

- Enable session persistence

aliyun.routing.session_sticky: true

- Disable session persistence

aliyun.routing.session_sticky: false

**Example:**

```
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
```

## lb

Expose the service port to the public network or intranet by customizing Alibaba Cloud Server Load Balancer NAT mapping. Agent must be upgraded to the latest version to support this scaling capacity label.

The label format is as follows, and variables with $ are placeholders.

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

**Example:**

```
web:
image: wordpress:4.2
```

```
ports:
- 80:80
- 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_8080: https://14a7ba06d3b-cn-hangzhou-dg-a01:80
aliyun.lb.port_53: udp://14a7ba06d3b-cn-hangzhou-dg-a01:53
db:
image: mysql
environment:
- MYSQL_ROOT_PASSWORD=password
```

To better utilize the custom load balancing lb label, you need to understand three ports used in a routing request: the Server Load Balancer front port, Server Load Balancer backend port (that is the ECS vm port) and container port.

Taking the first lb label aliyun.lb.port_80 of the above 4 lb labels as an example, from left to right, Port 80 in the key refers to the port exposed by the container, and Port 8080 refers to the front port exposed by the Server Load Balancer. The Server Load Balancer backend port is the ECS vm port, which can be obtained from host:container port mapping of the ports label. From this, you can find that Port 80 of the container corresponds to Port 80 of the host, and thus you can confirm that the backend port forwarded by the Server Load Balancer is Port 80. Therefore, the first label indicates that a request sent to the Web service first enters Port 8080 in front of the Server Load Balancer, then it is forwarded to Port 80 of the backend host vm, and finally according to the port mapping of ports, the request enters Port 80 of the container and is submitted to the WordPress process in the container for service provision.

**Note:** All server load balancings configured by this label do not go through the routing service built in the cluster, and the routing of the request is controlled by yourself.

**Format requirements of the label statement:**

- The name or ID of the Server Load Balancer instance can be used to specify the Server Load Balancer instance.
- The name of the Server Load Balancer instance is limited to 1~80 characters, including letters, numbers, hyphens (-), forward slashes (/), periods (.) and underscores (_).
- The container port values are limited to 1~65535.
- The Server Load Balancer front port values are limited to 1~65535.

**Constraints on deploying services with custom Server Load Balancer NAT mapping:**

You need to build a Server Load Balancer instance, name it and build the corresponding monitoring port. Then you need to provide the name of the Server Load Balancer instance like $slb_name or $slb_id, the port that will be exposed, the $scheme protocol (possible values include tcp, http, https, and udp) to be used and the mapping container port using the scale labels, and specify the front port $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.

> **Note:** You must specify the host port and what’ s more, this port cannot conflict with the host port mapped by other services. The Server Load Balancer will use the host port to bind the backend ECS vm machine.

One service can only use one or several Server Load Balancer instances to expose the service port. Because several services will be distributed to different vm backends, several services cannot share and use the same Server Load Balancer instance.

The host deployed with the service configured with the Server Load Balancer nat mapping uses the same host:container port mapping, and thus, these services only have one instance on each vm host.

The Server Load Balancer protocols $scheme supported include tcp, http, https, and udp.

You need to build the monitoring port on the official Server Load Balancer management console in Alibaba Cloud.

You can log on to the official Server Load Balancer management console to modify the specific configuration for the Server Load Balancer instance in the Container Service, such as bandwidth limitation.

The advantage for the lb label is that you don’ t need to bind the ECS vm server of the Server Load Balancer by yourself. After you configure the corresponding labels, the backend servers will be bound automatically. Therefore, except for binding the backend server of the Server Load Balancer, you need to set and modify the Server Load Balancer instances on the Alibaba Cloud Server Load Balancer management console.

The Container Service will help you generate a RAM sub-account (you need to activate RAM), and this account comes with some Server Load Balancer permissions (without permission to create and delete Server Load Balancer instances) to help you manage the Server Load Balancer instances used in the Container Service, for example, binding some nodes in the cluster as the service backend.

## global

Set this service as a global service.

Some services need to be deployed in every node, such as monitoring and logging services. Such service will be deployed whenever a new node is built.

When a service is set as global, this service will be deployed in every node of the cluster. When a new node is created or added in the cluster, a new container instance will be automatically deployed to the new node.

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

## Service deployment constraint (affinity:service)

Set the deployment constraints for a service.

The Container Service supports the container deployment constraints compatible with Docker Swarm, and you can control the deployment of a container with the Docker Swarm Filter.

In the community version for Docker Compose, there is no relevant capability in which to control the deployment constraint for the service directly.

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

**Example:**

In this example, affinity.service!=master is the deployment constraint for the slave service. In this way, the slave service will always be deployed on nodes on which the master service is not deployed. Therefore, when a node is invalid, the service availability will not be affected. When your cluster has only one node, the deployment will fail due to the specified hard anti-affinity, because the deployment cannot meet the specified mandatory constraints.

```
master:
image: mysql:5.6
environment:
```

```
- MYSQL_USER=user
- MYSQL_PASS=test
- REPLICATION_MASTER=true
- REPLICATION_USER=repl
- REPLICATION_PASS=repl
ports:
- 3306
slave:
image: mysql:5.6
environment:
- MYSQL_USER=user
- MYSQL_PASS=test
- REPLICATION_SLAVE=true
- affinity:service!=master
ports:
- 3306
links:
- master:mysql
```

## external

Set this service to forcibly link an external address.

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

- host: Set the link domain.
- ports: Set the link port.

**Example:**

Do not use the external label, and directly start a MySQL container.

```
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 a RDS service that has not been deployed in the cluster and provide it 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
```

## dns_options

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

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

## oom_kill_disable

Determine whether OOM Killer will be prohibited. The function of this label is the same with that of --oom-kill-disable in the docker run command.

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

## Substitute the variable

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

For example, you can define the parameter POSTGRES_VERSION.

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

When the preceding Compose template is deployed, you will be prompted to enter the value of the POSTGRES_VERSION parameter, such as 9.3. The Container Service will substitute the variable of the Compose template with this parameter value. In this example, a postgres:9.3 container will be deployed.

The 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 switch the character string containing $, and in this way, the Container Service will not erroneously treat it as the parameter.

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

## Container rescheduling

The Container Service supports the rescheduling of Docker container: when a node is invalid, the container can be automatically rescheduled to other available node for operation.

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

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

**Environment variable:**

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

**Label:**

```
redis:
image: redis
labels:
- com.docker.swarm.reschedule-policies=["on-node-failure"]
```

**Note:** If, after the container has been rescheduled, the persistent state is required for recovering the Docker container, use a Docker file volume that supports data migration and sharing.

## High availability scheduling

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

You can select the zone in the orchestration file using the environment variables in one of the following two formats.

availability:az==3

The service must be distributed in at least three zones. If there are less than three zones in the current cluster, or the service cannot be distributed in three different zones due to limited machine resources, the container creation will fail.

availability:az==~3

The service should be distributed in three different zones if possible; container can still be created even if the condition cannot be fulfilled.

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
```

## Docker Compose labels not supported

| Label | Description |
|---|---|
| build | This label is used to build container images using the Dockerfile and other files in the current directory.<br>The Container Service does not provide the |

| | |
|---|---|
| | function to build images, and it is recommended that you separate the building and deployment operations.<br>You can use Alibaba Cloud 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 | The Container Service does not support setting environment variables in files. You can add environment variables using the environment label. |
| mac_address | Mac address is not supported. |
| detach | All the images in the 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. |
| networks | The network in version 2 Compose file format allows the service container to start in the custom network, and the containers in the Container Service are all in the same cross-host interconnected container network. Therefore, the Container Service does not support the use of the networks label in version 2 Compose file format. Refer to Cross-host container network for network management and service discovery for the Container Service. |

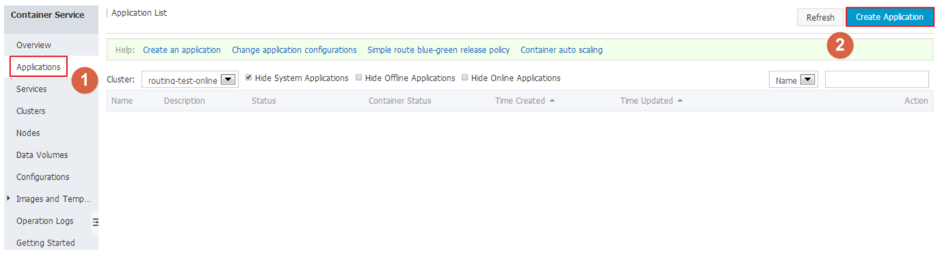
# Application management

# Create an application

## Operating procedure

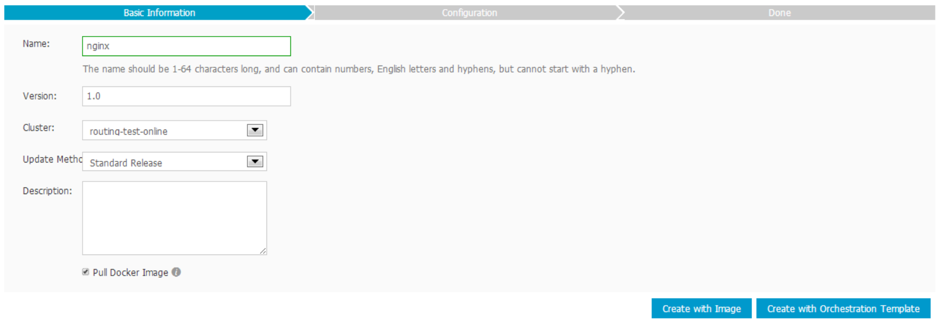
Log on to the Container Service console.

Click **Applications** in the left navigation pane and click **Create Application** in the upper-right corner.


Container Service
Overview
Applications
Services
Clusters
Nodes
Data Volumes
Configurations
Images and Temp...
Operation Logs
Getting Started
1
Application List
Refresh
Create Application
2
Help: Create an application Change application configurations Simple route blue-green release policy Container auto scaling
Cluster: routing-test-online
Hide System Applications
Hide Offline Applications
Hide Online Applications
Name
Name Description Status Container Status Time Created Time Updated Action

Set the basic application information.

- **Name:** The name of the application to be created. It must contain 1~64 characters, and can be composed of numbers, Chinese characters, English letters, and hyphen (-).
- **Version:** The version of the application to be created. By default, the version is 1.0.
- **Cluster:** The cluster which the application will be deployed to.
- **Description:** Information of the application. It can be left blank and, if entered, cannot exceed 1,024 characters. This information will be displayed in the **Application List** page.
- **Pull Docker Image:** Check to use the latest Docker image in registry.



Name: nginx

The name should be 1-64 characters long, and can contain numbers, English letters and hyphens, but cannot start with a hyphen.

Version: 1.0

Cluster: routing-test-online

Update Metho Standard Release

Description:

✓ Pull Docker Image

Click **Create with Image** or **Create with Orchestration Template**.

Click **Create with Image**.

a. Set the **Image Name** and **Image Version**.
You can select a recommended image from the Container Service (click **Select image**. Click the desired image and click **OK**) or enter the information of your own image. By default, the Container Service uses the latest version of the image. If you want to use another version of the image, click **Select image version**, click the desired version and click **OK**.

Set the number of containers (**Scale**).

Set the **Network Mode**.
Currently, the Container Service supports two network modes: **Default** and **host**. If you do not set this parameter, the **Default** mode is used by default.

Set the **Restart** parameter, namely whether to restart the container automatically in case of exception.

Set the launch command (**Command** and **Entrypoint**) of the container.
If specified, this will overwrite the image configuration.

Set the resource limits (**CPU Limit** and **Memory Limit**) of the container.
For more details of container resource limits, refer to Limit container resources.

Set the **Port Mapping**, **Web Routing** and **Load Balancer** parameters.

Set the container **Data Volume**.

Set the **Environment** variables.

Set the container **Labels**.
For more information on container labels, refer to Label description.

Set whether to enable container **Smooth Upgrade**.
For more details, refer to the description of rolling_updates in **Label description**.

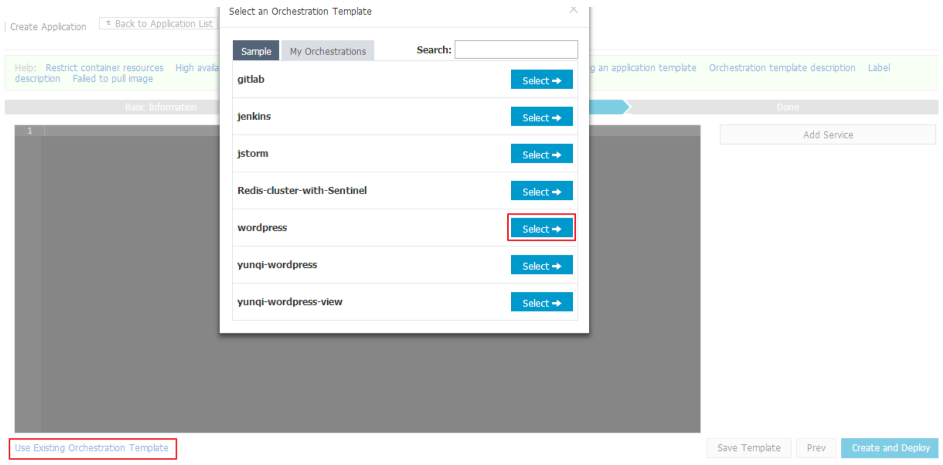
Set the container **Across Multiple zones** settings.
You can select **Ensure** to distribute the containers in two different zones; if you select this option, the container creation fails if there are less than two zones in the current cluster or if the containers cannot be distributed in two different zones due to limited machine resources. If you select **Try best**, the Container Service will distribute the containers in two different zones as long as possible and the containers will still be created successfully even if they cannot be deployed in two different zones.
If you do not set this setting, the Container Service will distribute the containers in a single zone by default.
For more information, refer to High-availability scheduling.

Set the container **Auto Scaling** rules.
For more details on how to set container auto scaling, refer to **Container auto scaling**.

Click **Create** and the Container Service creates the application according to the preceding settings.

Click **Create with Orchestration Template**.

Click **Use Existing Orchestration Template**.

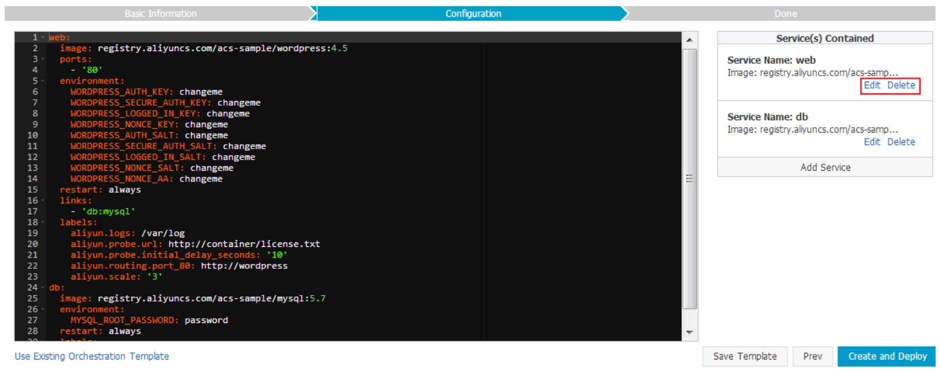


Select a template and click **Select**.

The content of the orchestration template must comply with the Docker Compose format.

Edit the template.

You can edit the orchestration template according to your needs. You can directly make modification in the template, or you can select a service at the right side and click **Edit** to make modifications or click **Delete** to delete the service.



```
web:
  image: registry.aliyuncs.com/acs-sample/wordpress:4.5
  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
    WORDPRESS_NONCE_AA: changeme
  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: '3'
db:
  image: registry.aliyuncs.com/acs-sample/mysql:5.7
  environment:
    MYSQL_ROOT_PASSWORD: password
  restart: always
```

**Service Name: web**
Image: registry.aliyuncs.com/acs-samp...

**Service Name: db**
Image: registry.aliyuncs.com/acs-samp...

In addition, you can click **Add Service**, select the desired image in the pop-up dialog box, and set desired configurations to add services to the template.

Click **Create and Deploy**.

## Restrict container resources

One of the main advantages of Docker containers is that they allow you to restrict resources, such as CPU, memory, and IO performance. Because these settings are relatively specialized, they are not shown on the interface. You can use these settings in orchestration templates.

### CPU restriction

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 parameter to specify CPU restrictions. cpu_shares: 50 indicates 0.5 cores.

### Memory restrictions

The mem_limit parameter is used to restrict memory usage. Memory is measured in bytes and the minimum memory is 4MB. If you set a memory restriction and a container applies for a memory that exceeds the limit, the container’ s operation will be stopped because of OOM.

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

```
n1:
expose:
- 443/tcp
- 80/tcp
image: 'nginx:latest'
```

```
cpu_shares: 50 #0.5 cores
mem_limit: 500000000 #500MB
labels:
aliyun.scale: '1'
restart: always
volumes:
- /var/cache/nginx
```

## Resource scheduling

In order to ensure that containers obtain sufficient specified resources, such as 0.5 CPU cores and 500MB of memory in the preceding example, we reserve 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 parameters, these containers will not be allocated resources by default.

## Other resource restrictions

For other resource restrictions, refer to the Docker Compose Instructions.

# High-availability scheduling

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

You can select the zone in the orchestration file using the environment variables and there are two formats.

- availability:az==3: The service is distributed in at least three zones. If the current cluster does not have three zones or the machine resources are insufficient for distribution in three zones, creating a container may fail.
- availability:az==~3: The service is distributed in three zones if possible. If this is not possible, the container can still be created.

The service should be distributed in at least two zones, as shown in the following example.

```
nnn:
expose:
- 443/tcp
- 80/tcp
image: 'nginx:latest'
environment:
- 'availability:az==2'
```

```
labels:
aliyun.scale: '8'
restart: always
volumes:
- /var/cache/nginx
```

## Specified node scheduling

To deploy a service on a specified node, you can use the **constraint** keyword.

In the example below, the service is deployed on node-1.

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

The Container Service supports the following expressions:

| Expression | Description |
|---|---|
| constraint:aliyun.node_index==1 | Deploy the service on node1. |
| constraint:aliyun.node_index!=1 | Do not deploy the service on node1. |
| constraint:aliyun.node_index==(1\|2\|3) | Deploy the service on node1, node2, or node3. |
| constraint:aliyun.node_index!=(1\|2\|3) | Deploy the service on a machine other than node1, node2, and node3. |
| affinity:image==~redis | Try to deploy the service on a machine with a Redis image. |
| affinity:service!=~redis | Try not to deploy the service on a machine with a Redis image. |

## Specified nodes scheduling

If you want to deploy a service on several specified nodes, you can use user tags and the constraint keyword.

Note:

- Deployment constraint is only valid for creating new containers; it does not work for updating the configurations of existing containers.
- After you use a user tag to deploy a service, deleting the user tag will not affect the service deployed, but it will do affect the next deployment of the service. Proceed with caution when deleting user tags.

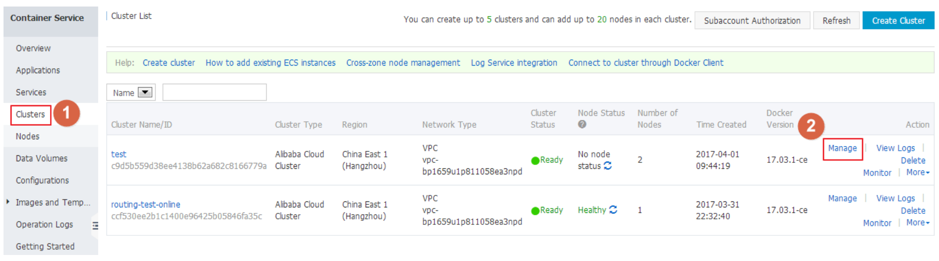
## Operating procedure

Add user tags for nodes.

Log on to the Container Service console.

Click **Clusters** in the left navigation pane.

Select the desired cluster and click **Manage**.



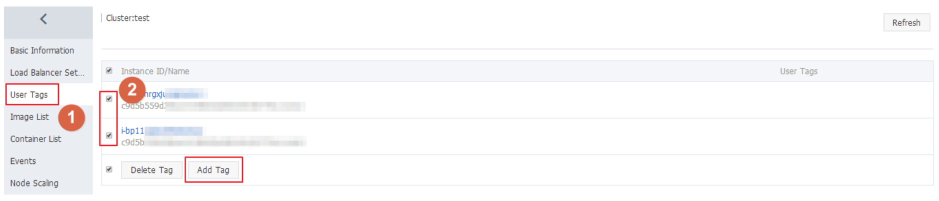
You can create up to 5 clusters and can add up to 20 nodes in each cluster.

Help: Create cluster How to add existing ECS instances Cross-zone node management Log Service integration Connect to cluster through Docker Client

| Cluster Name/ID | Cluster Type | Region | Network Type | Cluster Status | Node Status | Number of Nodes | Time Created | Docker Version | Action |
|---|---|---|---|---|---|---|---|---|---|
| test c9d5b559d38ee4138b62a682c8166779a | Alibaba Cloud Cluster | China East 1 (Hangzhou) | VPC vpc-bp1659u1p811058ea3npd | ● Ready | No node status | 2 | 2017-04-01 09:44:19 | 17.03.1-ce | Manage \| View Logs \| Delete Monitor \| More |
| routing-test-online ccf530ee2b1c1400e96425b05846fa35c | Alibaba Cloud Cluster | China East 1 (Hangzhou) | VPC vpc-bp1659u1p811058ea3npd | ● Ready | Healthy | 1 | 2017-03-31 22:32:40 | 17.03.1-ce | Manage \| View Logs \| Delete Monitor \| More |

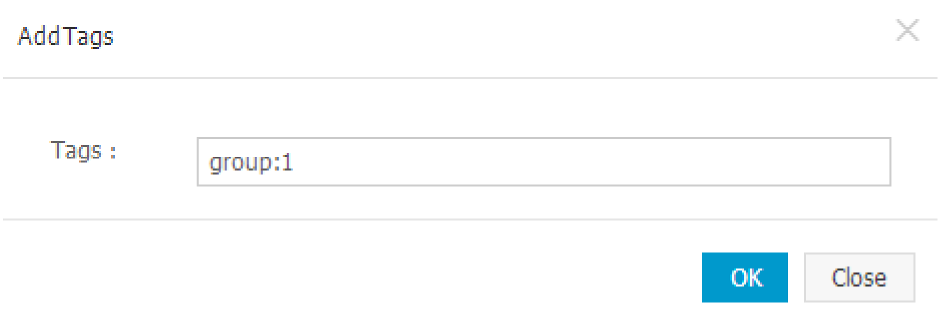
Click **User Tags** in the left navigation pane.

Select the desired nodes and click **Add Tag**.



| Instance ID/Name | User Tags |
|---|---|
| hrgxj[redacted] c9d5b559d[redacted] | |
| i-bp11[redacted] c9d5b[redacted] | |

Enter the tag key and tag value and click **OK**.


AddTags

Tags : group:1

Create an application, select **Create with Orchestration Template** and configure the constraint keyword in the template.

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

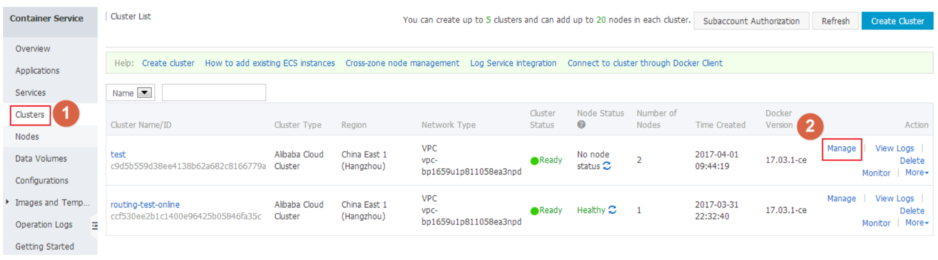
```
environment:
- constraint:group==1 #Indicates deploy the application on all the nodes with the "group:1" tag
```

## Delete a user tag

Log on to the Container Service console.

Click **Clusters** in the left navigation pane.

Select the desired cluster and click **Manage**.


Cluster List

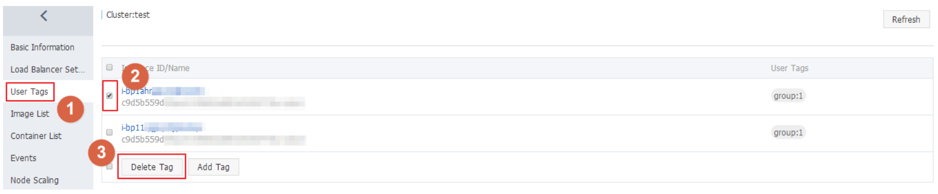
You can create up to 5 clusters and can add up to 20 nodes in each cluster.

Help: Create cluster How to add existing ECS instances Cross-zone node management Log Service integration Connect to cluster through Docker Client

| Cluster Name/ID | Cluster Type | Region | Network Type | Cluster Status | Node Status | Number of Nodes | Time Created | Docker Version | Action |
|---|---|---|---|---|---|---|---|---|---|
| test c9d5b559d38ee4138b62a682c8166779a | Alibaba Cloud Cluster | China East 1 (Hangzhou) | VPC vpc-bp1659u1p811058ea3npd | ● Ready | No node status | 2 | 2017-04-01 09:44:19 | 17.03.1-ce | Manage \| View Logs \| Delete Monitor \| More |
| routing-test-online ccf530ee2b1c1400e96425b05846fa35c | Alibaba Cloud Cluster | China East 1 (Hangzhou) | VPC vpc-bp1659u1p811058ea3npd | ● Ready | Healthy | 1 | 2017-03-31 22:32:40 | 17.03.1-ce | Manage \| View Logs \| Delete Monitor \| More |

Click **User Tags** in the left navigation pane.

Select the desired nodes and click **Delete Tag**.


Cluster:test

| Instance ID/Name | User Tags |
|---|---|
| i-bp1ahr[redacted] c9d5b559d[redacted] | group:1 |
| i-bp11[redacted] c9d5b559d[redacted] | group:1 |

Click **OK** in the pop-up dialog box.

# View application details

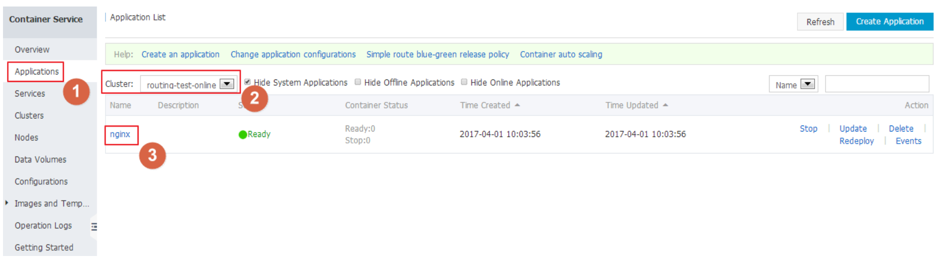
## Operating procedure

Log on to the Container Service console.

Click **Applications** in the left navigation pane.

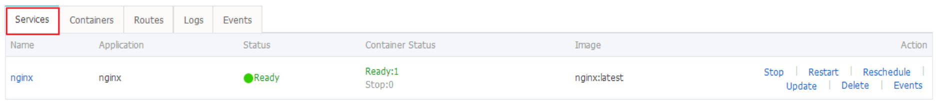
Select the cluster of the desired applications.

Click the name of the target application.



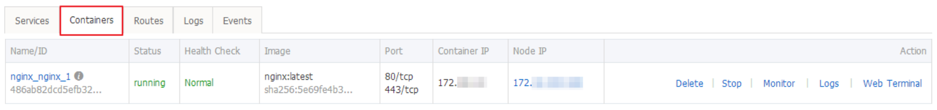
| Name | Description | S | Container Status | Time Created | Time Updated | Action |
|---|---|---|---|---|---|---|
| nginx | | ●Ready | Ready:0 Stop:0 | 2017-04-01 10:03:56 | 2017-04-01 10:03:56 | Stop \| Update \| Delete \| Redeploy \| Events |

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



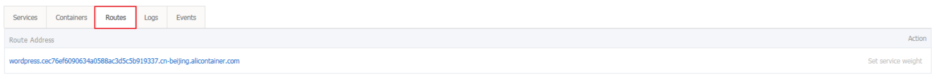
| Name | Application | Status | Container Status | Image | Action |
|---|---|---|---|---|---|
| nginx | nginx | ●Ready | Ready:1 Stop:0 | nginx:latest | Stop \| Restart \| Reschedule \| Update \| Delete \| Events |

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



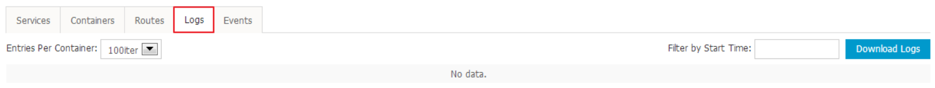
| Name/ID | Status | Health Check | Image | Port | Container IP | Node IP | Action |
|---|---|---|---|---|---|---|---|
| nginx_nginx_1 486ab82dcd5efb32... | running | Normal | nginx:latest sha256:5e69fe4b3... | 80/tcp 443/tcp | 172.[redacted] | 172.[redacted] | Delete \| Stop \| Monitor \| Logs \| Web Terminal |

Click **Routes** to view the routes of the application.



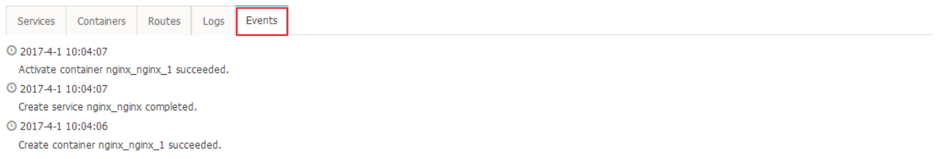
| Route Address | Action |
|---|---|
| wordpress.cec76ef6090634a0588ac3d5c5b919337.cn-beijing.alicontainer.com | Set service weight |

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



No data.

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



2017-4-1 10:04:07
Activate container nginx_nginx_1 succeeded.
2017-4-1 10:04:07
Create service nginx_nginx completed.
2017-4-1 10:04:06
Create container nginx_nginx_1 succeeded.

# Stop or activate an application

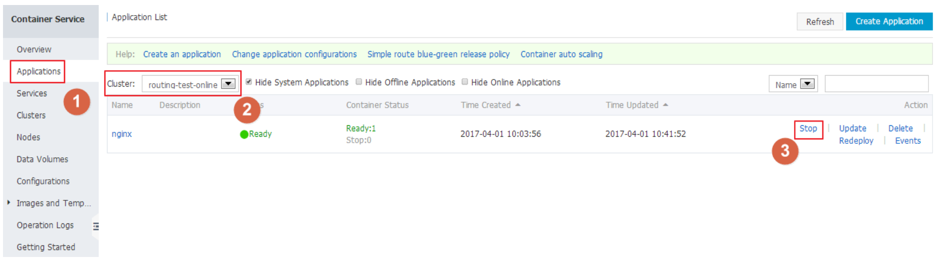
## Operating procedure

Log on to the Container Service console.

Click **Applications** in the left navigation pane.

Select the cluster of the desired application.

Locate the application to be started or stopped and click **Activate** or **Stop**.



Cluster: routing-test-online
☑ Hide System Applications

| Name | Description | s | Container Status | Time Created | Time Updated | Action |
|---|---|---|---|---|---|---|
| nginx | | ●Ready | Ready:1 Stop:0 | 2017-04-01 10:03:56 | 2017-04-01 10:41:52 | Stop \| Update \| Delete \| Redeploy \| Events |

In the pop-up dialog box, click **OK**.

# Change application configurations

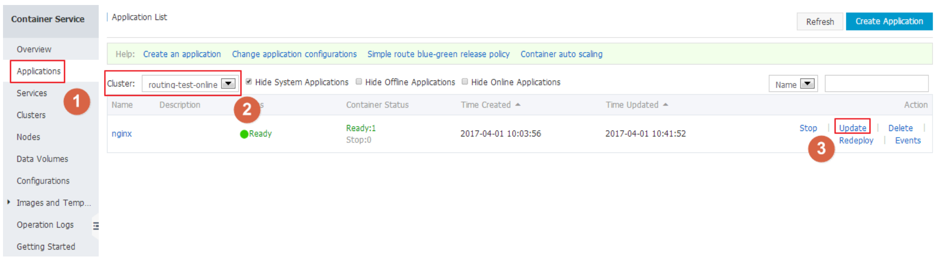
## Operating procedure

Log on to the Container Service console.

Click **Applications** in the left navigation pane.

Select the cluster of the desired application.

Locate the application to be updated and click **Update**.



| Name | Description | s | Container Status | Time Created | Time Updated | Action |
|---|---|---|---|---|---|---|
| nginx | | ●Ready | Ready:1 Stop:0 | 2017-04-01 10:03:56 | 2017-04-01 10:41:52 | Stop \| Update \| Delete \| Redeploy \| Events |

In the pop-up dialog box, modify the configuration.

**Note:** You must update **Version**; otherwise, the **OK** button is not available.

**Reschedule:** When you change the configurations of an application, in order to avoid losing the container data in the local data volume on the current machine, the Container Service will restart or recreate the container on the current machine. If you want to reschedule the container onto another machine, select **Reschedule** and the Container Service will create the container on another machine according to your setting in the **Template**.

**Note:** Selecting **Reschedule** to reschedule the container onto another machine will clear the container data in the local data volume. Proceed with caution when performing this operation.

**Use Existing Orchestration Template**: You can click to select a template to change the application configurations.

**Note:** The new template will overwrite the current template.

Click **OK**.

## Subsequent operation

After you updated the application configurations, if the application is not updated, you can redeploy it to apply the configuration modifications. For more information about how to redeploy an application, refer to Redeploy an application.

# Redeploy an application

After deploying an application, you can redeploy it if needed. Redeploying an application will pull the image used by the application; therefore, if you updated the application image after you deployed the application, redeployment will use the updated image to deploy the application.

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

You might use the redeployment function in the following situations:

- You updated your image after the application was deployed and want to deploy the application again according to the updated image.
- You stopped or deleted some containers and want to activate or recreate those containers. During redeployment, the Container Service activates the stopped containers or creates the deleted containers.

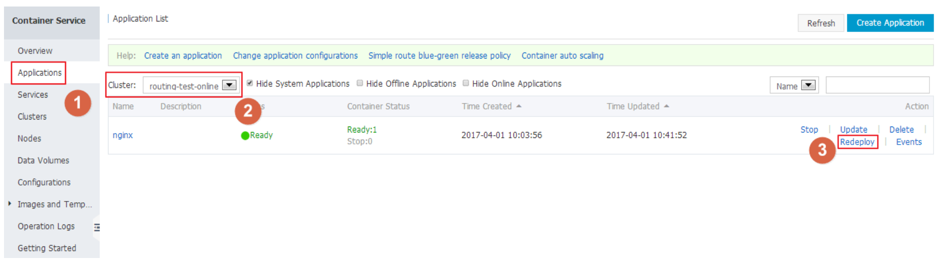
## Operating procedure

Log on to the Container Service console.

Click **Application** in the left navigation pane.

Select the cluster of the application.

Select the application to be redeployed and click **Redeploy** at the right.



| Name | Description | Status | Container Status | Time Created | Time Updated | Action |
|---|---|---|---|---|---|---|
| nginx | | ●Ready | Ready:1 Stop:0 | 2017-04-01 10:03:56 | 2017-04-01 10:41:52 | Stop \| Update \| Delete \| Redeploy \| Events |

Click **OK** in the pop-up confirmation dialog box.

## Check whether the redeployment succeeds

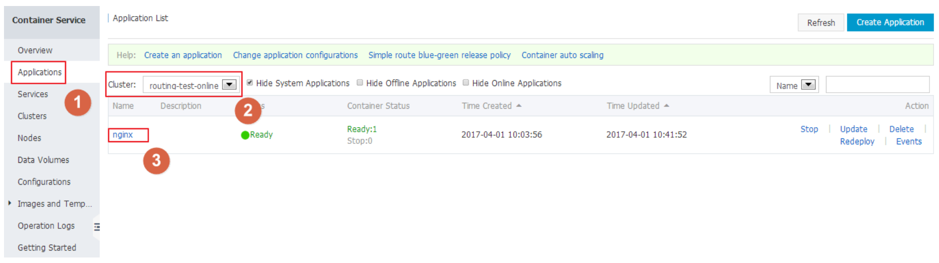
To confirm whether the deployment is successful, you can view the image sha256 to check whether the image of the redeployed container is the newest image.

Log on to the Container Service console.

Click **Application** in the left navigation pane.

Select the cluster of the application.

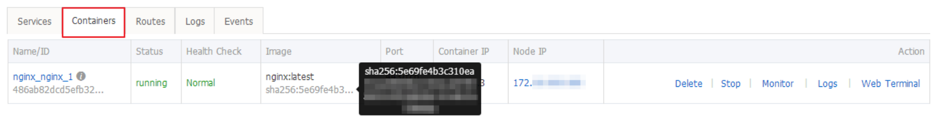
Click the name of the application.



| Name | Description | Status | Container Status | Time Created | Time Updated | Action |
|---|---|---|---|---|---|---|
| nginx | | ● Ready | Ready:1 Stop:0 | 2017-04-01 10:03:56 | 2017-04-01 10:41:52 | Stop \| Update \| Delete \| Redeploy \| Events |

Click **Containers** to view the image sha256.

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



| Name/ID | Status | Health Check | Image | Port | Container IP | Node IP | Action |
|---|---|---|---|---|---|---|---|
| nginx_nginx_1 486ab82dcd5efb32... | running | Normal | nginx:latest sha256:5e69fe4b3... | [redacted] | [redacted] | 172.[redacted] | Delete \| Stop \| Monitor \| Logs \| Web Terminal |

sha256:5e69fe4b3c310ea [redacted]

# Delete an application

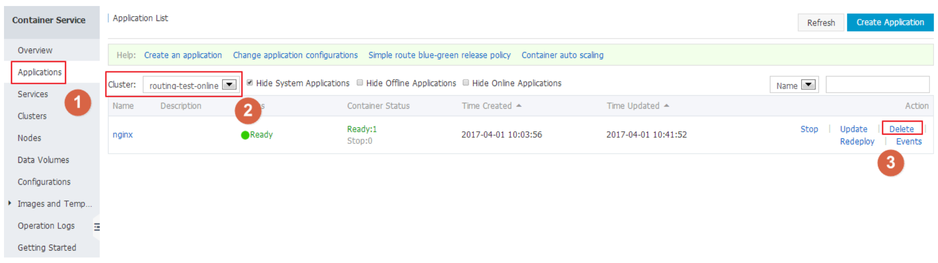
## Operating procedure

Log on to the Container Service console.

Click **Applications** in the left navigation pane.

Select the cluster of the desired application.

Locate the application to be deleted and click **Delete**.



In the pop-up dialog box, click **OK**.

# Run offline tasks

In terms of online applications, especially stateless ones, Docker containers have become the predominant standard for the execution layer, with many cloud service providers now providing container services. However, in the offline computing field, very few service providers offer this capability.

Offline computing is widely used in production, from individual script tasks to big data analysis. Offline computing requires stronger resource isolation and environment isolation. This is precisely an advantage of Docker containers. Alibaba Cloud Container Service abstracts basic offline computing models and introduces their functions based on Docker containers. Its core functions are as follows.

- Job orchestration
- Job scheduling and lifecycle management
- Integration of storage and other functions

## Basic concepts

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

| Concept | Offline applications | Online applications |
|---|---|---|
| Container | Task execution element | Service execution element |
| Operation history | 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 | A combination of several |

| | | |
|---|---|---|
| | tasks | services |

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 is used to describe and orchestrate jobs. Docker Compose supports the vast majority of Docker functions, such as the following:

- CPU, memory, and other resource restrictions
- Volumes
- Environment variables and labels
- Network models, port exposure, etc.

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

- Container quantity: The number of containers that each task is divided into.
- Retries: The number of retries made by a container.
- Remove container: Whether to delete a container after its operation is completed. You can select the following policies: remove-finished (deletes the container after the operation is finished), remove-failed (deletes failed containers), remove-all (deletes all containers), and remove-none (do not delete).
- DAG model task dependencies: There may be dependencies between the tasks in a single job. Tasks that others are dependent 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 version 2 is supported.
- At the job level, you must add the label aliyun.project_type: "batch". If this label is not added or its value is not batch, the job is considered 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 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 status of a container is determined by the container operation and exit status. The status of a task is determined by the status of all the containers in the task. The status of a job is determined by the statuses of all the tasks in the job.

### Container statuses

- Running: The container is running.
- Finished: The container has exited and ExitCode==0.
- Failed: The container has exited and ExitCode!=0.

### Task statuses

- Running: A container is running.
- Finished: All containers have finished their tasks.
- Failed: The number of failed containers exceeds the set value.

### Job statuses

- Running: A task is running.
- Finished: All tasks have been finished.
- Failed: A task has failed.

The preceding statuses can all be retrieved through the API to facilitate O&M automation.

## Shared storage

Data are 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 the Container Service, two types of shared storage can be used. Their features and application scenarios are compared below.

| Storage | Pros | Cons | Applicability |
|---|---|---|---|
| OSSFS volumes | Cross-host sharing. | Low reading, writing, and ls performance; Modifying a file will overwrite it. | Sharing configuration files; Attachment uploads. |
| A user’ s integrated third-party storage, such as Portworx | Virtually combines the cloud disks in the cluster into a large shared disk; High performance; Snapshots, multiple copies. | Requires certain O&M capabilities. | Data sharing for IO-intensive applications, such as file servers; Fast migration for IO-intensive applications, such as databases. |

## Integrated monitoring service

Monitoring is an important tool used to analyze offline jobs. Alibaba Cloud Container Service integrates the CloudMonitor function. Simply add a label in the orchestration template to collect container CPU, memory and other data to CloudMonitor. For specific instructions, refer to Container monitoring service.

## Operating procedure

Log on to the Container Service console.

Create a cluster.

For information on how to create a cluster, refer to Create a cluster.

Create an application using the orchestration template above.

For information on how to create a cluster, refer to Create an application.

Click the application name in the **Application List** page to view the application operation status.

# Timing task

You can select one or more machines and realize timing tasks through crontab. However, for large-scale or a large amount of timing tasks, the preceding method has limitations, 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 cannot be balanced.
- No retry mechanism. Tasks might fail.
- Cannot run large-scale distributed tasks.

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

**Note:** You can only use this function in case you updated your Agent since October 25th 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 simply by adding the aliyun.schedule label in the orchestration template.

```
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
```

Wherein, aliyun.schedule: "0-59/30 * * * * *" indicates executing this task every 30 seconds; the format of schedule is the same as that of crontab and uses Beijing time.

As the timing function is only applicable to offline tasks, once you add the aliyun.schedule label, the Container Service will automatically add the aliyun.project_type: "batch" label. Therefore, in the example above, you can skip the aliyun.project_type: "batch" label.

In addition, all the functions of offline tasks can still be used in timing tasks (for example, scale, retry_count, remove_containers and so on). For the meanings of these labels, refer to Run offline tasks .

### Execution procedure

After a timing task is created, it is in the “wait” state. When the specified time of the task is reached, the task is started and its subsequent states are the same with those of offline tasks. The application repeats these states when the next execution time is reached.

For the same timing task, only a single instance can be executed at a time. If the execution time of the task is longer than its execution period (for example, if the execution time of the above 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 **Events** in the application details page to view the execution history. Only the last 10 events are kept.

### High availability

The timing task control adopts the master-slave mode. When the master controller fails, the control function will be switched to the slave controller.

If the task is due to be executed during the master-slave switch period, it will be delayed and be executed until the switch is completed. If the same task is due to be executed for several times during the master-slave switch period, the task will only be executed 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.

## Default system application list

| Default system application | Included service | Brief description |
|---|---|---|
| acsrouting | routing | Layer-7 request routing service, which consists of Server Load Balancer and a HAProxy container. After a domain name is correctly configured, requests can be routed to the specified container. For how to use acsrouting, refer to Routing FAQs. |
| acslogging | logtail and logspout | Used in conjunction with Alibaba Cloud Log Service to upload the logs which are printed by application programs in containers to the Log Service for storage, making log search and analysis easier. |

| | | |
|---|---|---|
| acsmonitoring | acs-monitoring-agent | Integrated with Alibaba CloudMonitor and popular third-party open source monitoring frameworks to support monitoring information query and monitoring alarm configuration. For how to use acsmonitoring, refer to Monitoring service. |
| acsvolumedriver | volumedriver | Integrated with Alibaba Cloud Object Storage Service (OSS) to enable use of shared storage as data volumes and remove the need for stateful container O&M. For how to use acsvolumedriver, refer to Data volume service. |

# Service management

## Instructions for use

An application is composed of one or more services. You can either change the application configuration to upgrade the application, or you can change the configuration of an individual service to upgrade it.

## View service details

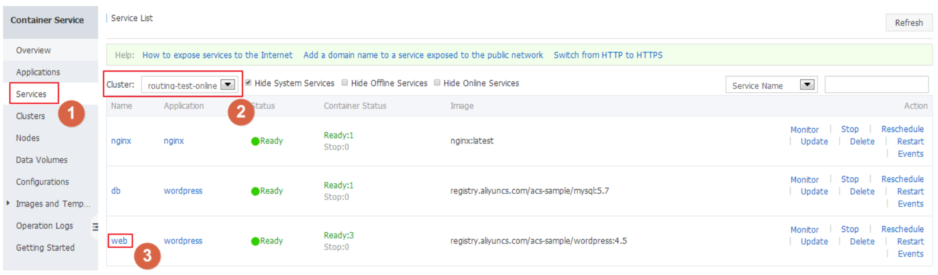
### Operating procedure

Log on to the Container Service console.

Click **Services** in the left navigation pane.

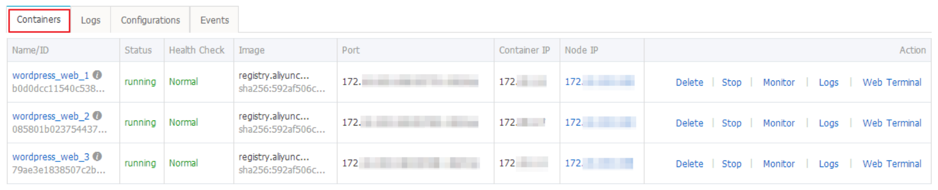
Select the cluster of the service to be viewed.

Click the name of the target service.



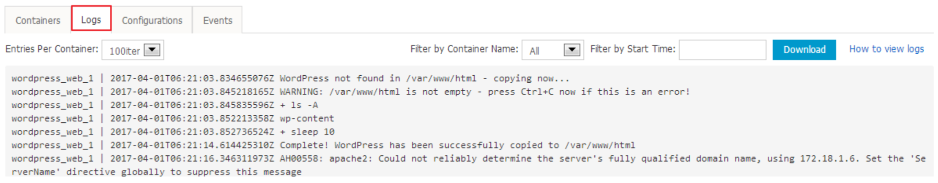
| Name | Application | Status | Container Status | Image | Action |
|---|---|---|---|---|---|
| nginx | nginx | Ready | Ready:1 Stop:0 | nginx:latest | Monitor \| Stop \| Reschedule \| Update \| Delete \| Restart \| Events |
| db | wordpress | Ready | Ready:1 Stop:0 | registry.aliyuncs.com/acs-sample/mysql:5.7 | Monitor \| Stop \| Reschedule \| Update \| Delete \| Restart \| Events |
| web | wordpress | Ready | Ready:3 Stop:0 | registry.aliyuncs.com/acs-sample/wordpress:4.5 | Monitor \| Stop \| Reschedule \| Update \| Delete \| Restart \| Events |

The service details page is displayed and you can view all the containers for the service.



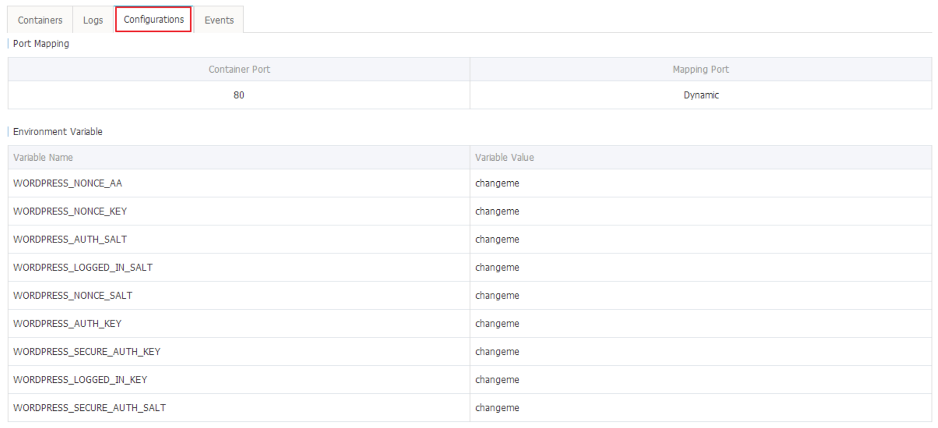
| Name/ID | Status | Health Check | Image | Port | Container IP | Node IP | Action |
|---|---|---|---|---|---|---|---|
| wordpress_web_1 b0d0dcc11540c538... | running | Normal | registry.aliyunc... sha256:592af506c... | 172.[redacted] | 172.[redacted] | 172.[redacted] | Delete \| Stop \| Monitor \| Logs \| Web Terminal |
| wordpress_web_2 085801b023754437... | running | Normal | registry.aliyunc... sha256:592af506c... | 172.[redacted] | 172.[redacted] | 172.[redacted] | Delete \| Stop \| Monitor \| Logs \| Web Terminal |
| wordpress_web_3 79ae3e1838507c2b... | running | Normal | registry.aliyunc... sha256:592af506c... | 172[redacted] | 172[redacted] | 172.[redacted] | Delete \| Stop \| Monitor \| Logs \| Web Terminal |

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



```
wordpress_web_1 | 2017-04-01T06:21:03.834655076Z WordPress not found in /var/www/html - copying now...
wordpress_web_1 | 2017-04-01T06:21:03.845218165Z WARNING: /var/www/html is not empty - press Ctrl+C now if this is an error!
wordpress_web_1 | 2017-04-01T06:21:03.845835596Z + ls -A
wordpress_web_1 | 2017-04-01T06:21:03.852213358Z wp-content
wordpress_web_1 | 2017-04-01T06:21:03.852736524Z + sleep 10
wordpress_web_1 | 2017-04-01T06:21:14.614425310Z Complete! WordPress has been successfully copied to /var/www/html
wordpress_web_1 | 2017-04-01T06:21:16.346311973Z AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using 172.18.1.6. Set the 'Se
rverName' directive globally to suppress this message
```

Click **Configurations** to view the configuration information of the service.



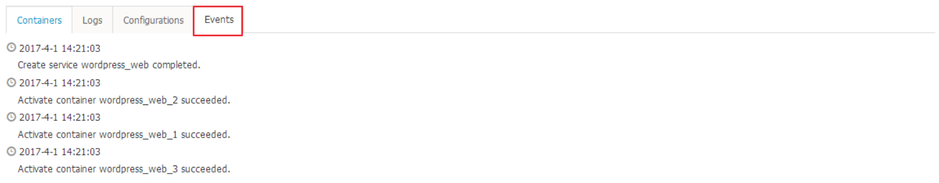
Port Mapping

| Container Port | Mapping Port |
|---|---|
| 80 | Dynamic |

Environment Variable

| Variable Name | Variable Value |
|---|---|
| WORDPRESS_NONCE_AA | changeme |
| WORDPRESS_NONCE_KEY | changeme |
| WORDPRESS_AUTH_SALT | changeme |
| WORDPRESS_LOGGED_IN_SALT | changeme |
| WORDPRESS_NONCE_SALT | changeme |
| WORDPRESS_AUTH_KEY | changeme |
| WORDPRESS_SECURE_AUTH_KEY | changeme |
| WORDPRESS_LOGGED_IN_KEY | changeme |
| WORDPRESS_SECURE_AUTH_SALT | changeme |

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



2017-4-1 14:21:03
Create service wordpress_web completed.
2017-4-1 14:21:03
Activate container wordpress_web_2 succeeded.
2017-4-1 14:21:03
Activate container wordpress_web_1 succeeded.
2017-4-1 14:21:03
Activate container wordpress_web_3 succeeded.

# Activate or stop a service

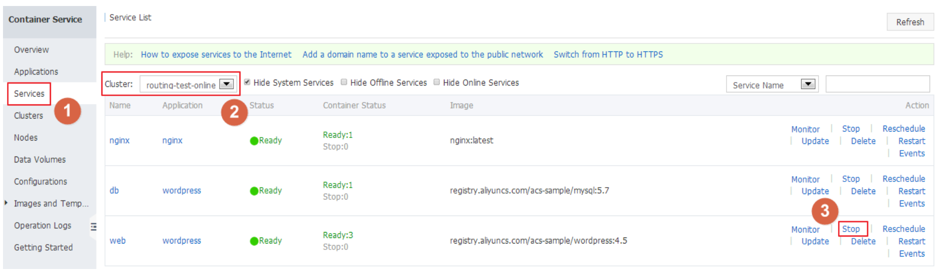
## Operating procedure

1. Log on to the Container Service console.

Click **Services** in the left navigation pane.

Select the cluster of the desired service.

Locate the service to be started or stopped and click **Activate** or **Stop**.



| Name | Application | Status | Container Status | Image | Action |
|---|---|---|---|---|---|
| nginx | nginx | ● Ready | Ready:1 Stop:0 | nginx:latest | Monitor \| Stop \| Reschedule \| Update \| Delete \| Restart \| Events |
| db | wordpress | ● Ready | Ready:1 Stop:0 | registry.aliyuncs.com/acs-sample/mysql:5.7 | Monitor \| Stop \| Reschedule \| Update \| Delete \| Restart \| Events |
| web | wordpress | ● Ready | Ready:3 Stop:0 | registry.aliyuncs.com/acs-sample/wordpress:4.5 | Monitor \| Stop \| Reschedule \| Update \| Delete \| Restart \| Events |

In the pop-up dialog box, click **OK**.

# Change service configuration

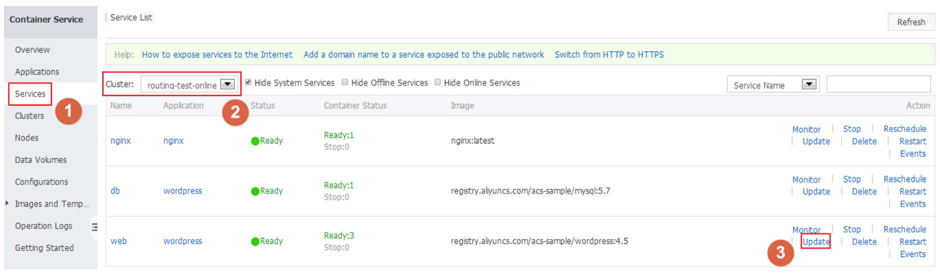
## Operating procedure

Log on to the Container Service console.

Click **Services** in the left navigation pane.

Select the cluster of the desired service.

Locate the service to be reconfigured and click **Update**.



Container Service

Overview
Applications
Services
Clusters
Nodes
Data Volumes
Configurations
Images and Temp...
Operation Logs
Getting Started

Service List

Refresh

Help: How to expose services to the Internet Add a domain name to a service exposed to the public network Switch from HTTP to HTTPS

Cluster: routing-test-online ✓ Hide System Services Hide Offline Services Hide Online Services Service Name

| Name | Application | Status | Container Status | Image | Action |
|---|---|---|---|---|---|
| nginx | nginx | ● Ready | Ready:1 Stop:0 | nginx:latest | Monitor \| Stop \| Reschedule \| Update \| Delete \| Restart \| Events |
| db | wordpress | ● Ready | Ready:1 Stop:0 | registry.aliyuncs.com/acs-sample/mysql:5.7 | Monitor \| Stop \| Reschedule \| Update \| Delete \| Restart \| Events |
| web | wordpress | ● Ready | Ready:3 Stop:0 | registry.aliyuncs.com/acs-sample/wordpress:4.5 | Monitor \| Stop \| Reschedule \| Update \| Delete \| Restart \| Events |

In the pop-up dialog box, modify the configuration.

Click **OK**.

# Reschedule a service

To rebalance the number of operational containers on each node, this operation moves containers from nodes with high loads to the newly created nodes or nodes with low loads. This rebalances the cluster load.

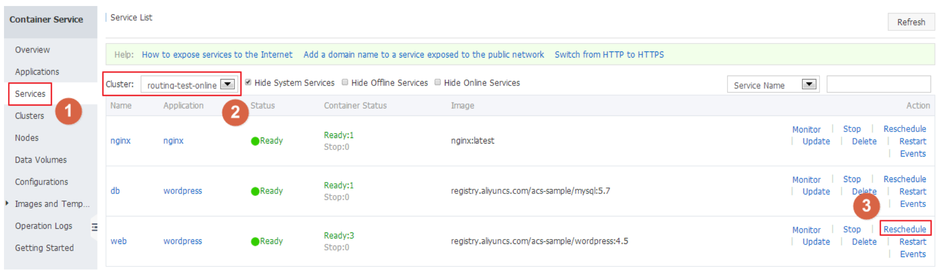
## Operating procedure

Log on to the Container Service console.

Click **Services** in the left navigation pane.

Select the cluster of the desired service.

Locate the service to be rescheduled and click **Reschedule**.



Container Service

Overview
Applications
Services
Clusters
Nodes
Data Volumes
Configurations
Images and Temp...
Operation Logs
Getting Started

Service List

Refresh

Help: How to expose services to the Internet Add a domain name to a service exposed to the public network Switch from HTTP to HTTPS

Cluster: routing-test-online ✓ Hide System Services Hide Offline Services Hide Online Services Service Name

| Name | Application | Status | Container Status | Image | Action |
|---|---|---|---|---|---|
| nginx | nginx | ● Ready | Ready:1 Stop:0 | nginx:latest | Monitor \| Stop \| Reschedule \| Update \| Delete \| Restart \| Events |
| db | wordpress | ● Ready | Ready:1 Stop:0 | registry.aliyuncs.com/acs-sample/mysql:5.7 | Monitor \| Stop \| Reschedule \| Update \| Delete \| Restart \| Events |
| web | wordpress | ● Ready | Ready:3 Stop:0 | registry.aliyuncs.com/acs-sample/wordpress:4.5 | Monitor \| Stop \| Reschedule \| Update \| Delete \| Restart \| Events |

In the pop-up dialog box, select the two parameters and click **OK**.

- **Ignore Local Volumes:** During rescheduling, containers with local volumes may be migrated to other machines, resulting in a loss of data. To ignore local data volumes, select this option. Otherwise, containers with local volumes will not be rescheduled.
- **Force Reschedule:** In order to ensure the stability of online services, by default, only machines with memory usage over 60% and CPU usage over 40% will be rescheduled. To remove this restriction, select this option. This will reschedule the machines by force ignoring their resource usage.

**Note:** The memory and CPU usage values may vary depending on the container configuration. Therefore, they do not always represent the actual usage conditions.

# Delete a service

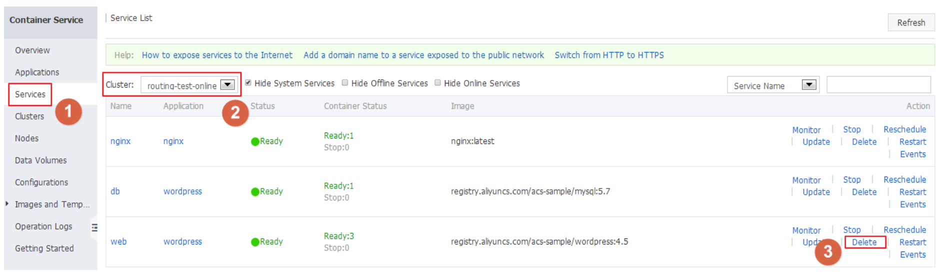
## Operating procedure

Log on to the Container Service console.

Click **Services** in the left navigation pane.

Select the cluster of the desired service.

Locate the service to be deleted and click **Delete**.



| Name | Application | Status | Container Status | Image | Action |
|---|---|---|---|---|---|
| nginx | nginx | ●Ready | Ready:1 Stop:0 | nginx:latest | Monitor \| Stop \| Reschedule \| Update \| Delete \| Restart \| Events |
| db | wordpress | ●Ready | Ready:1 Stop:0 | registry.aliyuncs.com/acs-sample/mysql:5.7 | Monitor \| Stop \| Reschedule \| Update \| Delete \| Restart \| Events |
| web | wordpress | ●Ready | Ready:3 Stop:0 | registry.aliyuncs.com/acs-sample/wordpress:4.5 | Monitor \| Stop \| Reschedule \| Update \| Delete \| Restart \| Events |

In the pop-up dialog box, click **OK**.

# Network management

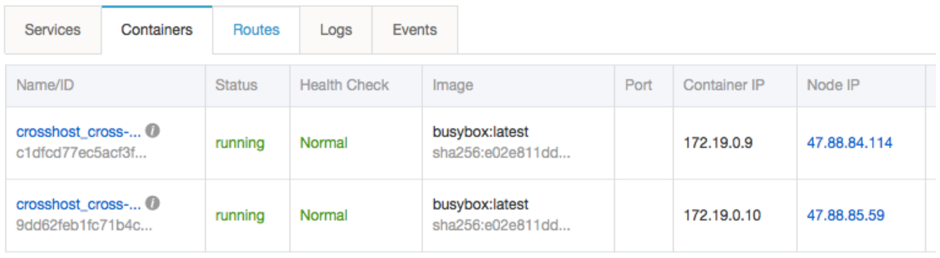
## Cross-host container network

The Container Service creates the global network for the container, and the container in the same cluster can access other containers through eth0 network interface of the container.

For example, create the container respectively on two machines and print their IP addresses.

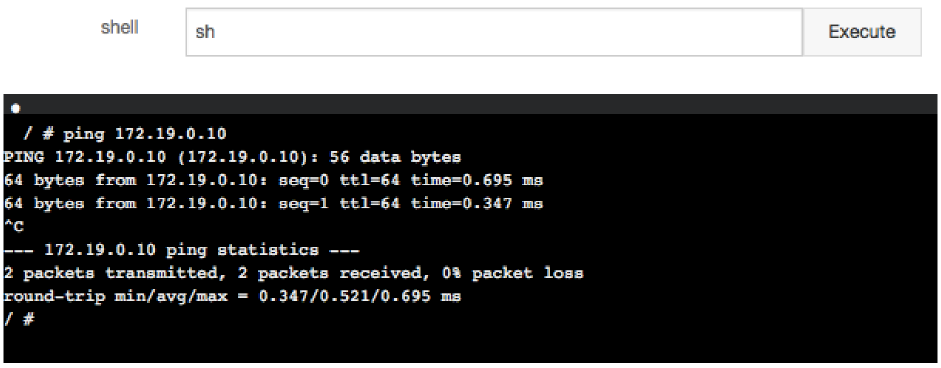
```
cross-host-network-test1:
image: busybox
command: sh -c 'ifconfig eth0; sleep 100000'
tty: true
environment:
- 'constraint:aliyun.node_index==(1)'

cross-host-network-test2:
image: busybox
command: sh -c 'ifconfig eth0; sleep 100000'
tty: true
environment:
- 'constraint:aliyun.node_index==(2)'
```

You can see that the containers of these two services are distributed in different nodes.



| Name/ID | Status | Health Check | Image | Port | Container IP | Node IP |
|---|---|---|---|---|---|---|
| crosshost_cross-... c1dfcd77ec5acf3f... | running | Normal | busybox:latest sha256:e02e811dd... | | 172.19.0.9 | 47.88.84.114 |
| crosshost_cross-... 9dd62feb1fc71b4c... | running | Normal | busybox:latest sha256:e02e811dd... | | 172.19.0.10 | 47.88.85.59 |

Using the ifconfig eth0 log output from the console or container 2, you can see that the IP address of container 2 is 172.19.0.10. Then, you can connect to a remote terminal to access the IP address of Container 2 from Container 1.



```
 / # ping 172.19.0.10
PING 172.19.0.10 (172.19.0.10): 56 data bytes
64 bytes from 172.19.0.10: seq=0 ttl=64 time=0.695 ms
64 bytes from 172.19.0.10: seq=1 ttl=64 time=0.347 ms
^C
--- 172.19.0.10 ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max = 0.347/0.521/0.695 ms
/ #
```

# Data volume

## Overview

In Docker, containers are non-persistent. This means that after a container is deleted, its data are also deleted. Although, Docker provides volumes that can be attached to host directories for persistent storage, in a cluster environment, host volumes have limitations, such as:

- Data cannot be migrated when containers are migrated between machines.
- Volumes cannot be shared between different machines.

To resolve these issues, Alibaba Cloud Container Service provides third-party data volumes. By packaging various cloud storage resources as data volumes, the service allows these data volumes to be directly attached to containers. This way, the data volumes can be automatically reattached when a container is restarted or migrated.

Currently, this service supports OSSFS storage.

## Create an OSSFS data volume

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

Because data must be synchronized to the cloud over the network, the performance and functions of OSSFS differ from those of local file systems. It is recommended that you do not run databases and other IO-intensive applications, or logs and other applications that require constantly writing files onto OSSFS. OSSFS is instead better suited for sharing configuration files across containers, attachment uploads, and other scenarios that do not require rewrite operations.

**OSSFS differs from local file systems in the following ways:**

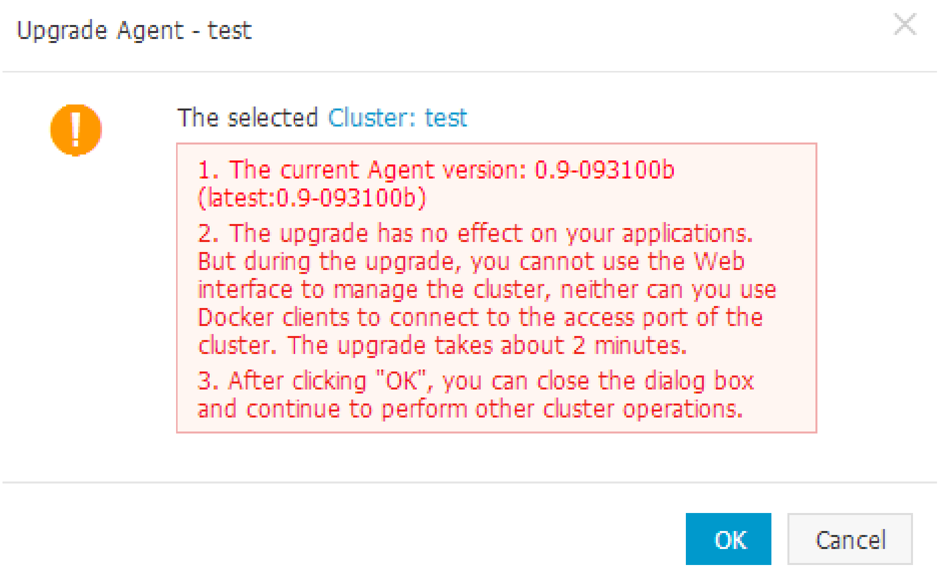
- Random and append write operations may overwrite the entire file.
- Metadata operations, such as list directory, provide poor performance because the system needs to remotely access the OSS server.
- The file/folder rename operation is not atomic.
- When multiple clients are attached to a single OSS bucket, you must coordinate the actions of each client on your own. For example, you need to avoid multiple clients from writing to a single file.
- Hard link is not supported.

## Prerequisites

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

The cluster Agent is version 0.6 or higher.

You can view your Agent version in the **Cluster List** page. Select the desired cluster, click **More** > **Upgrade Agent** on the right.


Upgrade Agent - test

The selected Cluster: test

1. The current Agent version: 0.9-093100b (latest:0.9-093100b)
2. The upgrade has no effect on your applications. But during the upgrade, you cannot use the Web interface to manage the cluster, neither can you use Docker clients to connect to the access port of the cluster. The upgrade takes about 2 minutes.
3. After clicking "OK", you can close the dialog box and continue to perform other cluster operations.

If your Agent version is too low, you need to first upgrade your Agent. For how to upgrade the Agent, refer to Upgrade Agent.

The acsvolumedriver application must be deployed in the cluster. In addition, it is recommended that you upgrade the acsvolumedriver to the latest version.

You can deploy and upgrade the acsvolumedriver application by upgrading the system services. For detailed operations, refer to Upgrade system services.

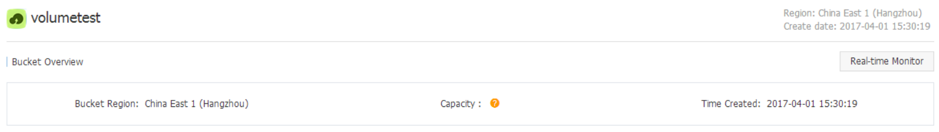
> **Note:** When upgrading or restarting the acsvolumedriver, the containers that use the OSSFS data volume will also be restarted, so will your services.

## Operating procedure

### Step 1 Create OSS bucket

Log on to the OSS console and create an OSS bucket (refer to Create a bucket).

In this example, an OSS bucket in the China East 1 (Hangzhou) region is created.


volumetest

Region: China East 1 (Hangzhou)
Create date: 2017-04-01 15:30:19

Bucket Overview

Real-time Monitor

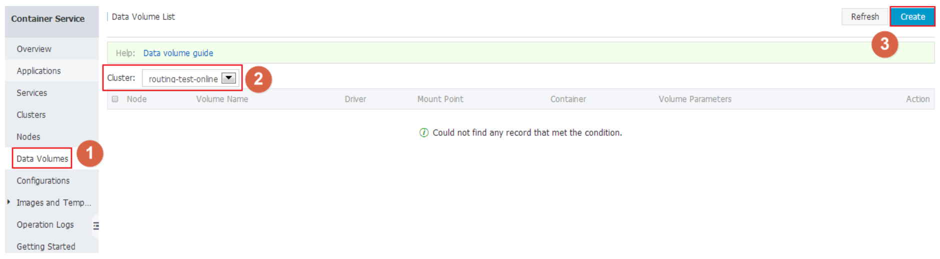
Bucket Region: China East 1 (Hangzhou) Capacity : Time Created: 2017-04-01 15:30:19

### Step 2 Create OSSFS data volume

Log on to the Container Service console.

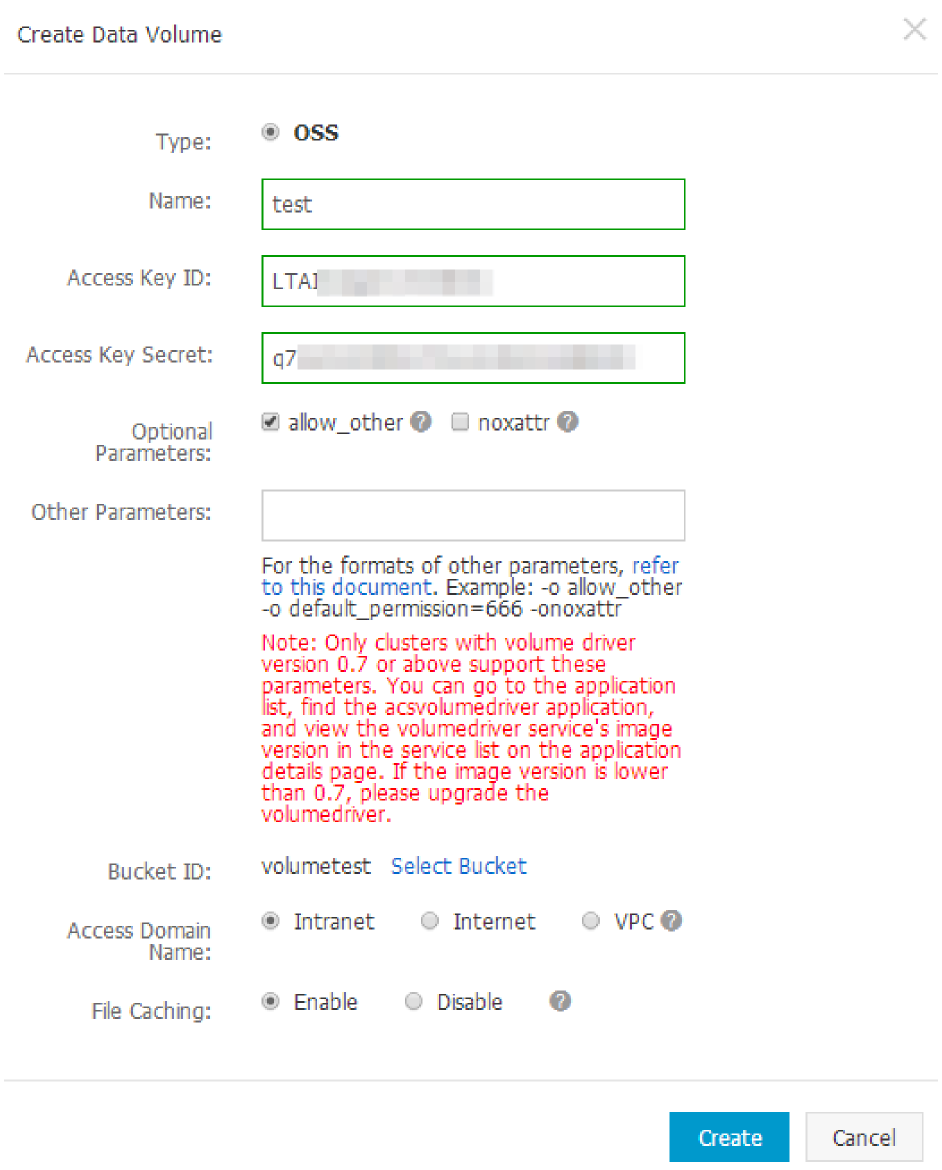
Click **Data Volumes** in the left navigation pane.

Select the target cluster and click **Create**.



Could not find any record that met the condition.

Set the data volume information and click **Create**.


Create Data Volume

| | |
|---|---|
| Type: | [✓ OSS] |
| Name: | test |
| Access Key ID: | LTAI[redacted] |
| Access Key Secret: | q7[redacted] |
| Optional Parameters: | [✓ allow_other] noxattr |
| Other Parameters: | |
| | For the formats of other parameters, refer to this document. Example: -o allow_other -o default_permission=666 -onoxattr |
| | Note: Only clusters with volume driver version 0.7 or above support these parameters. You can go to the application list, find the acsvolumedriver application, and view the volumedriver service's image version in the service list on the application details page. If the image version is lower than 0.7, please upgrade the volumedriver. |
| Bucket ID: | volumetest Select Bucket |
| Access Domain Name: | [✓ Intranet] Internet VPC |
| File Caching: | [✓ Enable] Disable |

- **Name:** This is the data volume ID and must be unique within the cluster.
- **Access Key ID**, **Access Key Secret:** These are the parts of the Access Key needed to access OSS. You can obtain them from the AccessKey console
- **Access Domain Name:** If a bucket shares a region with the ECS instance, select **Intranet** domain name. Otherwise, select **Internet** domain name.
- **File Caching:** If you need to synchronize modifications to the same file on multiple machines (for example, modify the file in Machine A and then read the modified content in Machine B), turn off file cache.

  **Note:** Turning off file cache will slow down the ls folder, especially when there are many files in the same folder. Therefore, if you do not require the above, turn on file cache to speed up the ls.

## Subsequent operations

After a data volume is created, you can use it in applications. For how to use data volumes in applications, refer to Use third party data volumes.

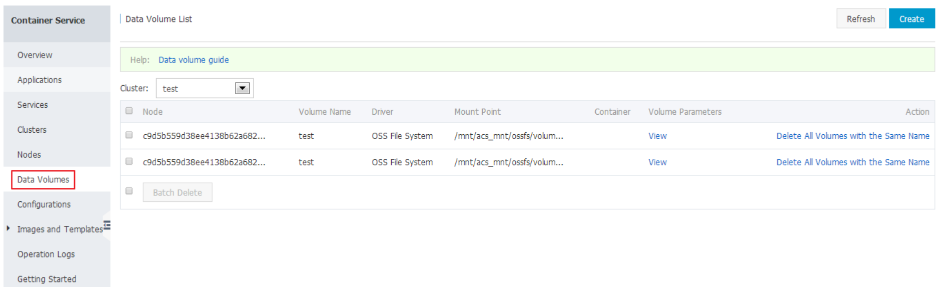
# View and delete data Volumes

You can view the data volumes created or delete the data volumes.

Log on to the Container Service console, click **Data Volumes** in the left navigation pane and select the target cluster.

All the data volumes in the current cluster are displayed, including local and third-party data volumes.

In addition, you can view the containers that reference the data volumes.



Cluster: test

| Node | Volume Name | Driver | Mount Point | Container | Volume Parameters | Action |
|---|---|---|---|---|---|---|
| c9d5b559d38ee4138b62a682... | test | OSS File System | /mnt/acs_mnt/ossfs/volum... | | View | Delete All Volumes with the Same Name |
| c9d5b559d38ee4138b62a682... | test | OSS File System | /mnt/acs_mnt/ossfs/volum... | | View | Delete All Volumes with the Same Name |

The names of local data volumes follow the format of node_name/volume_name.

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

When you create a third-party data volume, the Container Service creates a data volume on all the nodes in the cluster using the same data volume name, allowing containers to be migrated between nodes. You can delete these volumes by clicking **Delete All Volumes with the Same Name**.

> **Note:** You cannot delete a data volume when it is referenced by a container. To delete the data volume, you need to first delete the container.

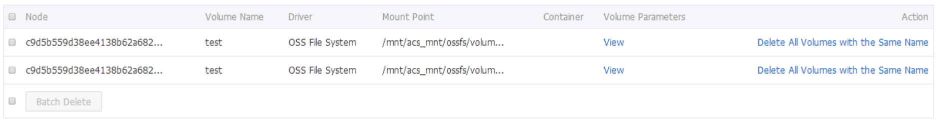
# Use third-party data volumes

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

You can set the data volume information when creating an application either by using an image or using an orchestration template.

## Operating procedure

This example makes illustrations by taking the OSSFS data volume **test** in the **test** cluster as an example.



| Node | Volume Name | Driver | Mount Point | Container | Volume Parameters | Action |
|---|---|---|---|---|---|---|
| c9d5b559d38ee4138b62a682... | test | OSS File System | /mnt/acs_mnt/ossfs/volum... | | View | Delete All Volumes with the Same Name |
| c9d5b559d38ee4138b62a682... | test | OSS File System | /mnt/acs_mnt/ossfs/volum... | | View | Delete All Volumes with the Same Name |

Batch Delete

### Create application from image

Log on to the Container Service console.

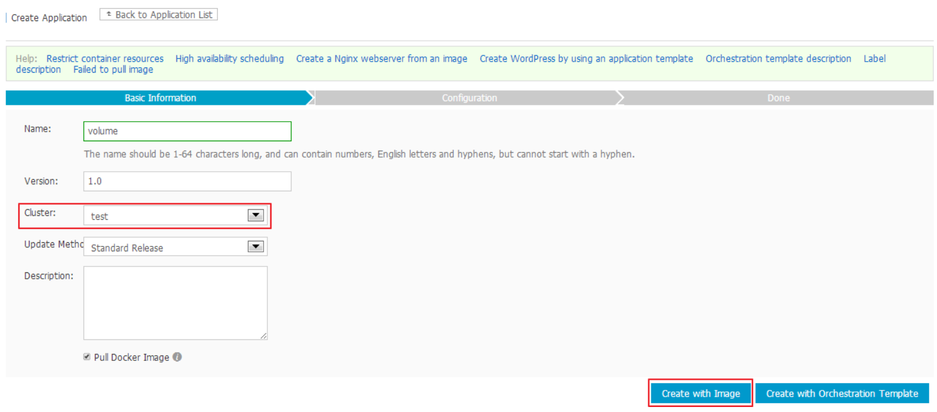
Click **Applications** in the left navigation pane.

Click **Create Application** in the upper-right corner.

Enter the basic information of the application (in this example, the application name is **volume**), and click **Create with Image**.

In this example, the **Cluster** is **test**.

> **Note:** The cluster of the application must be the same with that of the OSSFS data volume.



Create Application

Help: Restrict container resources High availability scheduling Create a Nginx webserver from an image Create WordPress by using an application template Orchestration template description Label description Failed to pull image

Name: volume

The name should be 1-64 characters long, and can contain numbers, English letters and hyphens, but cannot start with a hyphen.

Version: 1.0

Cluster: test

Update Metho Standard Release

Description:

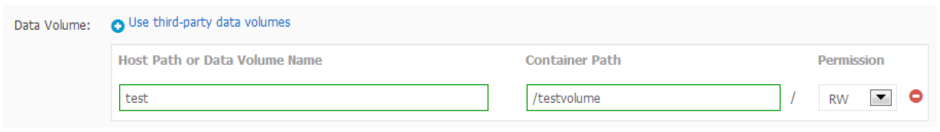
☑ Pull Docker Image

Select the desired image and set other parameters.

> **Note:** For how to create an application, refer to Create an application.

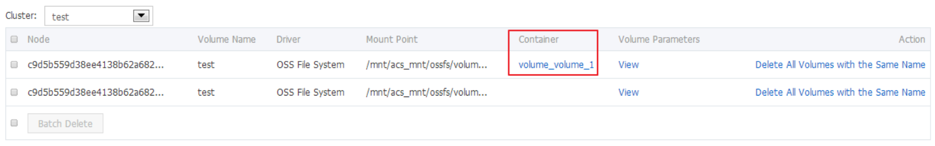
Click the plus icon in **Data Volume**, enter the data volume name in **Host Path or Data**

**Volume Name**, enter the **Container Path** and set the **Permission**.


Data Volume: Use third-party data volumes

| Host Path or Data Volume Name | Container Path | Permission |
|---|---|---|
| test | /testvolume | RW |

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


Cluster: test

| Node | Volume Name | Driver | Mount Point | Container | Volume Parameters | Action |
|---|---|---|---|---|---|---|
| c9d5b559d38ee4138b62a682... | test | OSS File System | /mnt/acs_mnt/ossfs/volum... | volume_volume_1 | View | Delete All Volumes with the Same Name |
| c9d5b559d38ee4138b62a682... | test | OSS File System | /mnt/acs_mnt/ossfs/volum... | | View | Delete All Volumes with the Same Name |

Batch Delete

## Create application from orchestration template

Log on to the Container Service console.

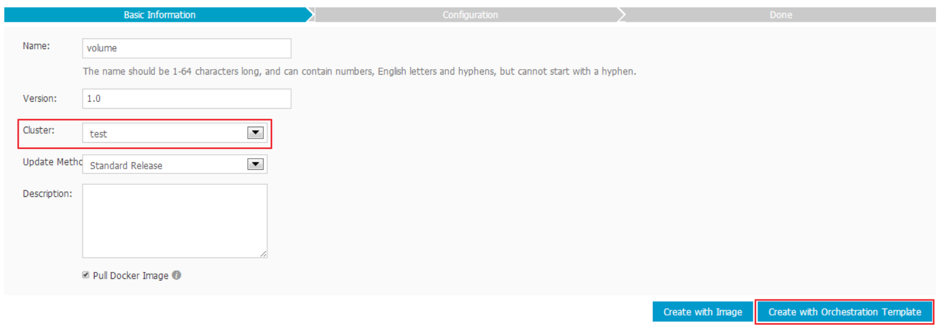
Click **Applications** in the left navigation pane.

Click **Create Application** in the upper-right corner.

Enter the basic information of the application (in this example, the application name is **volume**), and click **Create with Orchestration Template**.

In this example, the **Cluster** is **test**.

**Note:** The cluster of the application must be the same with that of the OSSFS data volume.



Name: volume

The name should be 1-64 characters long, and can contain numbers, English letters and hyphens, but cannot start with a hyphen.

Version: 1.0

Cluster: test

Update Metho Standard Release

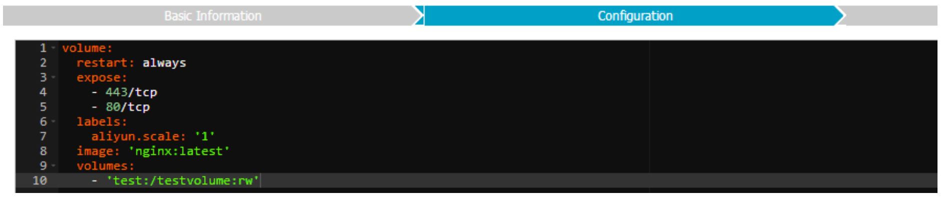
Description:

☑ Pull Docker Image

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

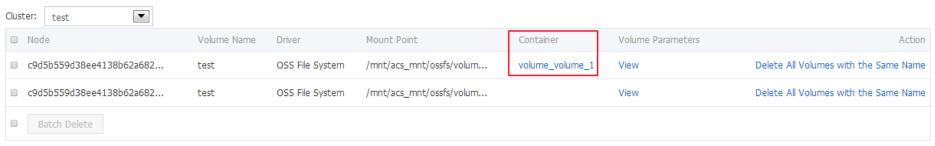
**Note :** For how to create an application, refer to Create an application.

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



```
volume:
  restart: always
  expose:
    - 443/tcp
    - 80/tcp
  labels:
    aliyun.scale: '1'
  image: 'nginx:latest'
  volumes:
    - 'test:/testvolume:rw'
```

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



Cluster: test

| Node | Volume Name | Driver | Mount Point | Container | Volume Parameters | Action |
|---|---|---|---|---|---|---|
| c9d5b559d38ee4138b62a682... | test | OSS File System | /mnt/acs_mnt/ossfs/volum... | volume_volume_1 | View | Delete All Volumes with the Same Name |
| c9d5b559d38ee4138b62a682... | test | OSS File System | /mnt/acs_mnt/ossfs/volum... |  | View | Delete All Volumes with the Same Name |

## Change the configuration of an existing application

Log on to the Container Service console.

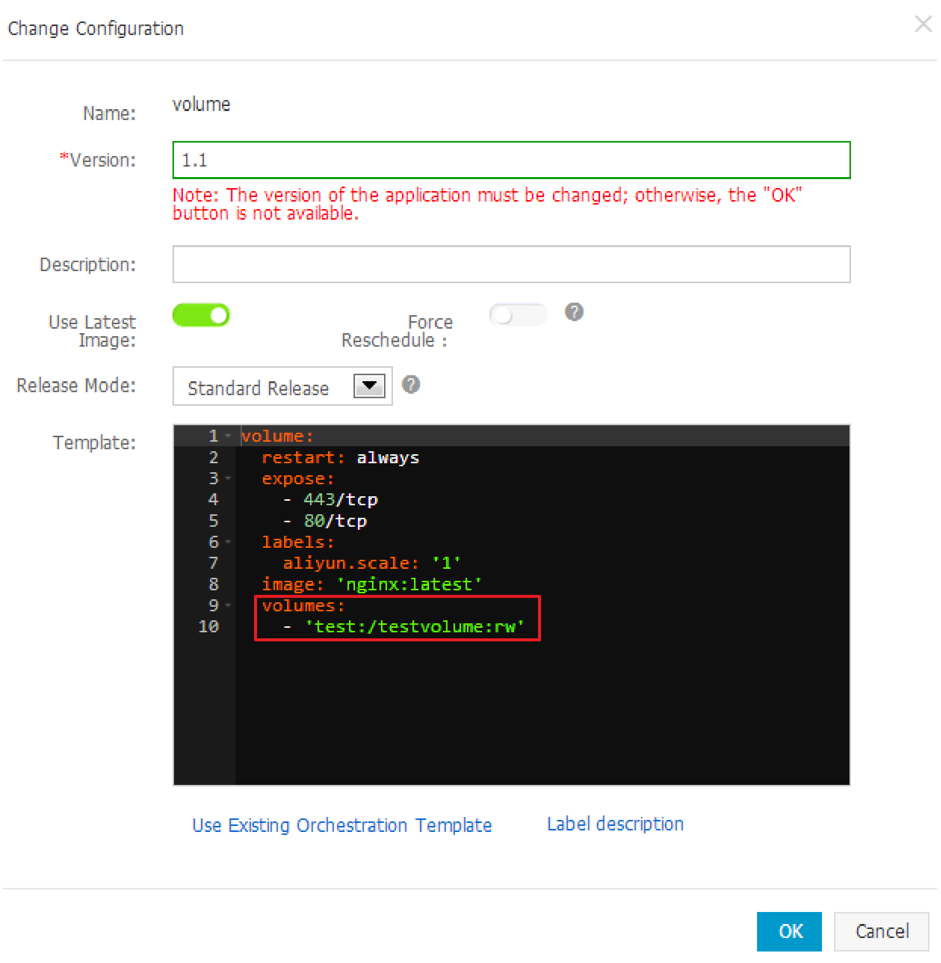
Click **Applications** in the left navigation pane.

Select the cluster (in this example, select the **test** cluster), select the desired application and Click **Update**.

For detailed information about how to change application configurations, refer to Change application configurations.

**Note:** The application must be in the same cluster as the OSSFS data volume.

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



Name: volume

*Version: 1.1

Note: The version of the application must be changed; otherwise, the "OK" button is not available.

Description:

Use Latest Image:

Force Reschedule :

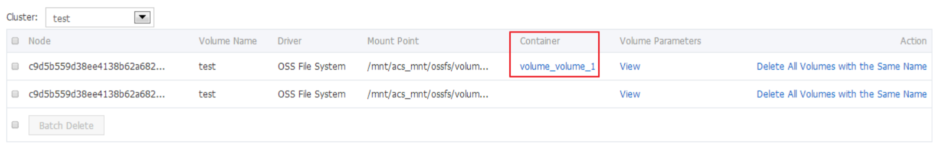
Release Mode: Standard Release

Template:

```
volume:
  restart: always
  expose:
    - 443/tcp
    - 80/tcp
  labels:
    aliyun.scale: '1'
  image: 'nginx:latest'
  volumes:
    - 'test:/testvolume:rw'
```

Click **OK**.

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



Cluster: test

| Node | Volume Name | Driver | Mount Point | Container | Volume Parameters | Action |
|---|---|---|---|---|---|---|
| c9d5b559d38ee4138b62a682... | test | OSS File System | /mnt/acs_mnt/ossfs/volum... | volume_volume_1 | View | Delete All Volumes with the Same Name |
| c9d5b559d38ee4138b62a682... | test | OSS File System | /mnt/acs_mnt/ossfs/volum... |  | View | Delete All Volumes with the Same Name |

## Troubleshooting

If you use the Volume name: directory in the image method for third-party data volumes (for example, o1:/data, when the /data directory exists in the image), the container cannot be launched. The system will report an error such as chown /mnt/acs_mnt/ossfs/XXXX: input/output error.

This error occurs because Docker will copy the files in the image to the data volume and use chown to set the relevant user permissions. However, Linux prohibits the use of chown for mount points.

To resolve this error, you can use one of the following solutions:

Upgrade Docker to version 1.11 or later, upgrade Agent to the latest version and specify nocopy in the template. Docker will not copy the files and thereby, no chown error will occur.

```
  volumes:
- o1:/data:nocopy
- /tmp:/bbb
```

If you need to copy the files, you can use the mount point path (for example, /mnt/acs_mnt/ossfs/XXXX:/data) instead of the data volume name to set the data volume. However, because this setting bypasses the volume driver, when the machine restarts, the container might be started before OSSFS is successfully mounted; in this way, a local data volume might be mounted onto the container. To avoid this issue, you need to use two data volumes with one data volume set by the data volume name and the other volume 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
```

# Log management

# View logs

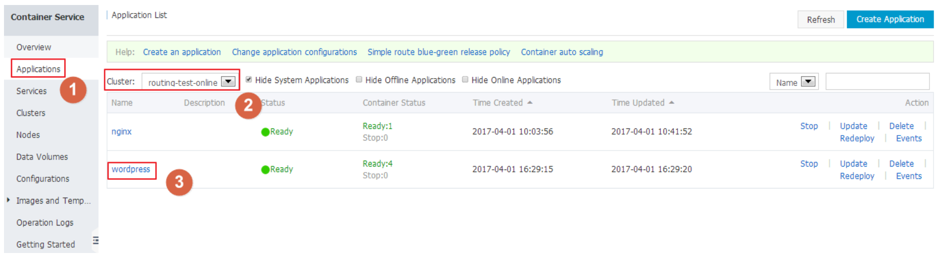
## View application logs

Log on to the Container Service console.

Click **Applications** in the left navigation pane.

Select the cluster of the desired application.

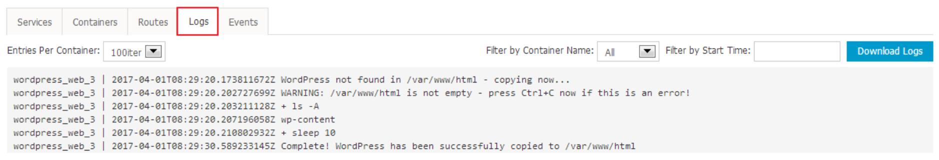
Click the name of the target application.



| Name | Description | Status | Container Status | Time Created | Time Updated | Action |
|---|---|---|---|---|---|---|
| nginx | | ● Ready | Ready:1 Stop:0 | 2017-04-01 10:03:56 | 2017-04-01 10:41:52 | Stop \| Update \| Delete \| Redeploy \| Events |
| wordpress | | ● Ready | Ready:4 Stop:0 | 2017-04-01 16:29:15 | 2017-04-01 16:29:20 | Stop \| Update \| Delete \| Redeploy \| Events |

Click **Logs** to view the application logs.

You can then select the log entries displayed, or download all logs to the local device.



```
wordpress_web_3 | 2017-04-01T08:29:20.173811672Z WordPress not found in /var/www/html - copying now...
wordpress_web_3 | 2017-04-01T08:29:20.202727699Z WARNING: /var/www/html is not empty - press Ctrl+C now if this is an error!
wordpress_web_3 | 2017-04-01T08:29:20.203211128Z + ls -A
wordpress_web_3 | 2017-04-01T08:29:20.207196058Z wp-content
wordpress_web_3 | 2017-04-01T08:29:20.210802932Z + sleep 10
wordpress_web_3 | 2017-04-01T08:29:30.589233145Z Complete! WordPress has been successfully copied to /var/www/html
```

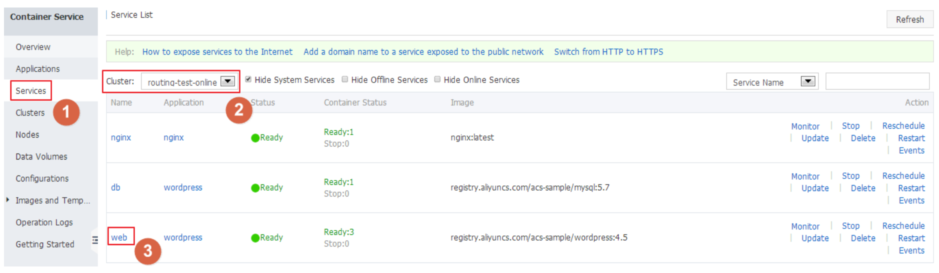
## View service logs

Log on to the Container Service console.

Click **Services** in the left navigation pane.

Select the cluster of the desired service.

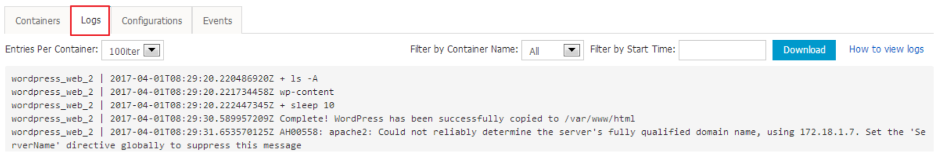
Click the name of the target service.



| Name | Application | Status | Container Status | Image | Action |
|---|---|---|---|---|---|
| nginx | nginx | ● Ready | Ready:1 Stop:0 | nginx:latest | Monitor \| Stop \| Reschedule \| Update \| Delete \| Restart \| Events |
| db | wordpress | ● Ready | Ready:1 Stop:0 | registry.aliyuncs.com/acs-sample/mysql:5.7 | Monitor \| Stop \| Reschedule \| Update \| Delete \| Restart \| Events |
| web | wordpress | ● Ready | Ready:3 Stop:0 | registry.aliyuncs.com/acs-sample/wordpress:4.5 | Monitor \| Stop \| Reschedule \| Update \| Delete \| Restart \| Events |

Click **Logs** to view the application logs.

You can select the log entries displayed, or download all logs to local device.



```
wordpress_web_2 | 2017-04-01T08:29:20.220486920Z + ls -A
wordpress_web_2 | 2017-04-01T08:29:20.221734458Z wp-content
wordpress_web_2 | 2017-04-01T08:29:20.222447345Z + sleep 10
wordpress_web_2 | 2017-04-01T08:29:30.589957209Z Complete! WordPress has been successfully copied to /var/www/html
wordpress_web_2 | 2017-04-01T08:29:31.653570125Z AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using 172.18.1.7. Set the 'Se
rverName' directive globally to suppress this message
```

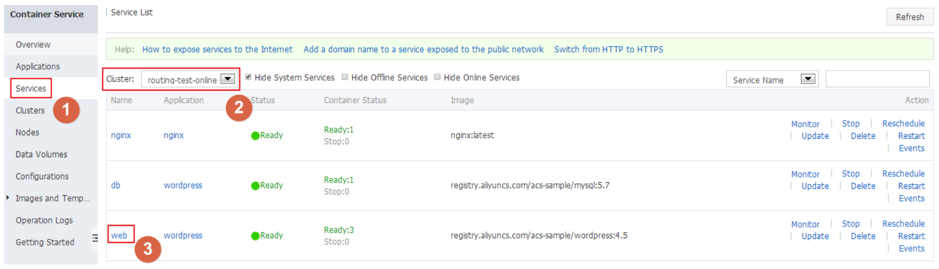
## View container logs

Log on to the Container Service console.

Click **Services** in the left navigation pane.

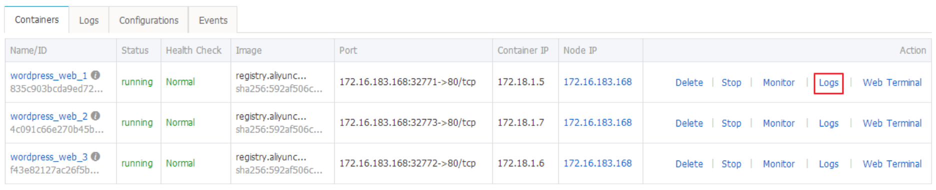
Select the cluster of the desired service.

Click the name of the target service.



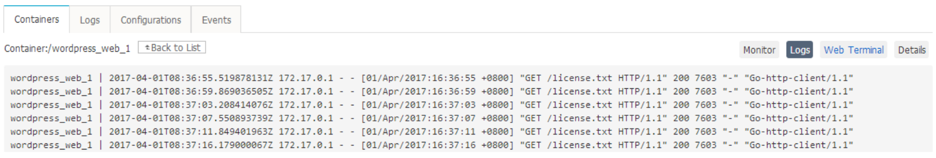
| Name | Application | Status | Container Status | Image | Action |
|---|---|---|---|---|---|
| nginx | nginx | ● Ready | Ready:1 Stop:0 | nginx:latest | Monitor \| Stop \| Reschedule \| Update \| Delete \| Restart \| Events |
| db | wordpress | ● Ready | Ready:1 Stop:0 | registry.aliyuncs.com/acs-sample/mysql:5.7 | Monitor \| Stop \| Reschedule \| Update \| Delete \| Restart \| Events |
| web | wordpress | ● Ready | Ready:3 Stop:0 | registry.aliyuncs.com/acs-sample/wordpress:4.5 | Monitor \| Stop \| Reschedule \| Update \| Delete \| Restart \| Events |

Locate a container and click **Logs**.



| Name/ID | Status | Health Check | Image | Port | Container IP | Node IP | Action |
|---|---|---|---|---|---|---|---|
| wordpress_web_1 835c903bcda9ed72... | running | Normal | registry.aliyunc... sha256:592af506c... | 172.16.183.168:32771->80/tcp | 172.18.1.5 | 172.16.183.168 | Delete \| Stop \| Monitor \| Logs \| Web Terminal |
| wordpress_web_2 4c091c66e270b45b... | running | Normal | registry.aliyunc... sha256:592af506c... | 172.16.183.168:32773->80/tcp | 172.18.1.7 | 172.16.183.168 | Delete \| Stop \| Monitor \| Logs \| Web Terminal |
| wordpress_web_3 f43e82127ac26f5b... | running | Normal | registry.aliyunc... sha256:592af506c... | 172.16.183.168:32772->80/tcp | 172.18.1.6 | 172.16.183.168 | Delete \| Stop \| Monitor \| Logs \| Web Terminal |

You can view the logs of this container.



```
wordpress_web_1 | 2017-04-01T08:36:55.519878131Z 172.17.0.1 - - [01/Apr/2017:16:36:55 +0800] "GET /license.txt HTTP/1.1" 200 7603 "-" "Go-http-client/1.1"
wordpress_web_1 | 2017-04-01T08:36:59.869036505Z 172.17.0.1 - - [01/Apr/2017:16:36:59 +0800] "GET /license.txt HTTP/1.1" 200 7603 "-" "Go-http-client/1.1"
wordpress_web_1 | 2017-04-01T08:37:03.208414076Z 172.17.0.1 - - [01/Apr/2017:16:37:03 +0800] "GET /license.txt HTTP/1.1" 200 7603 "-" "Go-http-client/1.1"
wordpress_web_1 | 2017-04-01T08:37:07.550893739Z 172.17.0.1 - - [01/Apr/2017:16:37:07 +0800] "GET /license.txt HTTP/1.1" 200 7603 "-" "Go-http-client/1.1"
wordpress_web_1 | 2017-04-01T08:37:11.849401963Z 172.17.0.1 - - [01/Apr/2017:16:37:11 +0800] "GET /license.txt HTTP/1.1" 200 7603 "-" "Go-http-client/1.1"
wordpress_web_1 | 2017-04-01T08:37:16.179000067Z 172.17.0.1 - - [01/Apr/2017:16:37:16 +0800] "GET /license.txt HTTP/1.1" 200 7603 "-" "Go-http-client/1.1"
```

# Monitoring

# Container monitoring service

Container monitoring service depends on Alibaba Cloud CloudMonitor service, and provides users with default monitoring, alarm rules configuration and other services for operating and maintaining the container. Moreover, the Container Service provides the capability to integrate with third-party open source solutions (for more information, refer to Integrate with third-party monitoring solutions).

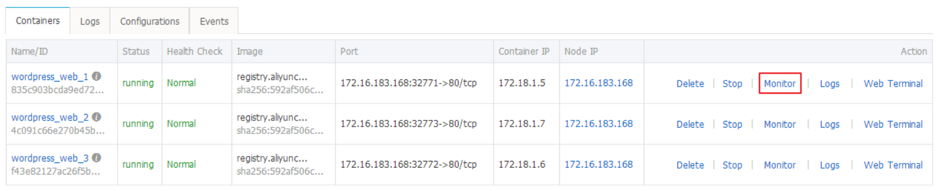
## Operating procedure

Log on to the Container Service console.

Use one of the following three methods to locate the container list page:

- Through nodes.
  i. Click **Nodes** in the left navigation pane.
  ii. Click the IP address of the node.
- Through applications.
  i. Click **Applications** in the left navigation pane.
  ii. Select the cluster of the desired application.
  iii. Click the name of the target application.
- Through services.
  i. Click **Services** in the left navigation pane.
  ii. Select the cluster of the desired service.
  iii. Click the name of the target service.

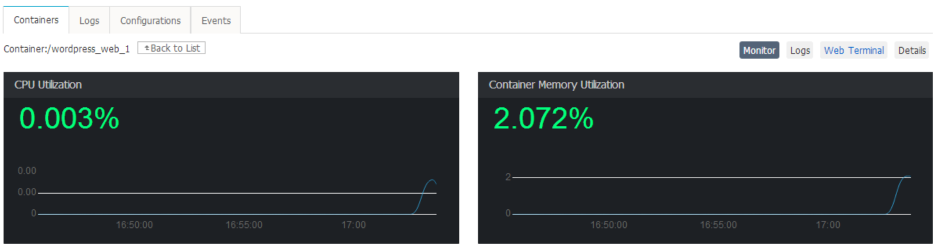
In the container list, locate the target container and click **Monitor**.



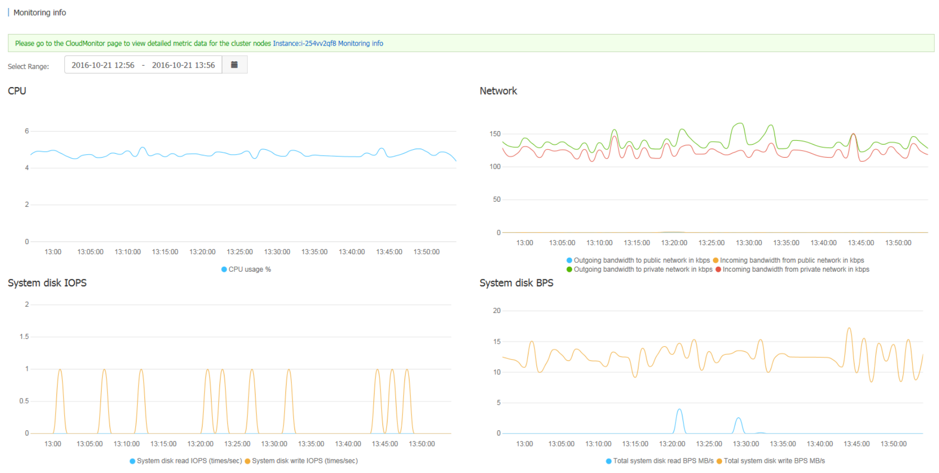
Containers | Logs | Configurations | Events

| Name/ID | Status | Health Check | Image | Port | Container IP | Node IP | Action |
|---|---|---|---|---|---|---|---|
| wordpress_web_1 835c903bcda9ed72... | running | Normal | registry.aliyunc... sha256:592af506c... | 172.16.183.168:32771->80/tcp | 172.18.1.5 | 172.16.183.168 | Delete \| Stop \| Monitor \| Logs \| Web Terminal |
| wordpress_web_2 4c091c66e270b45b... | running | Normal | registry.aliyunc... sha256:592af506c... | 172.16.183.168:32773->80/tcp | 172.18.1.7 | 172.16.183.168 | Delete \| Stop \| Monitor \| Logs \| Web Terminal |
| wordpress_web_3 f43e82127ac26f5b... | running | Normal | registry.aliyunc... sha256:592af506c... | 172.16.183.168:32772->80/tcp | 172.18.1.6 | 172.16.183.168 | Delete \| Stop \| Monitor \| Logs \| Web Terminal |

You can view the real-time monitoring information of the container.



Click **View History Monitoring Data/Set Alarm Rules** to go to the CloudMonitor console and view history monitoring data of this container.


Monitoring info

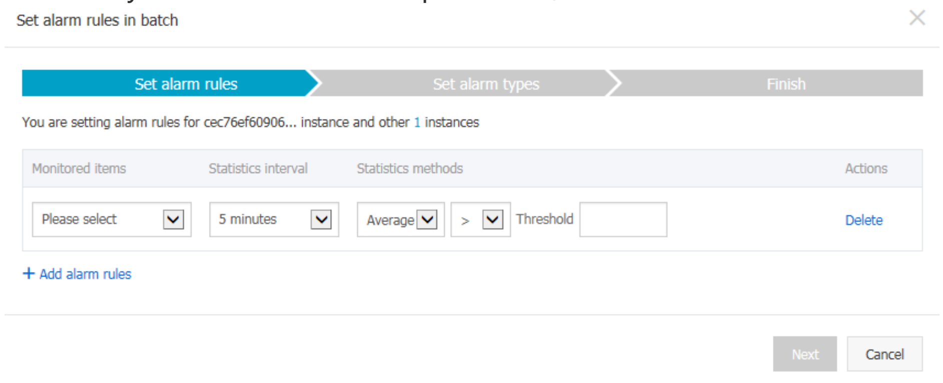
Please go to the CloudMonitor page to view detailed metric data for the cluster nodes Instance:i-254vv2qf8 Monitoring info

Select Range: 2016-10-21 12:56 - 2016-10-21 13:56

Set alarm rules.

For some services, you can add alarm rules according to your needs, and the container monitoring service will send a message to the contact listed in the cloud account whenever the monitoring metrics breach the alarm threshold value.

i. Click **New Alarm Rule** in the upper-right corner.
ii. Based on your actual business requirements, define the rules and click **Next**.


Set alarm rules in batch

You are setting alarm rules for cec76ef60906... instance and other 1 instances

| Monitored items | Statistics interval | Statistics methods | | | Actions |
|---|---|---|---|---|---|
| Please select | 5 minutes | Average | > | Threshold | Delete |

iii. Set the contact for alarms and click **OK**.
iv. A message is displayed indicating that the alarm rule has been set. Click **Close**.
v. Click **Alarm rules** to view alarm rules and relevant records.

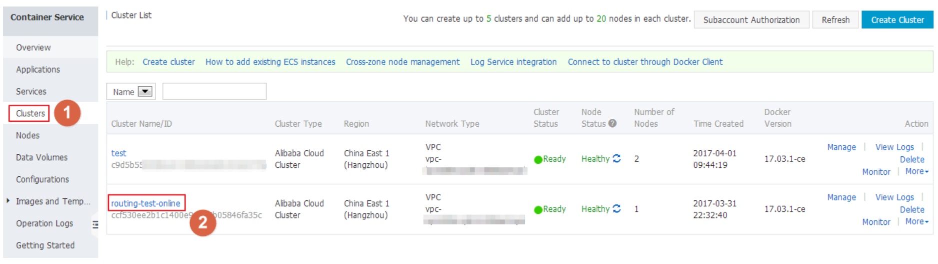
# View monitoring information

## View server monitoring information

Log on to the Container Service console.

Click **Clusters** in the left navigation pane.

Click the name of the target cluster.

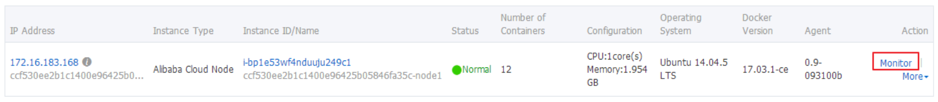


You can create up to 5 clusters and can add up to 20 nodes in each cluster.

Help: Create cluster How to add existing ECS instances Cross-zone node management Log Service integration Connect to cluster through Docker Client

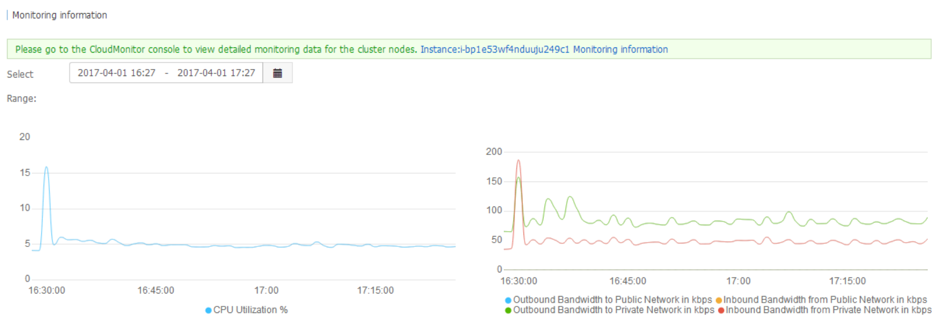
| Cluster Name/ID | Cluster Type | Region | Network Type | Cluster Status | Node Status | Number of Nodes | Time Created | Docker Version | Action |
|---|---|---|---|---|---|---|---|---|---|
| test c9d5b55[redacted] | Alibaba Cloud Cluster | China East 1 (Hangzhou) | VPC vpc-[redacted] | ● Ready | Healthy | 2 | 2017-04-01 09:44:19 | 17.03.1-ce | Manage \| View Logs \| Delete Monitor \| More |
| routing-test-online ccf530ee2b1c1400e9[illegible]b05846fa35c | Alibaba Cloud Cluster | China East 1 (Hangzhou) | VPC vpc-[redacted] | ● Ready | Healthy | 1 | 2017-03-31 22:32:40 | 17.03.1-ce | Manage \| View Logs \| Delete Monitor \| More |

Locate the target node and Click **Monitor**.



| IP Address | Instance Type | Instance ID/Name | Status | Number of Containers | Configuration | Operating System | Docker Version | Agent | Action |
|---|---|---|---|---|---|---|---|---|---|
| 172.16.183.168 ccf530ee2b1c1400e96425b0... | Alibaba Cloud Node | i-bp1e53wf4nduuju249c1 ccf530ee2b1c1400e96425b05846fa35c-node1 | ● Normal | 12 | CPU:1core(s) Memory:1.954 GB | Ubuntu 14.04.5 LTS | 17.03.1-ce | 0.9-093100b | Monitor More |

You can view the node monitoring information.


Monitoring information
Select
Range:
2017-04-01 16:27 - 2017-04-01 17:27
20
15
10
5
0
16:30:00
16:45:00
17:00
17:15:00
● CPU Utilization %
200
150
100
50
0
● Outbound Bandwidth to Public Network in kbps ● Inbound Bandwidth from Public Network in kbps
● Outbound Bandwidth to Private Network in kbps ● Inbound Bandwidth from Private Network in kbps

Please go to the CloudMonitor console to view detailed monitoring data for the cluster nodes. Instance:i-bp1e53wf4nduuju249c1 Monitoring information

## View container monitoring information

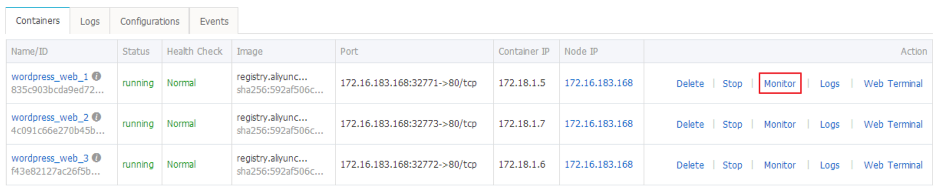
1. Log on to the Container Service console.

   Use one of the following three methods to locate the container list page:

   - Through nodes.
     i. Click **Nodes** in the left navigation pane.
     ii. Click the IP address of the node.
   - Through applications.
     i. Click **Applications** in the left navigation pane.
     ii. Select the cluster of the desired application.
     iii. Click the name of the target application.
   - Through services.
     i. Click **Services** in the left navigation pane.

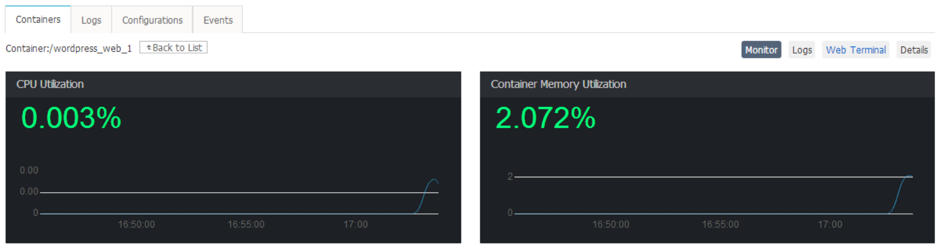
ii. Select the cluster of the desired service.
iii. Click the name of the target service.

In the container list, locate the target container and click **Monitor**.



| Name/ID | Status | Health Check | Image | Port | Container IP | Node IP | Action |
|---|---|---|---|---|---|---|---|
| wordpress_web_1 835c903bcda9ed72... | running | Normal | registry.aliyunc... sha256:592af506c... | 172.16.183.168:32771->80/tcp | 172.18.1.5 | 172.16.183.168 | Delete \| Stop \| Monitor \| Logs \| Web Terminal |
| wordpress_web_2 4c091c66e270b45b... | running | Normal | registry.aliyunc... sha256:592af506c... | 172.16.183.168:32773->80/tcp | 172.18.1.7 | 172.16.183.168 | Delete \| Stop \| Monitor \| Logs \| Web Terminal |
| wordpress_web_3 f43e82127ac26f5b... | running | Normal | registry.aliyunc... sha256:592af506c... | 172.16.183.168:32772->80/tcp | 172.18.1.6 | 172.16.183.168 | Delete \| Stop \| Monitor \| Logs \| Web Terminal |

You can view the container monitoring information.



# Custom monitoring

The container monitoring service integrates with the Alibaba CloudMonitor service and provides monitoring and alarm services for containers, applications, clusters, and nodes. The container monitoring service can meet the basic requirements for container monitoring. However, in many business scenarios, you may have to use customized monitoring capabilities for your systems and applications. In addition to the basic monitoring capabilities, the container monitoring service offers two custom monitoring modes, enabling you to submit your custom monitoring data by writing your own data collection script or exposing your HTTP monitoring data interface. The monitoring framework of the Container Service collects data once per minute by executing the script or calling the HTTP interface.

## Prerequisites

Before using the custom monitoring feature, you must integrate the container monitoring service with a third-party monitoring solution (for details, refer to Integrate with third-party monitoring solutions).

> **Note:** The container monitoring service only supports integration with InfluxDB and Prometheus by default.

Your custom monitoring data is submitted to your InfluxDB or Prometheus, and then displayed and analyzed.

## Submit monitoring data using a custom monitoring script

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.

```
weather,location=us-midwest temperature=82 1465839830100400200
| -------------------- -------------- |
| | | |
| | | |
+-----------+--------+-+---------+-+---------+
|measurement|,tag_set| |field_set| |timestamp|
+-----------+--------+-+---------+-+---------+
```

For details about the data format protocol, refer to InfluxDB Line Protocol.

Log on to the Container Service console. Use an orchestration template to create an application. Then 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"
```

This label defines the command in the application container that should be executed to collect monitoring data. The label is configured as below:

```
labels:
aliyun.monitoring.script: "command used to execute the script"
```

Open the InfluxDB Web-based management interface to view the database tables named after data indexes.

For information about how to view the database tables, refer to Integrate with third-party monitoring solutions.

# Collect data using the custom HTTP monitoring data interface

Create a Docker image and expose the HTTP interface.

This interface outputs the monitoring data. You can define the monitoring data format as long as it conforms 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. You need to use another configuration item to specify what type of JSON data has the tag attribute. For details, refer to Telegraf JSON data format.

Log on to the Container Service console and use an orchestration template to create an application. In the template, you must 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 the 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 shown as below:

```
{
"tag1": "tag1value",
"tag2": "tag2value",
"field1": 1,
"field2": 2,
"field3": true,
"field4": 1.5
}
```

The aliyun.monitoring.tags: "tag1,tag2" label defines that in the submitted JSON data, tag1 and tag2 are the tags of the submitted data.

Open the InfluxDB Web-based management interface 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 should be httpjson_nodejsapp_nodejsapp_1.

For details about how to view the database tables, refer to Integrate with third-party monitoring solutions.

# Integrate with third-party monitoring solutions

Alibaba Cloud Container Service provides the capability to integrate with third-party monitoring solutions.

**Note:** By default, only integration with InfluxDB and Prometheus is supported.

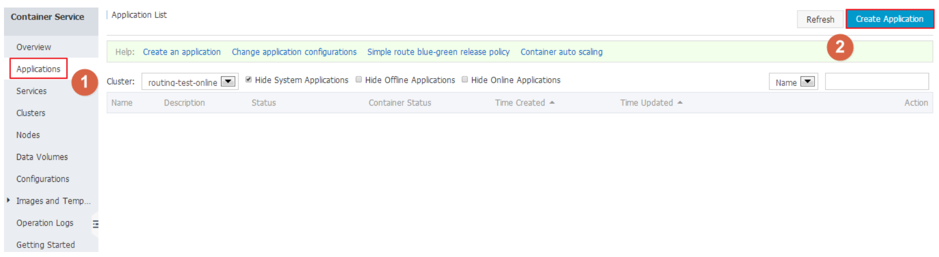
The following example introduces how to integrate the Container Service monitoring service with third-party monitoring solutions by taking InfluxDB as an example.

## Operating procedure

Log on to the Container Service console.

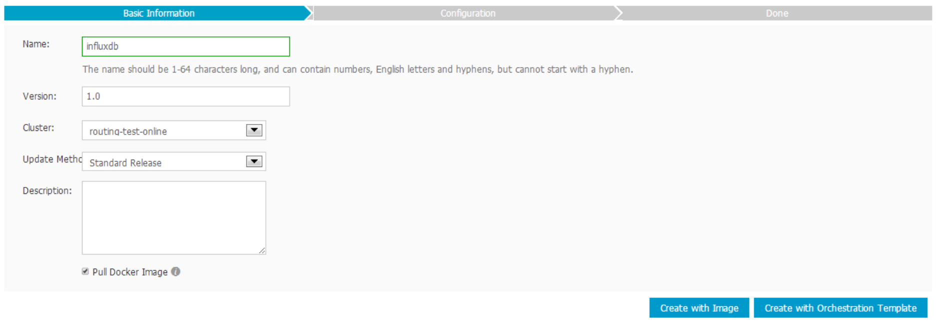
Click **Applications** in the left navigation pane.

Click **Create Application** in the upper-right corner.


Container Service
Overview
Applications
Services
Clusters
Nodes
Data Volumes
Configurations
Images and Temp...
Operation Logs
Getting Started
1
Application List
Refresh
Create Application
2
Help: Create an application Change application configurations Simple route blue-green release policy Container auto scaling
Cluster: routing-test-online
Hide System Applications
Hide Offline Applications
Hide Online Applications
Name
Name Description Status Container Status Time Created Time Updated Action

Enter the basic information of the application and click **Create with Orchestration Template**.

In this example, the name of the application is **influxdb**.



Name: influxdb

The name should be 1-64 characters long, and can contain numbers, English letters and hyphens, but cannot start with a hyphen.

Version: 1.0

Cluster: routing-test-online

Update Method: Standard Release

Description:

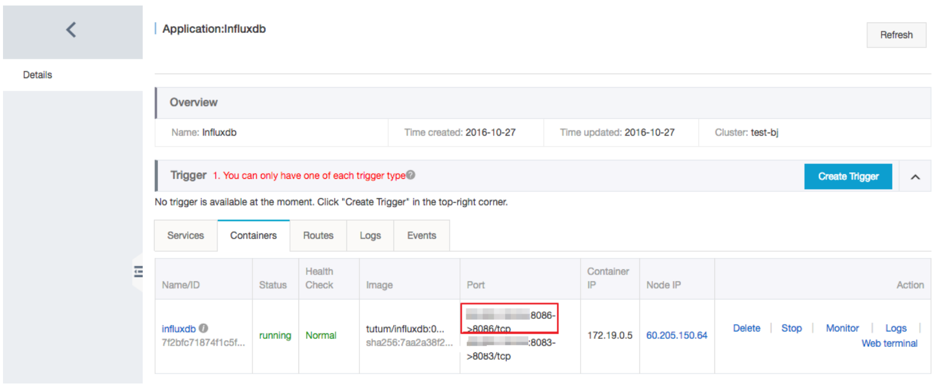
✓ Pull Docker Image

Enter the following template and click **Create and Deploy**.

> **Note:** In a real production environment, the template needs to be modified. The influxdb service definition should not expose the port to the host.

```
version: '2'
services:
#Define influxdb
influxdb:
image: tutum/influxdb:0.9
ports:
- "8083:8083" #Expose Web interface port
- "8086:8086" #Expose data API Web interface port
```

After the application is successfully created, click the name of the application **influxdb** in the **Application List** page to view the application details. Click **Containers** to view the node IP and the exposed port of **influxdb** and copy the values (in this example, copy the node IP and port number of 8086).



Application:Influxdb

Overview

Name: Influxdb | Time created: 2016-10-27 | Time updated: 2016-10-27 | Cluster: test-bj

Trigger 1. You can only have one of each trigger type

No trigger is available at the moment. Click "Create Trigger" in the top-right corner.

| Name/ID | Status | Health Check | Image | Port | Container IP | Node IP | Action |
|---|---|---|---|---|---|---|---|
| influxdb 7f2bfc71874f1c5f... | running | Normal | tutum/influxdb:0... sha256:7aa2a38f2... | [redacted]8086->8086/tcp [redacted]:8083->8083/tcp | 172.19.0.5 | 60.205.150.64 | Delete \| Stop \| Monitor \| Logs \| Web terminal |

Click **Applications** in the left navigation pane to return to the **Application List** page.

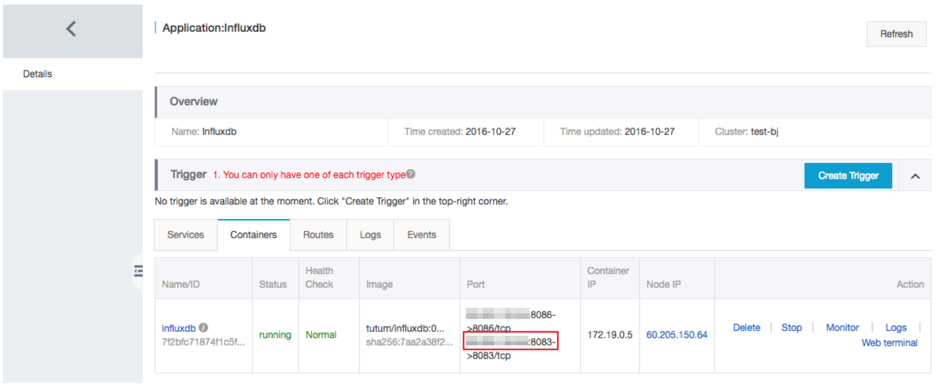
Click **Update**, add the following contents in the template to declare the integration of InfluxDB and the container monitoring service and click **OK**.

**Note:** By default, only integration with InfluxDB and Prometheus is supported. The labels for InfluxDB and Prometheus integration are aliyun.monitoring.addon.influxdb and aliyun.monitoring.addon.prometheus respectively. The label value must be in the format of schema:hostIp:port.

```
labels
aliyun.monitoring.addon.influxdb:"http://node IP : port"
```

As the container monitoring service Agent adopts host network mode, the Container Service cannot use link to identify InfluxDB. You need to first create the **influxdb** application and then add the data report address exposed by the application in the application label to inform the data acquisition client. After that, the monitoring service automatically writes the collected running status data of containers to influxdb.

In the **Application List** page, click the application name **influxdb** and click **Containers**. Copy the **influxdb** port that has been exposed externally.



Application:Influxdb

Overview

Name: Influxdb | Time created: 2016-10-27 | Time updated: 2016-10-27 | Cluster: test-bj

Trigger 1. You can only have one of each trigger type

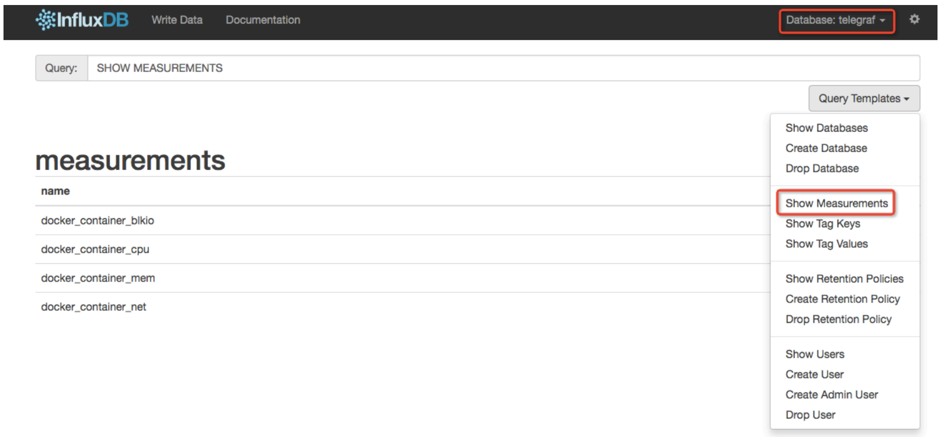
No trigger is available at the moment. Click "Create Trigger" in the top-right corner.

| Name/ID | Status | Health Check | Image | Port | Container IP | Node IP | Action |
|---|---|---|---|---|---|---|---|
| influxdb 7f2bfc71874f1c5f... | running | Normal | tutum/influxdb:0... sha256:7aa2a38f2... | [redacted]8086->8086/tcp [redacted]:8083->8083/tcp | 172.19.0.5 | 60.205.150.64 | Delete \| Stop \| Monitor \| Logs \| Web terminal |

Access the InfluxDB management page in the browser to view the metric data written by the container monitoring service.

i. Select **telegraf**.
ii. Click **Query Templates** and click **Show Measurements** in the drop-down menu.
iii. Press Enter.

You can view the database table, as shown in the figure below.

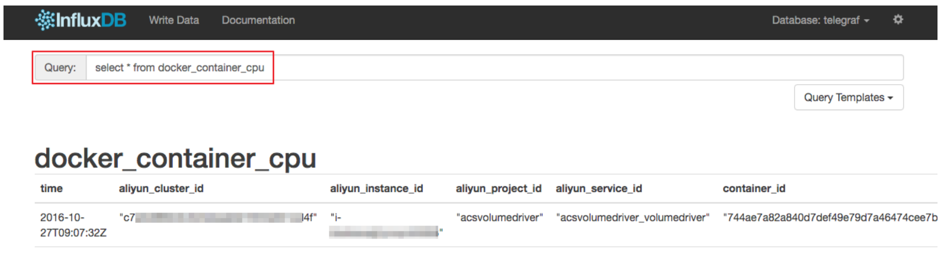


Query: SHOW MEASUREMENTS

measurements

| name |
|---|
| docker_container_blkio |
| docker_container_cpu |
| docker_container_mem |
| docker_container_net |

View detailed data in a table.



Query: select * from docker_container_cpu

docker_container_cpu

| time | aliyun_cluster_id | aliyun_instance_id | aliyun_project_id | aliyun_service_id | container_id |
|---|---|---|---|---|---|
| 2016-10-27T09:07:32Z | "c7[redacted]4f" | "i-[redacted]" | "acsvolumedriver" | "acsvolumedriver_volumedriver" | "744ae7a82a840d7def49e79d7a46474cee7b |

## What to do next

After the 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.

# Container auto scaling

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

You can configure container auto scaling rules when creating applications or add container auto scaling rules for existing applications by updating application configurations.

> **Note:** To use the container auto scaling function, you need to first upgrade the cluster Agent to the latest version. For detailed information about how to upgrade Agent, refer to Upgrade Agent
> .

**Auto scaling policies:**

- When the metric value exceeds the upper limit, the Container Service increases the number

of containers at the step defined by users.
- When the metric value is lower than the lower limit, the Container Service reduces the number of containers at the step defined by users.

**Service metrics:**

- Average CPU usage
- Average memory usage

**Note:** When judging whether the monitoring metric exceeds the specified upper limit or lower limit, the Container Service uses the average value of the monitoring metric (namely, the average CPU usage and the average memory usage) within a sample period (one minute) and it will only trigger container scaling when the average monitoring metrics of three consecutive sample periods all exceed the specified upper limit or lower limit, in order to avoid container scaling caused by monitoring data jitter.

## Set the container auto scaling

You can set the container auto scaling using one of the following three methods.

### Use image to create the application

While creating the application, click **Create with Image**.

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

In the **Deploy** area at the bottom of the page, select **Enable** for **Auto Scaling**, and set automatic scaling parameters.

**Constraints:**

- 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 of the application is less than or equal to the **Min Number of Containers**, contraction will not be performed. For expansion, if the number of containers of the application is greater than or equal to the **Max Number of Containers**, expansion will not be performed.

**Note:** Set scaling policies with due care. If the application already meets the specified scaling rules when you set the scaling rules and the application will still meet the scaling rules after container scaling, the monitor will continuously trigger container scaling.

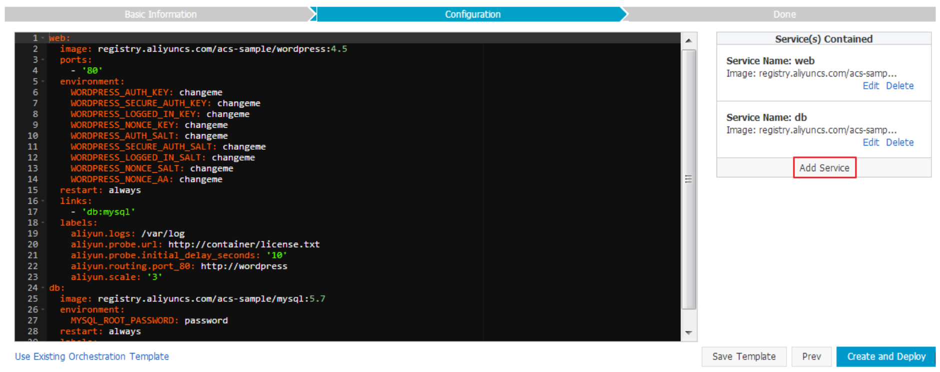
## Use orchestration template to create the application

While creating the application, click **Create with Orchestration Template**.

Select to **Use Existing Orchestration Template** or use your own orchestration template.

Set container auto scaling rules.

Use **Add Service**.



```
web:
  image: registry.aliyuncs.com/acs-sample/wordpress:4.5
  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
    WORDPRESS_NONCE_AA: changeme
  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: '3'
db:
  image: registry.aliyuncs.com/acs-sample/mysql:5.7
  environment:
    MYSQL_ROOT_PASSWORD: password
  restart: always
```

Service(s) Contained

Service Name: web
Image: registry.aliyuncs.com/acs-samp...

Service Name: db
Image: registry.aliyuncs.com/acs-samp...

In the pop-up dialog box, click **More Settings**, select **Enable** for auto scaling, and set auto scaling parameters.

Set container auto scaling rules in the template manually.

In the labels configuration in the application template, add corresponding labels:

- Specify the step size (the default value is 1): aliyun.auto_scaling.step
- Specify the maximum number of instances (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

## Update application configurations

You can add container auto scaling rules to existing applications by updating application configurations.

In the **Application List** page, select the desired application and click **Update** on the right.

In the labels configuration in **Template**, add the corresponding container auto scaling labels.

# Node auto scaling

To meet the needs of applications under different loads, in addition to container elastic scaling, the Container Service also provides node auto scaling. That is, you can configure the Container Service to automatically adjust the number of nodes according to the node resource usage monitoring information.

**Node scaling policies:**

When the monitoring metrics exceed the specified expansion conditions, the Container Service increases the number of nodes at the specified expansion step.

When the condition metrics are lower than the specified contraction conditions, the Container Service reduces the number of nodes at the default contraction step.

**Auto scaling monitoring metrics：**

Cluster CPU average usage

Cluster memory average usage

## Prerequisites

Upgrade the cluster Agent to the latest version. Refer to Upgrade Agent.

Upgrade the cluster monitoring service (acsmonitoring) to the latest version. Refer to Upgrade system services.

Activate the RAM service and update the access authorization information (Operation path: on the Container Service console, click **Clusters** in the left navigation pane > select the desired cluster > click **More** > **Update RAM Authorization Information**).

## Instructions

When judging whether the monitoring metric exceeds the specified upper limit or lower limit, the Container Service uses the average value of the monitoring metric (namely, the average CPU usage and the average memory usage) within a sample period (one minute) and it will only trigger container scaling when the average monitoring metrics of three consecutive sample period all exceed the specified upper limit or lower limit, so as to avoid container scaling caused by monitoring data jitter.

Node contraction only contracts nodes that are added through node expansion; nodes created or added manually by users will not be affected. If you want to perform auto scaling on those nodes added manually, you need to add the following label for those nodes.

aliyun.reschedule==true

During node contraction, the system will delete the ECS instances in the cluster; therefore, back up data before node scaling.

Do not schedule stateful service onto nodes that enable node contraction. For related information, refer to constraint in Docker Compose.

When new ECS instances are expanded, the deployed containers will not be affected. The Container Service will deploy new containers according to the container deployment settings.

During node contraction, the Container Service will migrate containers on the delete ECS instances onto other ECS instances.

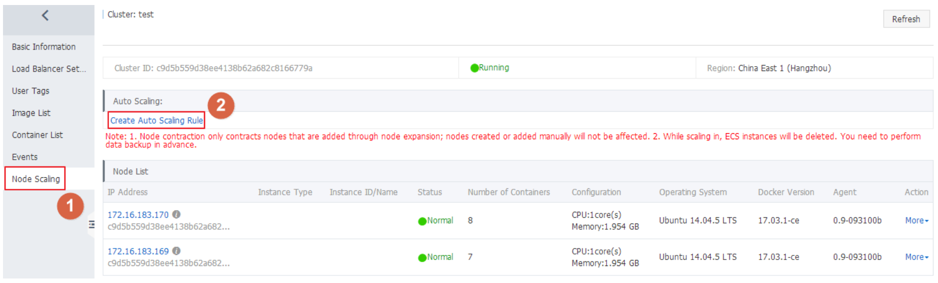
## Create a node scaling rule

Log on to the Container Service console.

Click **Clusters** in the left navigation pane.

Locate the target cluster and click **Manage**.

Click **Node Scaling** in the left navigation pane and click **Create Auto Scaling Rule**.



Cluster: test

Cluster ID: c9d5b559d38ee4138b62a682c8166779a | Running | Region: China East 1 (Hangzhou)

Auto Scaling:

Create Auto Scaling Rule

Note: 1. Node contraction only contracts nodes that are added through node expansion; nodes created or added manually will not be affected. 2. While scaling in, ECS instances will be deleted. You need to perform data backup in advance.

Node List

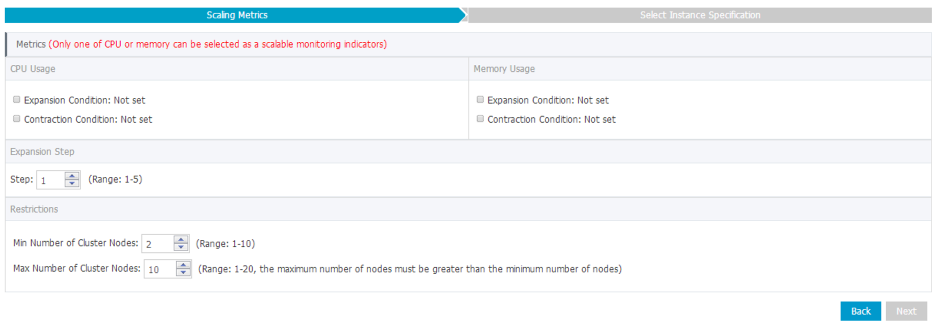
| IP Address | Instance Type | Instance ID/Name | Status | Number of Containers | Configuration | Operating System | Docker Version | Agent | Action |
|---|---|---|---|---|---|---|---|---|---|
| 172.16.183.170 c9d5b559d38ee4138b62a682... | | | Normal | 8 | CPU:1core(s) Memory:1.954 GB | Ubuntu 14.04.5 LTS | 17.03.1-ce | 0.9-093100b | More |
| 172.16.183.169 c9d5b559d38ee4138b62a682... | | | Normal | 7 | CPU:1core(s) Memory:1.954 GB | Ubuntu 14.04.5 LTS | 17.03.1-ce | 0.9-093100b | More |

Configure the scaling rules and click **Next**.

**Constraints:**

- 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, the default value of the **Contraction Step** is 1 and you cannot modify it.
- Set the **Min Number of Cluster Nodes** and **Max Number of Cluster Nodes**. For node contraction, when the number of nodes of the cluster is less than or equal to the **Min Number of Cluster Nodes**, node contraction will not be performed. For node expansion, when the number of nodes of the cluster is greater than or equal to the **Max Number of Cluster Nodes**, node expansion will not be performed.

**Note:** Set scaling policies with due care. If the cluster already meets the specified scaling rules when you set the scaling rules and the cluster will still meet the scaling rules after node scaling, the monitor will trigger node scaling continuously.



Metrics (Only one of CPU or memory can be selected as a scalable monitoring indicators)

CPU Usage

Expansion Condition: Not set

Contraction Condition: Not set

Memory Usage

Expansion Condition: Not set

Contraction Condition: Not set

Expansion Step

Step: 1 (Range: 1-5)

Restrictions

Min Number of Cluster Nodes: 2 (Range: 1-10)

Max Number of Cluster Nodes: 10 (Range: 1-20, the maximum number of nodes must be greater than the minimum number of nodes)

Configure the instance specifications and click **Submit**.

During expansion, the Container Service adds nodes into the cluster according to the specifications configure here. For more information on instance specifications, refer to Create a cluster.

In addition, you can set to add the IP addresses of the newly added ECS instances to RDS instance white lists, so that the ECS instances can access the RDS instances.

**Note:** The ECS instances must be in the same region as the RDS instances.

## View the created node scaling rule

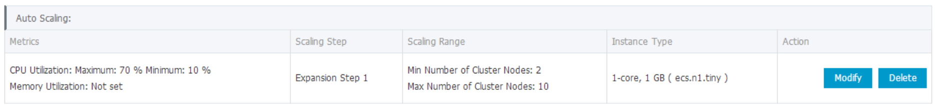
Log on to the Container Service console.

Click **Clusters** in the left navigation pane.

Locate the target cluster and click **Manage**.

Click **Node Scaling** in the left navigation pane.

You can view the created node scaling rule.



Auto Scaling:

| Metrics | Scaling Step | Scaling Range | Instance Type | Action |
|---|---|---|---|---|
| CPU Utilization: Maximum: 70 % Minimum: 10 %<br>Memory Utilization: Not set | Expansion Step 1 | Min Number of Cluster Nodes: 2<br>Max Number of Cluster Nodes: 10 | 1-core, 1 GB ( ecs.n1.tiny ) | Modify Delete |

You can click **Modify** to modify the node scaling rule or click **Delete** to delete this rule.

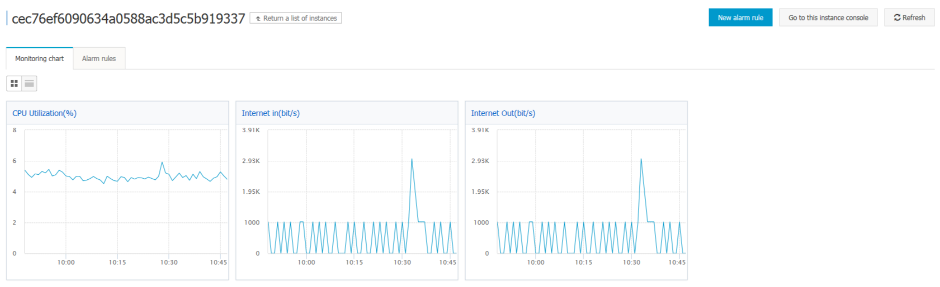
## View monitoring metrics

Click **Clusters** in the left navigation pane.

Locate the target cluster and click **Monitor**.

The interface switches to the CloudMonitor console and you can see the cluster monitoring information.

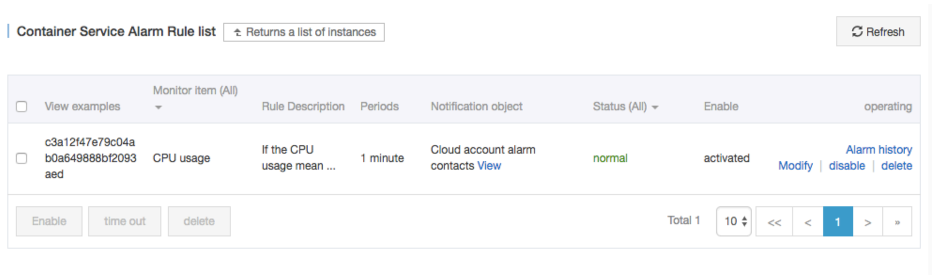
**Note:** If no monitoring data is displayed, check whether the monitoring service (acsmonitoring) is correctly installed, whether the cluster Agent is the latest version, and whether the monitoring service (acsmonitoring) is the latest version; if not, perform upgrade.



Click **Container Service** in the left navigation pane to view the cluster list.

Click **View all rules** to view the alarm rules of auto scaling.

> **Note:** If no alarm rule is displayed, update the access authorization information of the cluster (in the Cluster List page, select the desired cluster, click More > Update RAM Authorization Information). Activate the RAM service before update the authorization information; otherwise, error message will be displayed.



Container Service Alarm Rule list | Returns a list of instances | Refresh

| | View examples | Monitor item (All) | Rule Description | Periods | Notification object | Status (All) | Enable | operating |
|---|---|---|---|---|---|---|---|---|
| ☐ | c3a12f47e79c04a b0a649888bf2093 aed | CPU usage | If the CPU usage mean ... | 1 minute | Cloud account alarm contacts View | normal | activated | Alarm history Modify \| disable \| delete |

Enable | time out | delete — Total 1 | 10 | << | < | 1 | > | »

Select an alarm rule. You can then modify the alarm conditions, modify the contact and notice type (including message and email), or suspend the alarm rule.

# Authentication management

# Resource access management

When you create several clusters with the Container Service and multiple users in your organization need to use these clusters, if these users share the access key of your Alibaba cloud account, the following problems will arise:

- It has a high risk of information leakage.
- You cannot restrict the access right of users, which may cause security risks due to improper

operations.

Resource Access Management (RAM) is an Alibaba Cloud service designed for controlling resource access. Through RAM, you can manage your users and control users’ right to access the resources under your name in a centralized manner. For the Container Service, over 90% of users are granted cluster dimensions rights which are mainly used to restrict the addition or deletion of clusters. Therefore, in order to simplify access control, the Container Service provides two authentication policies: resource authentication and fine-grained API authentication. For general users, resource allocation authentication policy can meet most demands. For users who possess knowledge of RAM and want to perform user authentication at the API level, API authentication is available.

## Application scenario

In the following example, an app company want to use Alibaba Cloud service to provide APP backend service, infrastructure and middleware. Users of the Alibaba Cloud service will include backend developers, test engineers, operation & maintenance engineers, and basic platform engineers. In terms of responsibility assignment:

- The backend developers are responsible for developing APP backend service.
- The test engineers are responsible for testing APP backend service and maintenance of basic test environment.
- The operation & maintenance engineers are responsible for the operation and maintenance of all kinds of environment (scaling and monitoring of clusters).
- The basic platform engineers are responsible for the maintenance of middleware and construction of basic design, including GitLab, Jenkins, Sentry, and so on.

The different roles of these users mean different demands for resources. In order to ensure that each role has enough authority to complete its own tasks without violating others’ rights. In this scenario, resource allocation authentication can be implemented.

## Operating procedure

**Note:** When you enter the RAM console, you may find some RAM child accounts starting with acs_. These accounts are automatically created by the Container Service. Do not delete them.

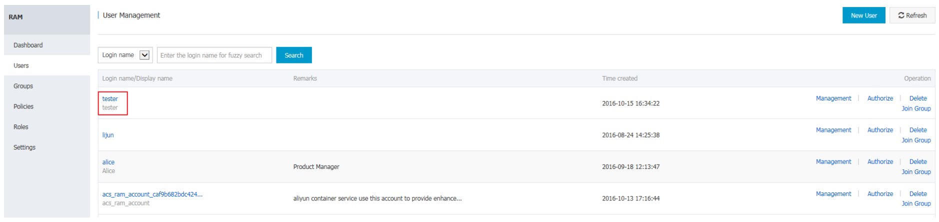
### Master account authorization

Log on to the RAM console.

Create a RAM subaccount.

Enable console login.

For details about how to create a subaccount, refer to the RAM documentation.



| Login name/Display name | Remarks | Time created | Operation |
|---|---|---|---|
| tester<br>tester | | 2016-10-15 16:34:22 | Management \| Authorize \| Delete<br>Join Group |
| lijun | | 2016-08-24 14:25:38 | Management \| Authorize \| Delete<br>Join Group |
| alice<br>Alice | Product Manager | 2016-09-18 12:13:47 | Management \| Authorize \| Delete<br>Join Group |
| acs_ram_account_caf9b682bdc424...<br>acs_ram_account | aliyun container service use this account to provide enhance... | 2016-10-13 17:16:44 | Management \| Authorize \| Delete<br>Join Group |

Log on to the Container Service console.

Click **Clusters** in the left navigation pane.

Click **Subaccount Authorization**.

Select the corresponding subaccount and click **Next**.

Select corresponding cluster resources and permission type. Click **Next** and click **Confirm** in the pop-up dialog box.

Click **Finish** to complete the authorization.

## Use the subaccount

Use a subaccount to log on (use a master account to log on to the RAM management console and inform the subusers of the logon address of the management console) and enter the Container Service console.

The assigned clusters are displayed in the cluster list.

# Understand authentication policy

Authentication policy is the verification of a user’ s right to use cloud resources, which is divided into resource allocation authentication and fine-grained API authentication.

Fine-grained API authentication is based on RAM (Resource Access Management) for authentication.

Resource allocation authentication is enveloped by the Container Service, and is an addition to fine-grained API authentication. It mainly facilitates resource allocation in the Container Service that cannot be accomplished through the fine-grained API authentication. At the same time, it encapsulates the fine-grained API authentication policies into read-only and read-write permissions. With the read-only permission, you can create and delete applications, but cannot perform cluster-level add, delete, modify and view operations. With the read-write permission, you can fully manage clusters.

## Advanced usage

Fine-grained API authentication can realize API-level authentication. At present, the following post authentication conditions are provided.

| Authentication Action name | Explanation |
|---|---|
| GetClusterById | Get cluster description. |
| GetClusterCerts | Get cluster certificates. |
| CreateCluster | Create cluster. |
| DeleteCluster | Delete cluster. |
| UpdateClusterSizeById | Expand cluster. |

Resource List:

| Resource | Explanation |
|---|---|
| cluster | Cluster |

Example:

```
{
"Statement": [
{
"Action": "cs:*",
"Effect": "Allow",
"Resource": "acs:cs:*:*:cluster/cc6b56877fd64407fb615dd09ff85303e"
},
{
"Action": "cs:*",
"Effect": "Allow",
"Resource": "acs:cs:*:*:cluster/cee52159dd72d4ead9c0ee1b1708b7065"
}
],
"Version": "1"
}
```

In the preceding example, grant all rights to clusters cc6b56877fd64407fb615dd09ff85303e and cee52159dd72d4ead9c0ee1b1708b7065.

For more information about fine-grained API authentication usage, refer to RAM documentation.

# DevOps

## Jenkins-based continuous delivery

As an important step in agile development, continuous integration is designed to maintain high quality while accelerating product iteration. Every time codes are updated, an automatic test is performed to test the codes and function validity. The codes can only be delivered and deployed after they pass the automatic test.

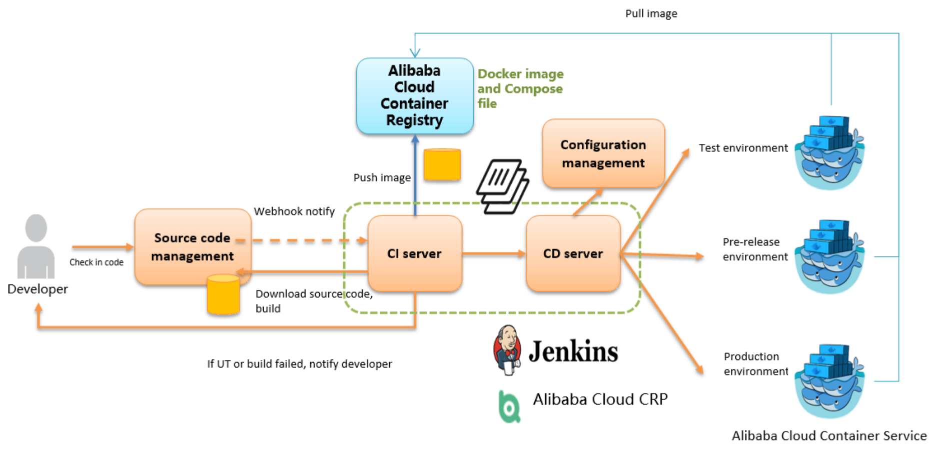
This section describes how to combine Jenkins, one of the most popular integration tools, with Alibaba Cloud Container Service to realize automatic test and image building push.

The following example demonstrates how to perform automatic test and build a Docker image through Alibaba Cloud Container Service, which will lead to 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 shown in the following figure.



Slave-nodejs is a slave node used for unit testing and for building and pushing the image.

### Jenkins introduction

Jenkins is an open-source continuous integration tool developed on Java. It monitors and triggers continuously-repeated work and supports expansion of multiple platforms and plugins. Jenkins is an open-source 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 through 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:** Jenkins job and projects mentioned here all refer to a build unit of Jenkins, that is, 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 execute the test and building. Through slave images and sample templates provided by Alibaba Cloud Container Service for different environments such as Python, nodejs and go, you can quickly and easily generate Jenkins applications and various slave nodes, configure node information in Jenkins applications, and designate execution nodes in the build projects, so as to implement the entire integration process.

**Note:** For images provided by Alibaba Cloud Container Service for developing slave nodes, refer to https://github.com/AliyunContainerService/jenkins-slaves.

### 1.1 Create a Jenkins orchestration template

Create a new template and create the orchestration based on the following content.

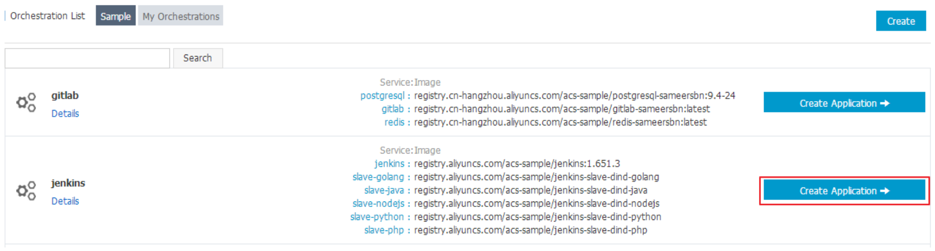
**Note:** For how to create an orchestration template, refer to Create an orcgestration template.

```
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'
```

### 2.1 Use the template to create Jenkins applications and slave nodes

You can also directly use a Jenkins sample template provided by Alibaba Cloud Container Service to create Jenkins applications and slave nodes.



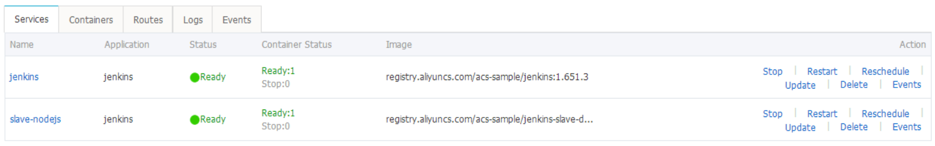
gitlab

Service:Image
postgresql : registry.cn-hangzhou.aliyuncs.com/acs-sample/postgresql-sameersbn:9.4-24
gitlab : registry.cn-hangzhou.aliyuncs.com/acs-sample/gitlab-sameersbn:latest
redis : registry.cn-hangzhou.aliyuncs.com/acs-sample/redis-sameersbn:latest

jenkins

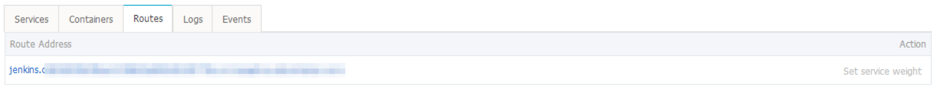
Service:Image
jenkins : registry.aliyuncs.com/acs-sample/jenkins:1.651.3
slave-golang : registry.aliyuncs.com/acs-sample/jenkins-slave-dind-golang
slave-java : registry.aliyuncs.com/acs-sample/jenkins-slave-dind-java
slave-nodejs : registry.aliyuncs.com/acs-sample/jenkins-slave-dind-nodejs
slave-python : registry.aliyuncs.com/acs-sample/jenkins-slave-dind-python
slave-php : registry.aliyuncs.com/acs-sample/jenkins-slave-dind-php

After a successful creation, Jenkins applications and slave nodes will be displayed in the service list.



| Name | Application | Status | Container Status | Image | Action |
|---|---|---|---|---|---|
| jenkins | jenkins | ● Ready | Ready:1 Stop:0 | registry.aliyuncs.com/acs-sample/jenkins:1.651.3 | Stop \| Restart \| Reschedule \| Update \| Delete \| Events |
| slave-nodejs | jenkins | ● Ready | Ready:1 Stop:0 | registry.aliyuncs.com/acs-sample/jenkins-slave-d... | Stop \| Restart \| Reschedule \| Update \| Delete \| Events |

After opening the access endpoint provided by the Container Service, the Jenkins application deployed can now be used.

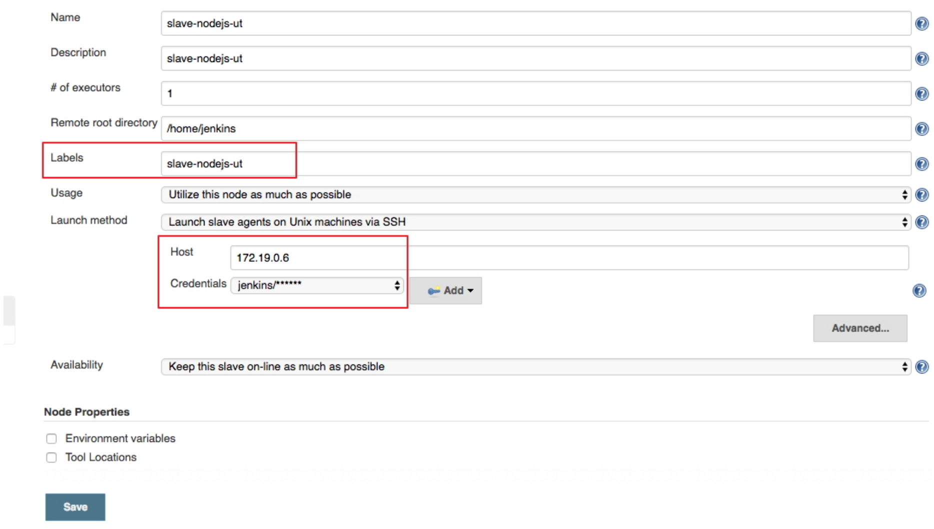


| Route Address | Action |
|---|---|
| jenkins.c[redacted] | Set service weight |

## Step 2 Realize automatic test and automatic build and push of image

### 2.1 Configure the slave container as the slave node of the Jenkins application

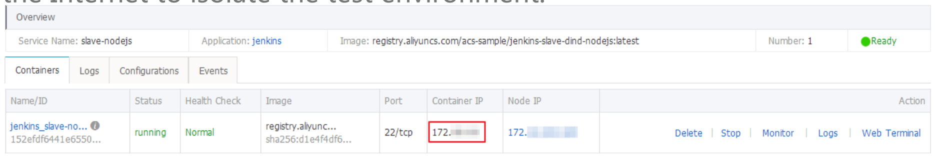
Open the Jenkins application and enter the **System Settings** interface. Select **Manage Node** > **Create Node**, and configure corresponding parameters.



| | |
|---|---|
| Name | slave-nodejs-ut |
| Description | slave-nodejs-ut |
| # of executors | 1 |
| Remote root directory | /home/jenkins |
| Labels | slave-nodejs-ut |
| Usage | Utilize this node as much as possible |
| Launch method | Launch slave agents on Unix machines via SSH |
| Host | 172.19.0.6 |
| Credentials | jenkins/****** |
| Availability | Keep this slave on-line as much as possible |

Note:

- Label is the only identifier of the slave.
- The slave container and Jenkins container run on the Alibaba Cloud platform at the same time. Therefore, you can fill in a container node IP address that is inaccessible to the Internet to isolate the test environment.



Overview

Service Name: slave-nodejs | Application: jenkins | Image: registry.aliyuncs.com/acs-sample/jenkins-slave-dind-nodejs:latest | Number: 1 | ●Ready

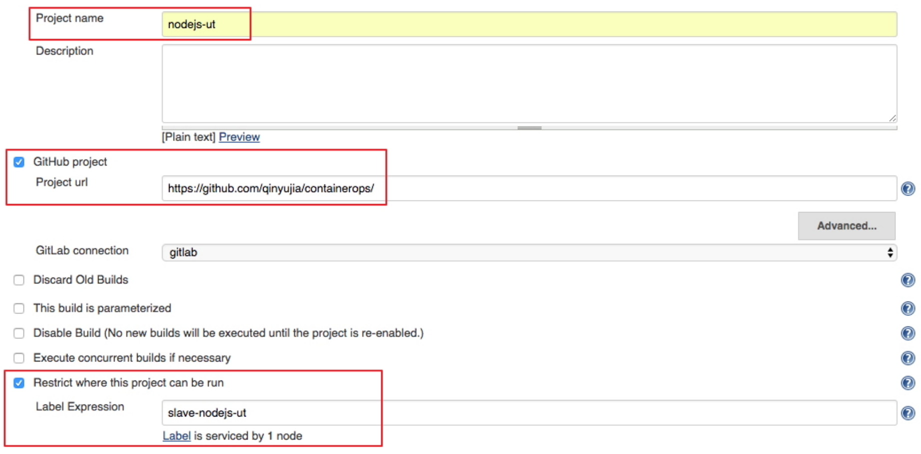
| Name/ID | Status | Health Check | Image | Port | Container IP | Node IP | Action |
|---|---|---|---|---|---|---|---|
| jenkins_slave-no... 152efdf6441e6550... | running | Normal | registry.aliyunc... sha256:d1e4f4df6... | 22/tcp | 172.[redacted] | 172.[redacted] | Delete \| Stop \| Monitor \| Logs \| Web Terminal |

- Use the jenkins account and password (the initial password is jenkins) in Dockerfile for the creation of the slave-nodejs image when adding Credential. Image Dockerfile address: https://github.com/AliyunContainerService/jenkins-slaves/tree/master/jenkins-slave-dind-nodejs.

## 2.2 Create a project to implement the automatic test.

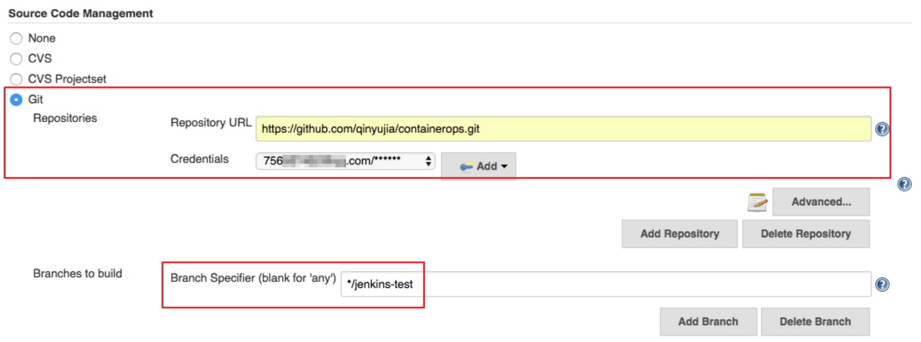
Create an item and choose to build a free style software project.

Enter the project name and select a node for running the project. In this example, enter the slave-nodejs-ut node created above.



| | |
|---|---|
| Project name | nodejs-ut |
| Description | |
| | [Plain text] Preview |
| ✓ GitHub project | |
| Project url | https://github.com/qinyujia/containerops/ |
| GitLab connection | gitlab |
| ☐ Discard Old Builds | |
| ☐ This build is parameterized | |
| ☐ Disable Build (No new builds will be executed until the project is re-enabled.) | |
| ☐ Execute concurrent builds if necessary | |
| ✓ Restrict where this project can be run | |
| Label Expression | slave-nodejs-ut |
| | Label is serviced by 1 node |

Configure the source code management and code branch. In this example, use GitHub to manage source codes.

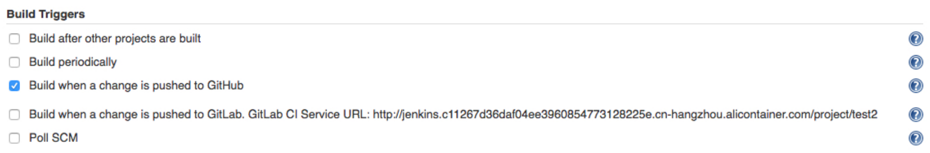


**Source Code Management**

- ○ None
- ○ CVS
- ○ CVS Projectset
- ● Git

| | | |
|---|---|---|
| Repositories | Repository URL | https://github.com/qinyujia/containerops.git |
| | Credentials | 756[redacted].com/****** |
| Branches to build | Branch Specifier (blank for 'any') | */jenkins-test |

Configure the trigger for building. In this example, automatically trigger project execution by combining GitHub Webhooks and services.



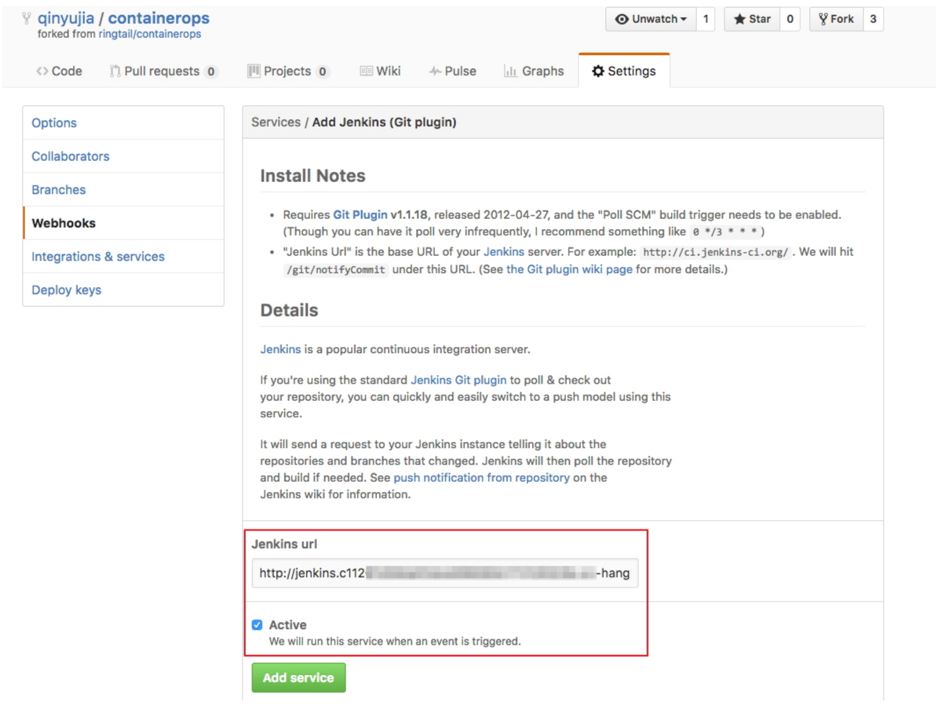
**Build Triggers**

- ☐ Build after other projects are built
- ☐ Build periodically
- ✓ Build when a change is pushed to GitHub
- ☐ Build when a change is pushed to GitLab. GitLab CI Service URL: http://jenkins.c11267d36daf04ee3960854773128225e.cn-hangzhou.alicontainer.com/project/test2
- ☐ Poll SCM

Add the Jenkins service hook to GitHub to implement automatic triggering.

Click the **Settings** tab on the GitHub project homepage, and click **Webhooks & services** > **Add service** and select **Jenkins (Git plugin)**. Enter **${Jenkins IP}/github-webhook/** in the Jenkins hook URL dialog box.

```
http://jenkins.cd***************.cn-beijing.alicontainer.com/github-webhook/
```



qinyujia / containerops
forked from ringtail/containerops

Services / Add Jenkins (Git plugin)

**Install Notes**

- Requires Git Plugin v1.1.18, released 2012-04-27, and the "Poll SCM" build trigger needs to be enabled. (Though you can have it poll very infrequently, I recommend something like `0 */3 * * *`)
- "Jenkins Url" is the base URL of your Jenkins server. For example: `http://ci.jenkins-ci.org/`. We will hit `/git/notifyCommit` under this URL. (See the Git plugin wiki page for more details.)

**Details**

Jenkins is a popular continuous integration server.

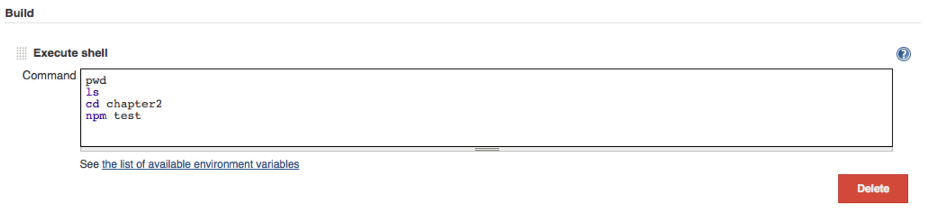
If you're using the standard Jenkins Git plugin to poll & check out your repository, you can quickly and easily switch to a push model using this service.

It will send a request to your Jenkins instance telling it about the repositories and branches that changed. Jenkins will then poll the repository and build if needed. See push notification from repository on the Jenkins wiki for information.

Jenkins url: http://jenkins.c112[redacted]-hang

[✓] Active
We will run this service when an event is triggered.

Add a build step of Executes shell type and write shell scripts to execute the test.



Build

Execute shell

Command

```
pwd
ls
cd chapter2
npm test
```

See the list of available environment variables

The command in this example is as follows.

```
 pwd
ls
cd chapter2
npm test
```

## SVN code example

Select **Subversion** in Source Code Management and enter the SVN repository address in **Repository URL** (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 perform automatic project execution. Enter your token in **Token Name**.

Log on to the SVN server. Create a post-commit in the hooks directory in the source code repository ( svn-java-demo).

```
cd /home/svn/svn-java-demo/hooks
cp post-commit.tmpl post-commit
chmod 755 post-commit
```

Add curl -u ${Jenkins_account}:${password} ${Jenkins_url}/job/svn/build?token=${token} in the post-commit fiel. For example, curl -u test:test http://127.0.0.1:8080/jenkins/job/svn/build?token=qinyujia.

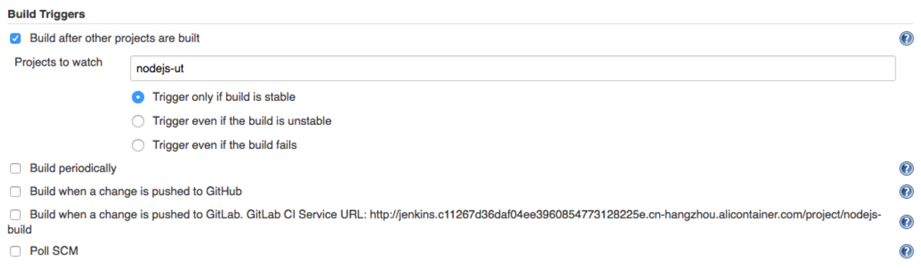
## 2.3 Create a project to automatically build and push images

Create an item and choose to build a free style software project.

Enter the project name and select a node to run the project. In this example, enter the slave-nodejs-ut node created above.

Configure the source code management and code branch. In this example, use GitHub to manage source codes.

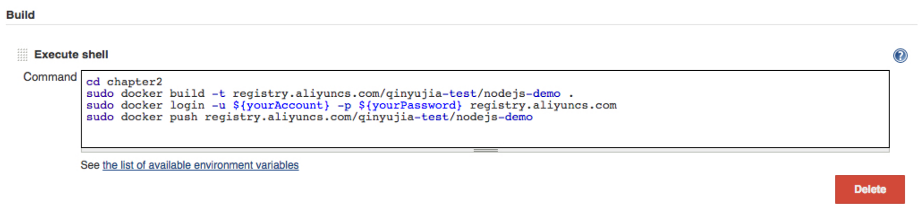
Add the following trigger and set it to implement automatic image building only after success of the unit test.



**Build Triggers**

- [✓] Build after other projects are built
  - Projects to watch: nodejs-ut
  - [✓ Trigger only if build is stable]
  - Trigger even if the build is unstable
  - Trigger even if the build fails
- Build periodically
- Build when a change is pushed to GitHub
- Build when a change is pushed to GitLab. GitLab CI Service URL: http://jenkins.c11267d36daf04ee3960854773128225e.cn-hangzhou.alicontainer.com/project/nodejs-build
- Poll SCM

Write shell scripts for building and pushing images.



**Build**

**Execute shell**

Command

```
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
```

See the list of available environment variables

The command in this example is 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 above to the Container Service and create the nodejs-demo application.

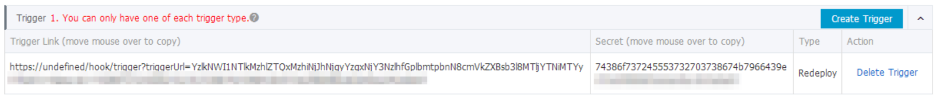
**Example:**

```
express:
image: 'registry.aliyuncs.com/qinyujia-test/nodejs-demo'
expose:
- '22'
- '3000'
restart: always
labels:
aliyun.routing.port_3000: express
```

### 3.2 Automatic redeployment

Select the application nodejs-demo just created, and create the trigger.

**Note:** For how to create a trigger, refer to Trigger.



Trigger 1. You can only have one of each trigger type.

| Trigger Link (move mouse over to copy) | Secret (move mouse over to copy) | Type | Action |
|---|---|---|---|
| https://undefined/hook/trigger?triggerUrl=YzlkNWI1NTlkMzhlZTQxMzhiNjJhNjgyYzgxNjY3NzlhfGplbmtpbnN8cmVkZXBsb3l8MTljYTNiMTYy [redacted] | 74386f737245553732703738674b7966439e [redacted] | Redeploy | Delete Trigger |

Add a line to shell scripts wrote in 2.3. The address is the trigger link given by the trigger created above.

```
curl   'https://cs.console.aliyun.com/hook/trigger?triggerUrl=***==&secret=***'
```

Change the Command in the example of 2.3 as follows.

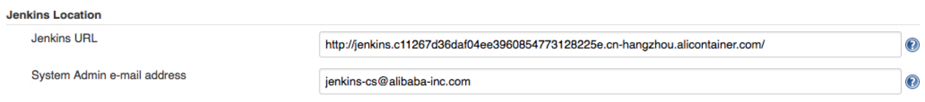
```
 cd chapter2
sudo docker build -t registry.aliyuncs.com/qinyujia-test/nodejs-demo .
sudo docker login -u ${yourAccount} -p ${yourPassword} registry.aliyuncs.com
sudo docker push registry.aliyuncs.com/qinyujia-test/nodejs-demo
curl   'https://cs.console.aliyun.com/hook/trigger?triggerUrl=***==&secret=***'
```

After pushing the image, Jenkins automatically triggers redeployment of the nodejs-demo application.

## Step 4 Configure email notification of the results

If you want to send the unit test or image configuration results to relevant developers or project execution initiators through email, perform the following configurations.

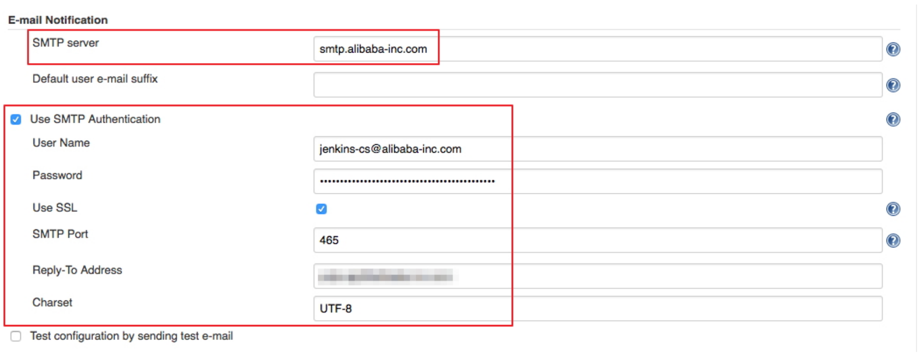
On the Jenkins homepage, click **System Management** > **System Settings**, and configure a Jenkins system administrator email.



**Jenkins Location**

| | |
|---|---|
| Jenkins URL | http://jenkins.c11267d36daf04ee3960854773128225e.cn-hangzhou.alicontainer.com/ |
| System Admin e-mail address | jenkins-cs@alibaba-inc.com |

Install the Extended Email Notification plugin, configure SMTP server and other relevant information, and set the default recipient list.

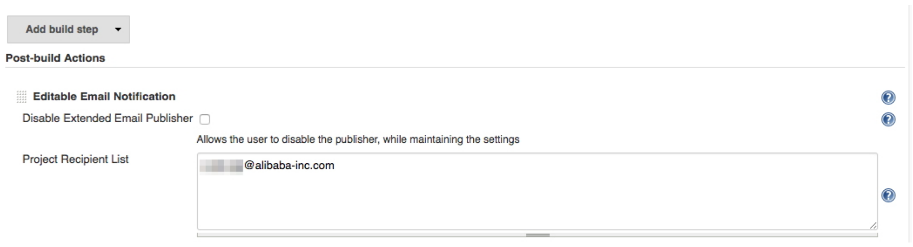


**E-mail Notification**

| | |
|---|---|
| SMTP server | smtp.alibaba-inc.com |
| Default user e-mail suffix | |
| ☑ Use SMTP Authentication | |
| User Name | jenkins-cs@alibaba-inc.com |
| Password | ········································· |
| Use SSL | ☑ |
| SMTP Port | 465 |
| Reply-To Address | [redacted] |
| Charset | UTF-8 |
| ☐ Test configuration by sending test e-mail | |

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 through email.

Add post-building operation steps in the Jenkins project, select **Editable Email Notification**, and enter a recipient list.

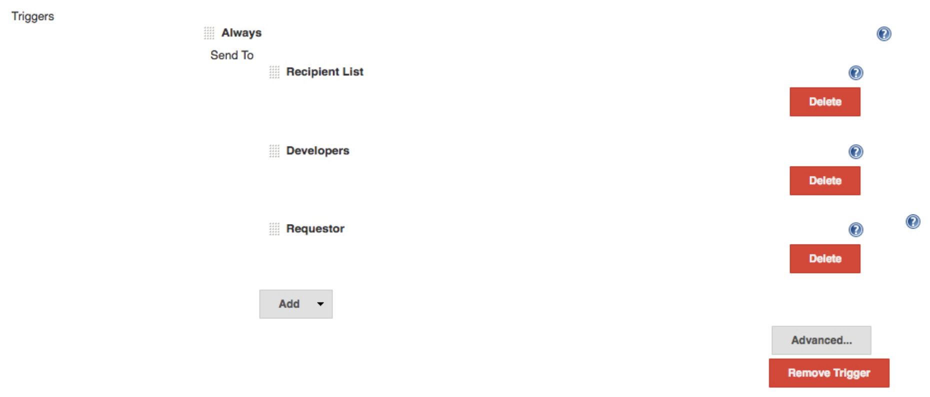


**Post-build Actions**

**Editable Email Notification**

| | |
|---|---|
| Disable Extended Email Publisher | ☐ |
| | Allows the user to disable the publisher, while maintaining the settings |
| Project Recipient List | [redacted]@alibaba-inc.com |

Add a mailing trigger.



# Service discovery and load balancing

## Overview

Communication is classified into two types: communication with externally exposed services and communication between internal services. Service discovery and load balancing are designed to address the issue of reliable communication.

The following section introduces different application scenarios of service discovery and load balancing as well as their solutions.

### Scenario 1

Simple routing is recommended for common and simple Layer-7 load balancing and reverse proxy for web services. For details, refer to Expose HTTP services through acsrouting, Add domain names to services exposed to the public network, and Change exposed HTTP services to HTTPS services.

### Scenario 2

In Layer-4 load balancing, loads are evenly distributed to functionally identical containers, and the services of non-container clusters access the services of containers deployed in clusters during the migration of a traditional architecture to a container architecture. In this scenario, Expose services using custom Server Load Balancer is recommended.

### Scenario 3

Services in the same cluster need the ability to discover and communicate with each other, and need load balancing capabilities. In this scenario, Load balancing and automatic service discovery between microservices is recommended.

## Scenario 4

Services in the same cluster need the ability to discover and communicate with each other, but do not need load balancing capabilities. In this scenario, Service discovery between containers is recommended.

## Scenario 5

Load balancing and service discovery are customized according to relatively high requirements, such as support for wildcard domain names, custom error page creation, access logging, selection of backend services based on URL parameter values, and creation of custom HAProxy configuration files.

# Expose HTTP services through acsrouting

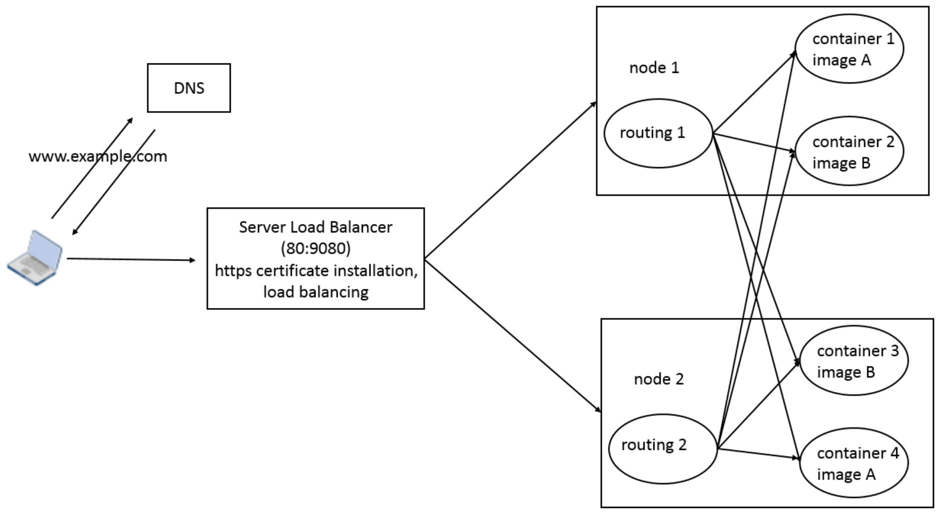
## Applicable scenario

In simple Layer-7 load balancing with the web routing service, services in a container cluster access each other using Layer-7 protocols through communication proxy and Server Load Balancer.

## Concept

When a cluster is created, by default, it is assigned a Server Load Balancer instance used to add all nodes in the cluster to the instance backend. Port 80 is exposed at the frontend, and Port 9080 is exposed on all machines on which backend nodes run. Alibaba Cloud Container Service Routing (acsrouting) is an ongoing global project and has only one service (routing service). acsrouting sees that a copy (or a container) of the service (or an image) is deployed on each node (a node is also called a host or an ECS VM instance). Each node uses its container to route HTTP services or HTTPS services.

See the figure below. For HTTP services, the mapping between Server Load Balancer frontend and backend ports is 80:9080, and the host-container port mapping 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 by the web service. After you set the host-container port mapping during container startup, the routing service can obtain the corresponding port for request routing. For a complete example about how to expose HTTP services, refer to Create Nginx from an image.



## Setup methods

### Set through the Container Service console

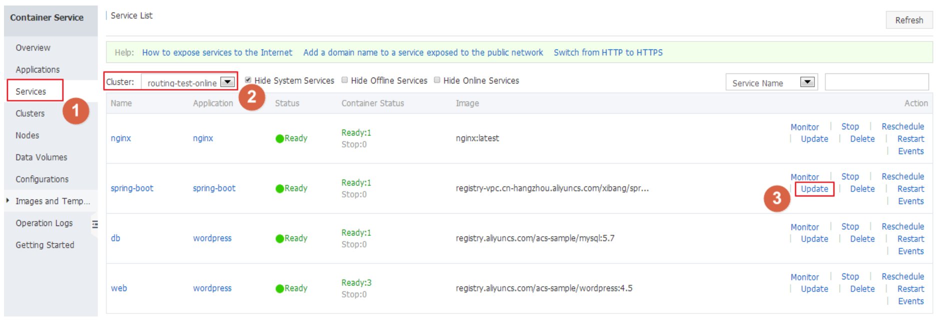
#### Set through Services > Update

Log on to the Container Service console.

Click **Services** in the left navigation pane.

Select the cluster of the desired service.

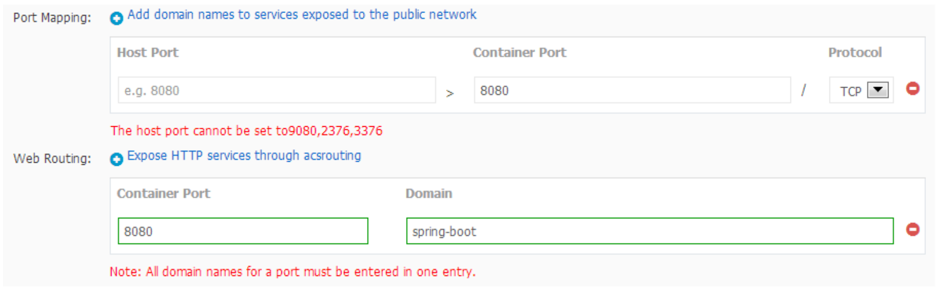
Locate the service to be exposed (**spring-boot** is used as an example) and click **Update**.



| Name | Application | Status | Container Status | Image | Action |
|---|---|---|---|---|---|
| nginx | nginx | ● Ready | Ready:1 Stop:0 | nginx:latest | Monitor \| Stop \| Reschedule \| Update \| Delete \| Restart \| Events |
| spring-boot | spring-boot | ● Ready | Ready:1 Stop:0 | registry-vpc.cn-hangzhou.aliyuncs.com/xibang/spr... | Monitor \| Stop \| Reschedule \| Update \| Delete \| Restart \| Events |
| db | wordpress | ● Ready | Ready:1 Stop:0 | registry.aliyuncs.com/acs-sample/mysql:5.7 | Monitor \| Stop \| Reschedule \| Update \| Delete \| Restart \| Events |
| web | wordpress | ● Ready | Ready:3 Stop:0 | registry.aliyuncs.com/acs-sample/wordpress:4.5 | Monitor \| Stop \| Reschedule \| Update \| Delete \| Restart \| Events |

Configure the host-container port mapping, as shown in the figure below.

As shown in the following figure, the host port is empty, indicating that a random port on the host is exposed (when HTTP or HTTPS services are exposed, you do not need to know what port is exposed on the host, because the container port is directly accessed through an overlay network or VPC network). The container port is Port 8080. Use spring-boot to enable default exposure of Port 8080 of the web service to provision HTTP services. The used protocol is TCP.



Port Mapping: Add domain names to services exposed to the public network

| Host Port | Container Port | Protocol |
|---|---|---|
| e.g. 8080 | 8080 | TCP |

The host port cannot be set to9080,2376,3376

Web Routing: Expose HTTP services through acsrouting

| Container Port | Domain |
|---|---|
| 8080 | spring-boot |

Note: All domain names for a port must be entered in one entry.

Routing configuration enables service exposure through a domain name. You must specify the exposed port (Port 8080 of the web service is used as an example). You only need to enter the domain name prefix in **Domain**. If the domain name prefix is XXX, you obtain the domain name XXX.$cluster_id.$region_id.alicontainer.com used for testing. In this example, you obtain the domain name spring-boot.c0cffb4340aee47ccb26dea062cfb0b2e.cn-beijing.alicontainer.com. You can enter your own domain name, which needs to be resolved 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 used to access HTTP services, refer to **routing** in Label description.

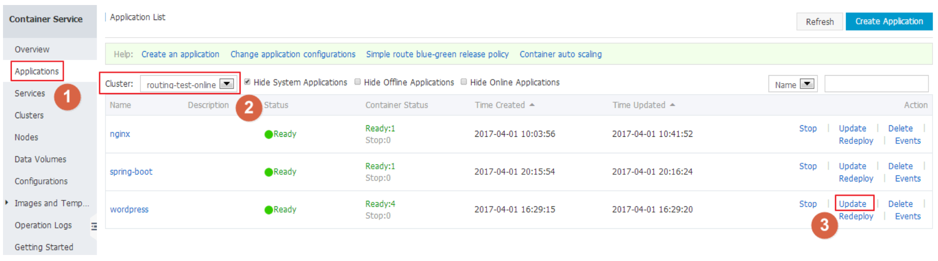
## Set through Applications > Update

Log on to the Container Service console.

Click **Applications** in the left navigation pane.

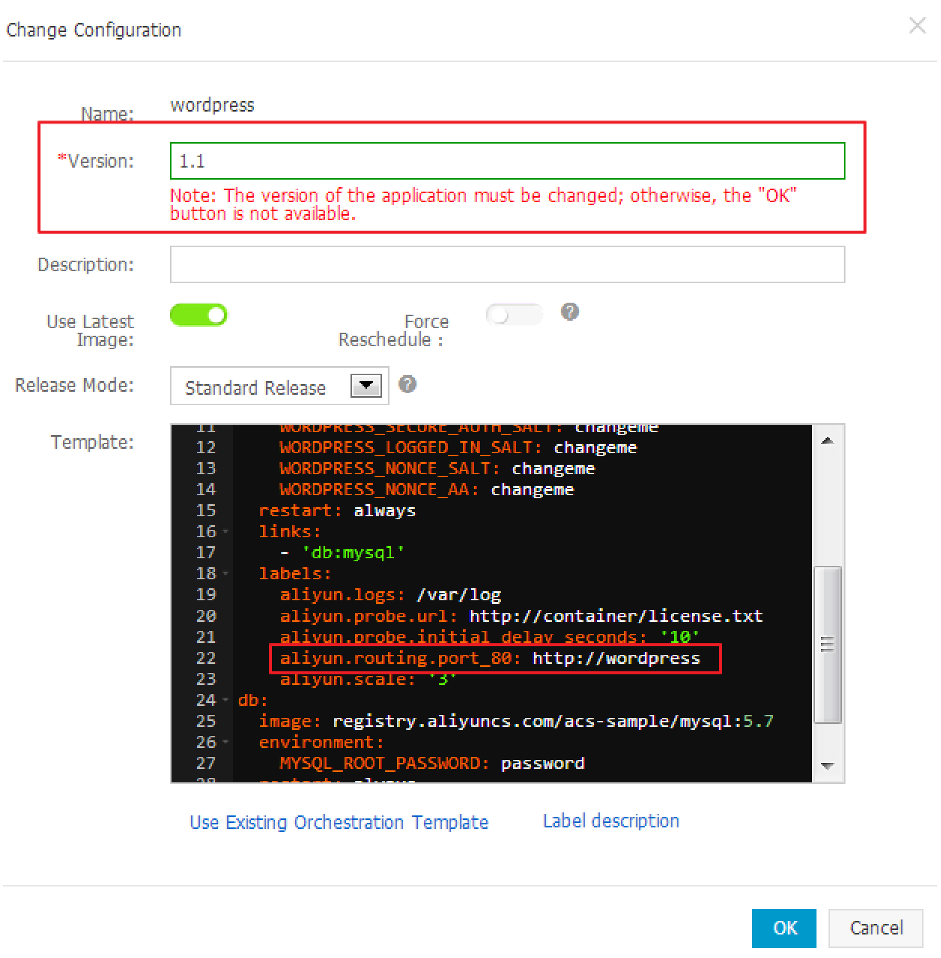
Select the cluster of the desired application.

Locate the application (in this example, the application **wordpress** is used) and click **Update**.



| Name | Description | Status | Container Status | Time Created | Time Updated | Action |
|---|---|---|---|---|---|---|
| nginx | | Ready | Ready:1 Stop:0 | 2017-04-01 10:03:56 | 2017-04-01 10:41:52 | Stop \| Update \| Delete \| Redeploy \| Events |
| spring-boot | | Ready | Ready:1 Stop:0 | 2017-04-01 20:15:54 | 2017-04-01 20:16:24 | Stop \| Update \| Delete \| Redeploy \| Events |
| wordpress | | Ready | Ready:4 Stop:0 | 2017-04-01 16:29:15 | 2017-04-01 16:29:20 | Stop \| Update \| Delete \| Redeploy \| Events |

Add a routing label in **Template**, define the corresponding domain name or domain name prefix, and check whether the updated project version is correct. Then click **OK** to update the domain name.



Change Configuration

Name: wordpress

*Version: 1.1

Note: The version of the application must be changed; otherwise, the "OK" button is not available.

Description:

Use Latest Image:

Force Reschedule :

Release Mode: Standard Release

Template:

```
      WORDPRESS_SECURE_AUTH_SALT: changeme
      WORDPRESS_LOGGED_IN_SALT: changeme
      WORDPRESS_NONCE_SALT: changeme
      WORDPRESS_NONCE_AA: changeme
    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: '3'
  db:
    image: registry.aliyuncs.com/acs-sample/mysql:5.7
    environment:
      MYSQL_ROOT_PASSWORD: password
```

## Set through client

- docker help run: View the used “-p” option. Set the routing configuration on the Container Service management console.
- docker-compose: View the supported “ports” option. For routing configuration rules, refer to **routing** in Label description.

# Add domain names to services exposed to the public network

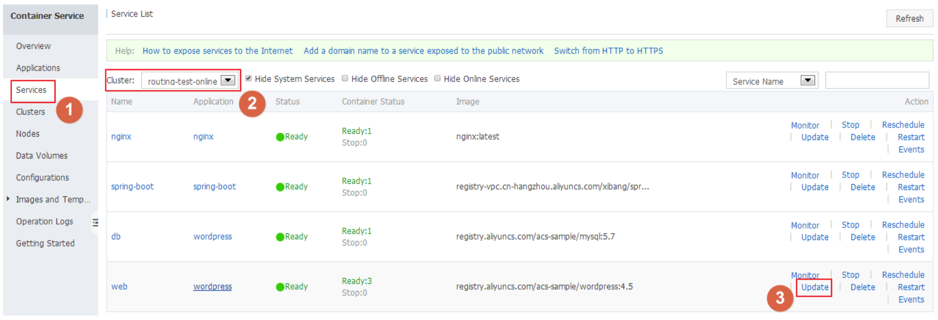
Log on to the Container Service console.

Click **Services** in the left navigation pane.

Select the cluster of the desired service.

Locate the service to which you want to add a domain name and click **Update**.

The service **web** belonging to the application **wordpress** is used as an example.

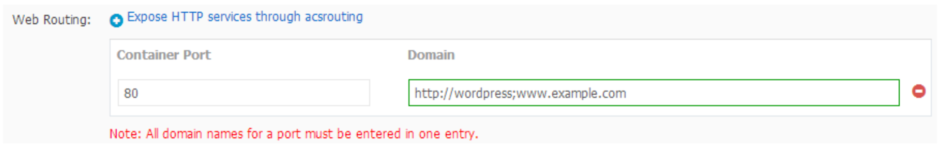


Cluster: routing-test-online

| Name | Application | Status | Container Status | Image | Action |
|---|---|---|---|---|---|
| nginx | nginx | ● Ready | Ready:1 Stop:0 | nginx:latest | Monitor \| Stop \| Reschedule \| Update \| Delete \| Restart \| Events |
| spring-boot | spring-boot | ● Ready | Ready:1 Stop:0 | registry-vpc.cn-hangzhou.aliyuncs.com/xibang/spr... | Monitor \| Stop \| Reschedule \| Update \| Delete \| Restart \| Events |
| db | wordpress | ● Ready | Ready:1 Stop:0 | registry.aliyuncs.com/acs-sample/mysql:5.7 | Monitor \| Stop \| Reschedule \| Update \| Delete \| Restart \| Events |
| web | wordpress | ● Ready | Ready:3 Stop:0 | registry.aliyuncs.com/acs-sample/wordpress:4.5 | Monitor \| Stop \| Reschedule \| Update \| Delete \| Restart \| Events |

Click the plus icon next to **Web Routing**, enter the domain name to be added (www.example.com is used as an example), and click **OK** to update the configuration.

> **Note:** When multiple domain names are added under the same port for the same service, they must be separated by a semicolon (;).



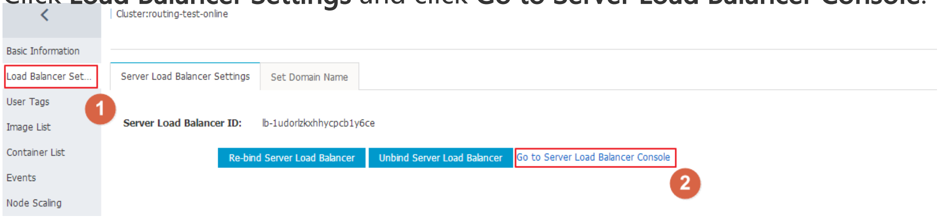
| Container Port | Domain |
|---|---|
| 80 | http://wordpress;www.example.com |

Note: All domain names for a port must be entered in one entry.

Wait until the service finishes the update and enters the ready state. After that, the routing service **acsrouting_routing** finishes domain name configuration. When requests containing the domain name www.example.com are sent for access to the service **wordpress_web**, the requests can be correctly parsed and forwarded to the corresponding service.

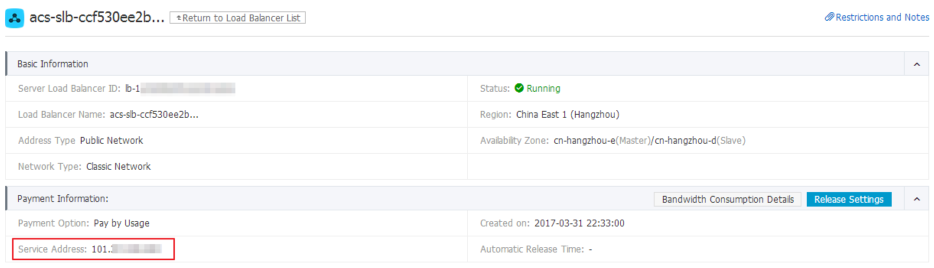
Resolve the domain name to the cluster of the Container Service. When you create a cluster, the Container Service assigns a Server Load Balancer instance to each cluster, and the instance belongs to you.

i. Click **Clusters** in the left navigation pane in the **Container Service console**.
ii. Locate the corresponding cluster (the cluster **routing-test-online** is used as an example) and click **Manage**.
iii. Click **Load Balancer Settings** and click **Go to Server Load Balancer Console**.


Cluster:routing-test-online
Basic Information
Load Balancer Set...
User Tags
Image List
Container List
Events
Node Scaling
Server Load Balancer Settings
Set Domain Name
Re-bind Server Load Balancer
Unbind Server Load Balancer
Go to Server Load Balancer Console
1
2

Server Load Balancer ID: lb-1udorlzkxhhycpcb1y6ce

You can view the service address of the Server Load Balancer instance.



acs-slb-ccf530ee2b... Return to Load Balancer List | Restrictions and Notes

**Basic Information**

| | |
|---|---|
| Server Load Balancer ID: lb-1[redacted] | Status: Running |
| Load Balancer Name: acs-slb-ccf530ee2b... | Region: China East 1 (Hangzhou) |
| Address Type Public Network | Availability Zone: cn-hangzhou-e(Master)/cn-hangzhou-d(Slave) |
| Network Type: Classic Network | |

**Payment Information:** Bandwidth Consumption Details | Release Settings

| | |
|---|---|
| Payment Option: Pay by Usage | Created on: 2017-03-31 22:33:00 |
| Service Address: 101.[redacted] | Automatic Release Time: - |

Ask your DNS resolver service provider to resolve your domain (www.example.com in this example) to the Server Load Balancer VIP address, namely add an **A** record.

Access http://www.example.com. You will see the Hello World page of WordPress.

# Change exposed HTTP services to HTTPS services

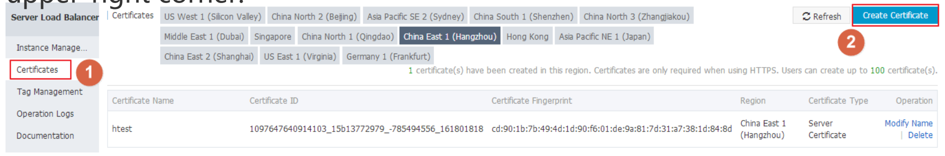
## Prerequisite

An HTTP domain name for access has been configured. For details, refer to Add domain names to services exposed to the public network.

## Operating procedure

HTTPS is supported at the Server Load Balancer layer. To support HTTPS, a Server Load Balancer certificate must be created.

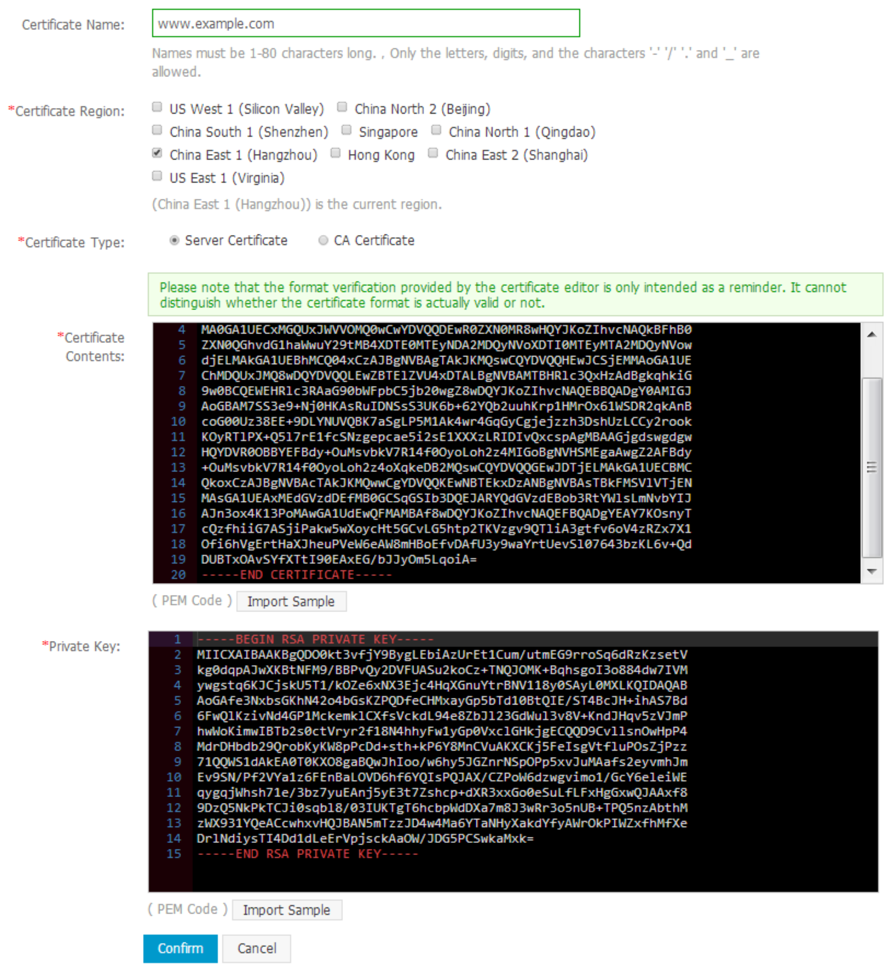
i. Log on to the Server Load Balancer console.
ii. Click **Certificates** on the left navigation pane and click **Create Certificate** in the upper-right corner.



1 certificate(s) have been created in this region. Certificates are only required when using HTTPS. Users can create up to 100 certificate(s).

| Certificate Name | Certificate ID | Certificate Fingerprint | Region | Certificate Type | Operation |
|---|---|---|---|---|---|
| htest | 1097647640914103_15b13772979_-785494556_161801818 | cd:90:1b:7b:49:4d:1d:90:f6:01:de:9a:81:7d:31:a7:38:1d:84:8d | China East 1 (Hangzhou) | Server Certificate | Modify Name \| Delete |

iii. Enter certificate information, and click **Confirm**.



Certificate Name: www.example.com

Names must be 1-80 characters long. , Only the letters, digits, and the characters '-' '/' '.' and '_' are allowed.

*Certificate Region: US West 1 (Silicon Valley) China North 2 (Beijing) China South 1 (Shenzhen) Singapore China North 1 (Qingdao) [✓ China East 1 (Hangzhou)] Hong Kong China East 2 (Shanghai) US East 1 (Virginia)

(China East 1 (Hangzhou)) is the current region.

*Certificate Type: [✓ Server Certificate] CA Certificate

Please note that the format verification provided by the certificate editor is only intended as a reminder. It cannot distinguish whether the certificate format is actually valid or not.

*Certificate Contents:

```
MA0GA1UECxMGQUxJWVVOMQ0wCwYDVQQDEwR0ZXN0MR8wHQYJKoZIhvcNAQkBFhB0
ZXN0QGhvdG1haWwuY29tMB4XDTE0MTEyNDA2MDQyNVoXDTI0MTEyMTA2MDQyNVow
djELMAkGA1UEBhMCQ04xCzAJBgNVBAgTAkJKMQswCQYDVQQHEwJCSjEMMAoGA1UE
ChMDQUxJMQ8wDQYDVQQLEwZBTE1ZVU4xDTALBgNVBAMTBHRlc3QxHzAdBgkqhkiG
9w0BCQEWEHRlc3RAaG90bWFpbC5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJ
AoGBAM7SS3e9+Nj0HKAsRuIDNSsS3UK6b+62YQb2uuhKrp1HMrOx61WSDR2qkAnB
coG00Uz38EE+9DLYNUVQBK7aSgLP5M1Ak4wr4GqGyCgjejzzh3DshUzLCCy2rook
KOyRT1PX+Q517rE1fcSNzgepcae5i2sE1XXXzLRIDIvQxcspAgMBAAGjgdswgdgw
HQYDVR0OBBYEFBdy+OuMsvbkV7R14f0OyoLoh2z4MIGoBgNVHSMEgaAwgZ2AFBdy
+OuMsvbkV7R14f0OyoLoh2z4oXqkeDB2MQswCQYDVQQGEwJDTjELMAkGA1UECBMC
QkoxCzAJBgNVBAcTAkJKMQwwCgYDVQQKEwNBTEkxDzANBgNVBAsTBkFMSV1VTjEN
MAsGA1UEAxMEdGVzdDEfMB0GCSqGSIb3DQEJARYQdGVzdEBob3RtYWlsLmNvbYIJ
AJn3ox4K13PoMAwGA1UdEwQFMAMBAf8wDQYJKoZIhvcNAQEFBQADgYEAY7KOsnyT
cQzfhiiG7ASjiPakw5wXoycHt5GCvLG5htp2TKVzgv9QTliA3gtfv6oV4zRZx7X1
Ofi6hVgErtHaXJheuPVeW6eAW8mHBoEfvDAfU3y9waYrtUevSl07643bzKL6v+Qd
DUBTxOAvSYfXTtI90EAxEG/bJJyOm5LqoiA=
-----END CERTIFICATE-----
```

( PEM Code ) Import Sample

*Private Key:

```
-----BEGIN RSA PRIVATE KEY-----
MIICXAIBAAKBgQDO0kt3vfjY9BygLEbiAzUrEt1Cum/utmEG9rroSq6dRzKzsetV
kg0dqpAJwXKBtNFM9/BBPvQy2DVFUASu2koCz+TNQJOMK+BqhsgoI3o884dw7IVM
ywgstq6KJCjskU5T1/kOZe6xNX3Ejc4HqXGnuYtrBNV118y0SAyL0MXLKQIDAQAB
AoGAfe3NxbsGKhN42o4bGsKZPQDfeCHMxayGp5bTd10BtQIE/ST4BcJH+ihAS7Bd
6FwQlKzivNd4GP1MckemklCXfsVckdL94e8ZbJl23GdWul3v8V+KndJHqv5zVJmP
hwWoKimwIBTb2s0ctVryr2f18N4hhyFw1yGp0VxclGHkjgECQQD9CvllsnOwHpP4
MdrDHbdb29QrobKyKW8pPcDd+sth+kP6Y8MnCVuAKXCKj5FeIsgVtfluPOsZjPzz
71QQWS1dAkEA0T0KXO8gaBQwJhIoo/w6hy5JGZnrNSpOPp5xvJuMAafs2eyvmhJm
Ev9SN/Pf2VYa1z6FEnBaLOVD6hf6YQIsPQJAX/CZPoW6dzwgvimo1/GcY6eleiWE
qygqjWhsh71e/3bz7yuEAnj5yE3t7Zshcp+dXR3xxGo0eSuLfLFxHgGxwQJAAxf8
9DzQ5NkPkTCJi0sqbl8/03IUKTgT6hcbpWdDXa7m8J3wRr3o5nUB+TPQ5nzAbthM
zWX931YQeACcwhxvHQJBAN5mTzzJD4w4Ma6YTaNHyXakdYfyAWrOkPIWZxfhMfXe
DrlNdiysTI4Dd1dLeErVpjsckAaOW/JDG5PCSwkaMxk=
-----END RSA PRIVATE KEY-----
```

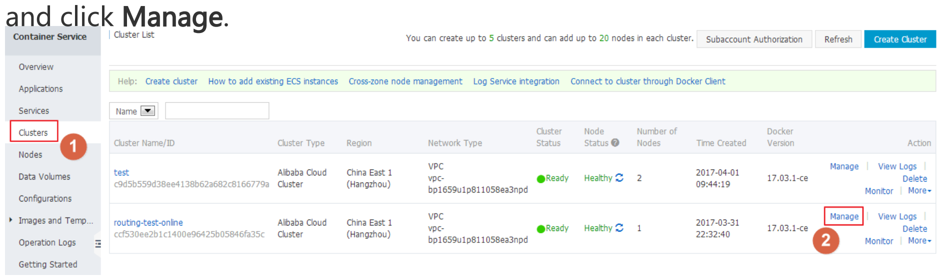
( PEM Code ) Import Sample

Confirm Cancel

After the certificate is successfully created, locate the Server Load Balancer instance that is assigned during cluster creation.

When you create clusters, the Container Service assigns a Server Load Balancer instance to each cluster, and the instance belongs to you.

i. On the **Container Service console**, click **Clusters** on the left navigation pane, select the corresponding cluster (the cluster **routing-test-online** is used as an example), and click **Manage**.

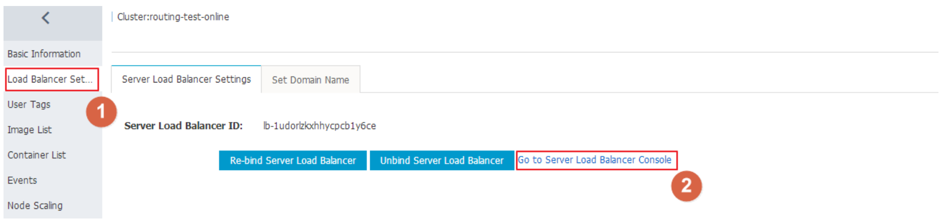


You can create up to 5 clusters and can add up to 20 nodes in each cluster.

Help: Create cluster How to add existing ECS instances Cross-zone node management Log Service integration Connect to cluster through Docker Client

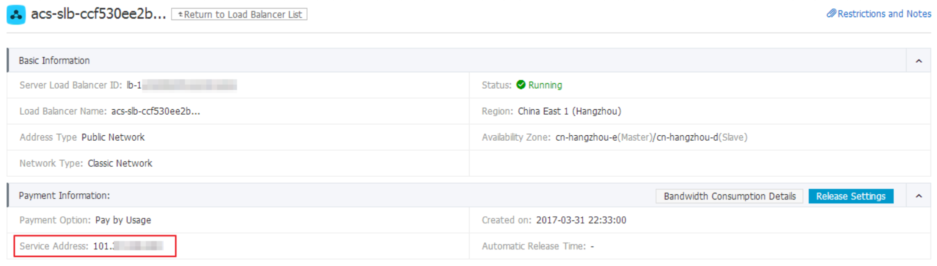
| Cluster Name/ID | Cluster Type | Region | Network Type | Cluster Status | Node Status | Number of Nodes | Time Created | Docker Version | Action |
|---|---|---|---|---|---|---|---|---|---|
| test c9d5b559d38ee4138b62a682c8166779a | Alibaba Cloud Cluster | China East 1 (Hangzhou) | VPC vpc-bp1659u1p811058ea3npd | ● Ready | Healthy | 2 | 2017-04-01 09:44:19 | 17.03.1-ce | Manage \| View Logs \| Delete Monitor \| More |
| routing-test-online ccf530ee2b1c1400e96425b05846fa35c | Alibaba Cloud Cluster | China East 1 (Hangzhou) | VPC vpc-bp1659u1p811058ea3npd | ● Ready | Healthy | 1 | 2017-03-31 22:32:40 | 17.03.1-ce | Manage \| View Logs \| Delete Monitor \| More |

ii. Click **Load Balancer Settings** and click **Go to Server Load Balancer Console**.


Cluster:routing-test-online
Basic Information
Load Balancer Set...
User Tags
Image List
Container List
Events
Node Scaling
Server Load Balancer Settings
Set Domain Name
Re-bind Server Load Balancer
Unbind Server Load Balancer
Go to Server Load Balancer Console
1
2

Server Load Balancer ID: lb-1udorlzkxhhycpcb1y6ce

You can view the service address of the Server Load Balancer instance.

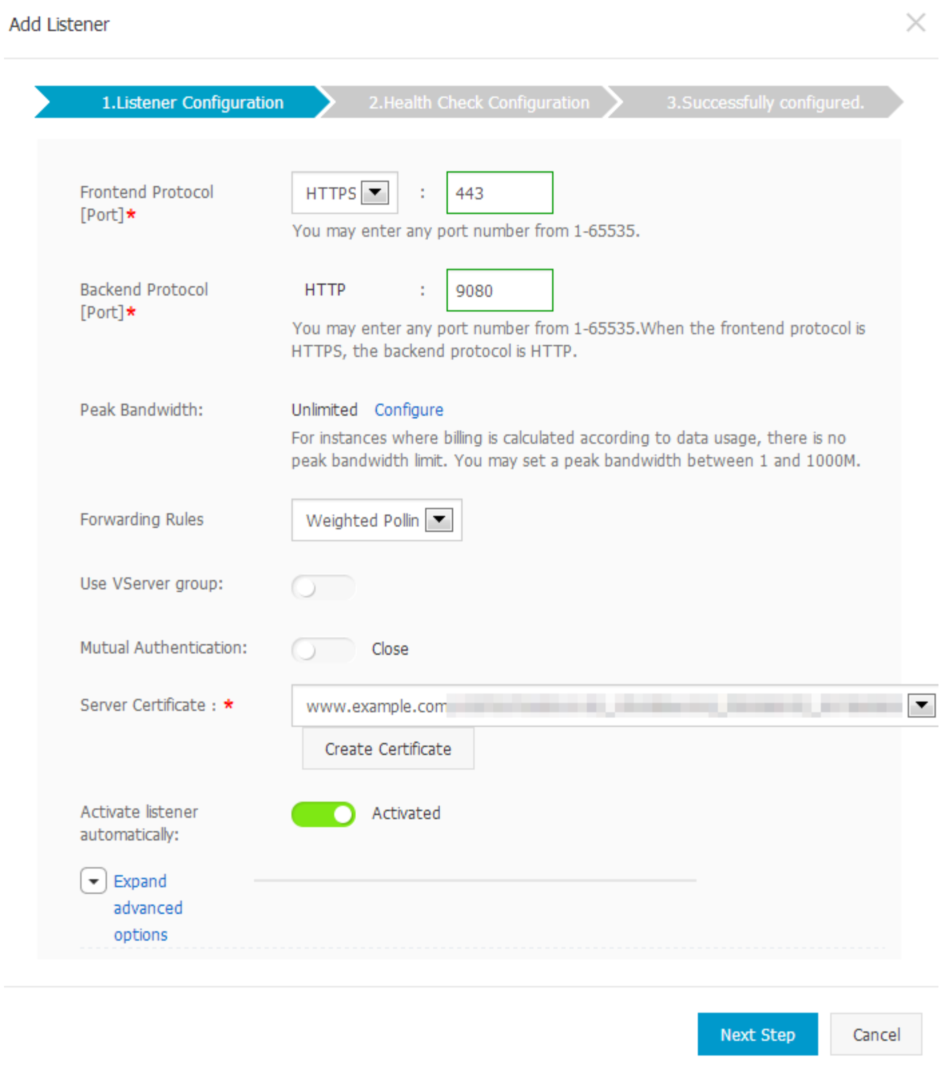


| | |
|---|---|
| Server Load Balancer ID: lb-1[redacted] | Status: Running |
| Load Balancer Name: acs-slb-ccf530ee2b... | Region: China East 1 (Hangzhou) |
| Address Type: Public Network | Availability Zone: cn-hangzhou-e(Master)/cn-hangzhou-d(Slave) |
| Network Type: Classic Network | |
| Payment Option: Pay by Usage | Created on: 2017-03-31 22:33:00 |
| Service Address: 101.[redacted] | Automatic Release Time: - |

Click **Listening** in the left navigation pane and click **Create Listener**. In the **Add Listener** page, enter the following port information.

```
 +----------------+-------+------+
| | Protocol | Port |
+----------------+-------+------+
| Front-end protocol (port) | HTTPS | 443 |
+----------------+-------+------+
| Backend protocol (port) | HTTP | 9080 |
+----------------+-------+------+
```

i. Select HTTPS for the frontend protocol.
ii. Set the frontend port to Port 443 and backend port to Port 9080. (Port 9080 is exposed by the routing service **acsrouting_routing** on each ECS host. All HTTP requests are forwarded based on the HOST HTTP header 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 Step**.

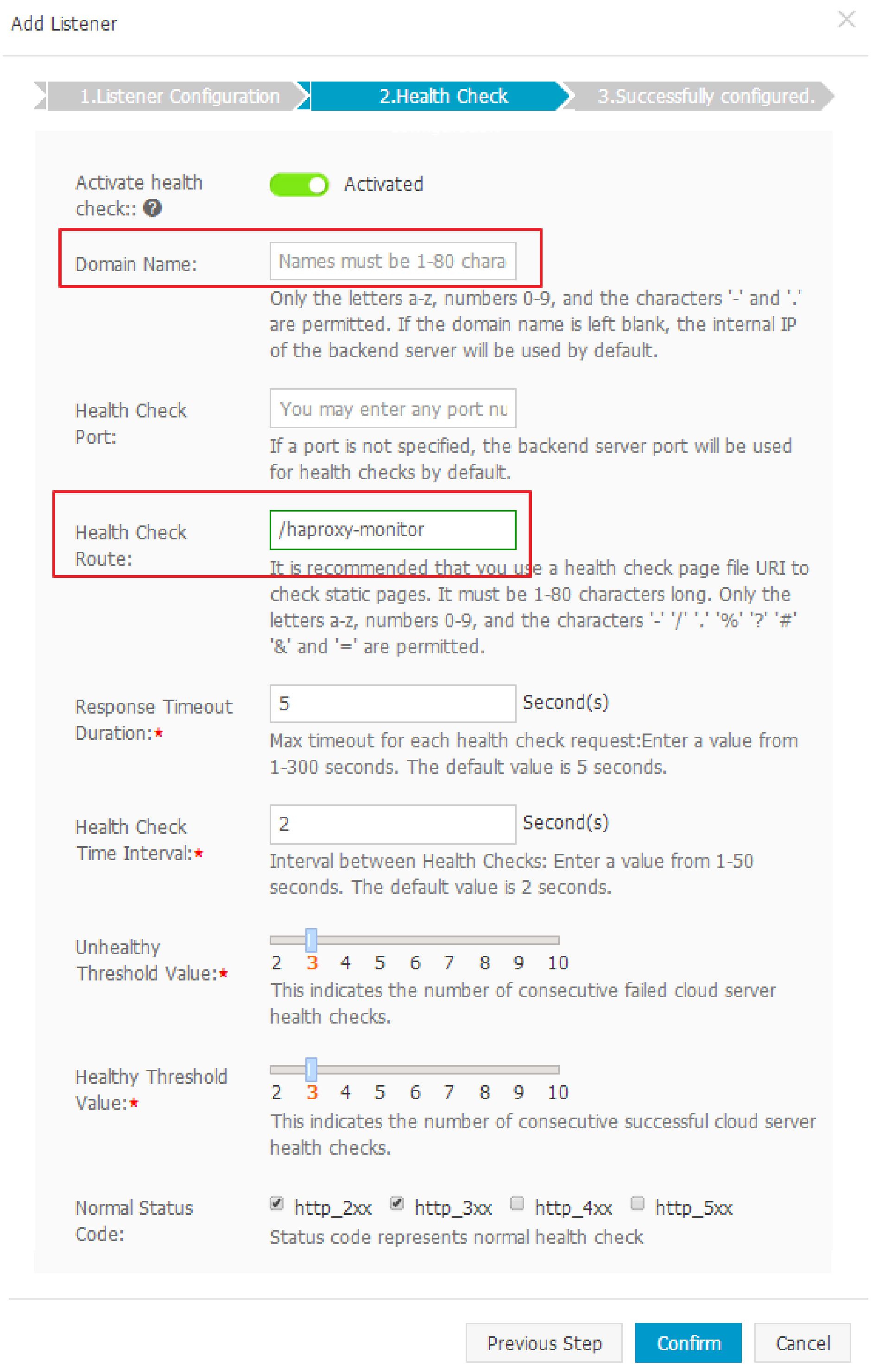

Add Listener

1.Listener Configuration | 2.Health Check Configuration | 3.Successfully configured.

| | |
|---|---|
| Frontend Protocol [Port]* | HTTPS : 443<br>You may enter any port number from 1-65535. |
| Backend Protocol [Port]* | HTTP : 9080<br>You may enter any port number from 1-65535.When the frontend protocol is HTTPS, the backend protocol is HTTP. |
| Peak Bandwidth: | Unlimited Configure<br>For instances where billing is calculated according to data usage, there is no peak bandwidth limit. You may set a peak bandwidth between 1 and 1000M. |
| Forwarding Rules | Weighted Pollin |
| Use VServer group: | |
| Mutual Authentication: | Close |
| Server Certificate : * | www.example.com<br>Create Certificate |
| Activate listener automatically: | Activated |

Complete settings on the **Health check** page. Then click **Confirm**.

You can select to disable or enable the health check. If you select to enable the health check, you have to enter your own domain name in **Domain Name** or enter /haproxy-monitor in **Health Check Route**; otherwise, the health check will be abnormal.


Add Listener

1.Listener Configuration | 2.Health Check | 3.Successfully configured.

Activate health check:: Activated

Domain Name: Names must be 1-80 chara
Only the letters a-z, numbers 0-9, and the characters '-' and '.' are permitted. If the domain name is left blank, the internal IP of the backend server will be used by default.

Health Check Port: You may enter any port nu
If a port is not specified, the backend server port will be used for health checks by default.

Health Check Route: /haproxy-monitor
It is recommended that you use a health check page file URI to check static pages. It must be 1-80 characters long. Only the letters a-z, numbers 0-9, and the characters '-' '/' '.' '%' '?' '#' '&' and '=' are permitted.

Response Timeout Duration:* 5 Second(s)
Max timeout for each health check request:Enter a value from 1-300 seconds. The default value is 5 seconds.

Health Check Time Interval:* 2 Second(s)
Interval between Health Checks: Enter a value from 1-50 seconds. The default value is 2 seconds.

Unhealthy Threshold Value:* 2 [✓ 3] 4 5 6 7 8 9 10
This indicates the number of consecutive failed cloud server health checks.

Healthy Threshold Value:* 2 [✓ 3] 4 5 6 7 8 9 10
This indicates the number of consecutive successful cloud server health checks.

Normal Status Code: ☑ http_2xx ☑ http_3xx ☐ http_4xx ☐ http_5xx
Status code represents normal health check

After the listener is successfully created and started, click **Confirm**.

Access the page https://www.example.com.

# Load balancing and automatic service discovery between microservices

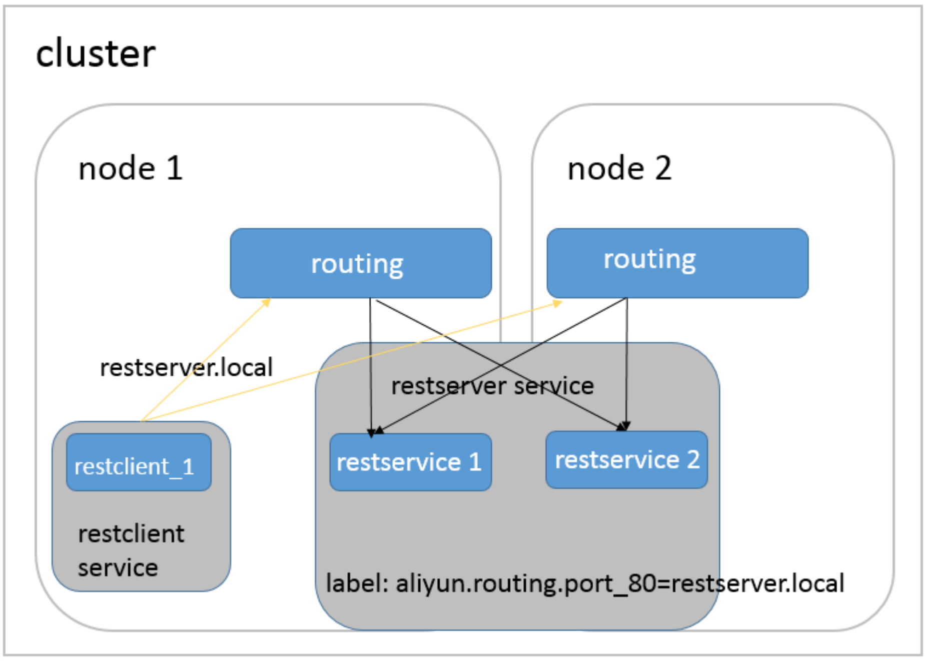
In the Container Service, the HTTP service can be exposed based on domain using acsrouting, and health checks can be used to automatically enable load balancing and service discovery. When a container has a problem, the routing will automatically remove the container with a failed health check at the backend. This enables automatic service discovery.

However, the preceding method exposes the service to Internet, so how can automatic service discovery and load balancing be achieved between services using this method? The Container Service uses the Server Load Balancer. You only need to use the domain ending with .local, and add this domain to external_links in the dependent service, and then the dependent service can access the service using this domain ending with .local, and work with health checks to achieve automatic service discovery.

## Concept

1. Docker version later than 1.10 supports alias resolution in the container. In the container of the restserver.local service which depends on the load, the restserver.local domain is resolved into the address of the routing service. In this way, the HTTP request can be forwarded to the routing service container, with HOST in the requested header of restserver.local.

   Routing monitors the health status of the container of the service configured with aliyun.routing_port_xxx: restserver.local, and attaches the status to the backend of HAProxy. After HAProxy receives the HTTP request with the restserver.local HOST header, it can forward that to the corresponding container.



## Advantages

Compared with the DNS-based method using link or hostname, the inconsistent handling of DNS caching by different clients will firstly delay service discovery, and secondly, the DNS solution only include round robin, which is insufficient to support micro-services.

Compared with other microservice discovery solutions, this provides a mechanism to achieve unrelated service discovery and Server Load Balancer, which can be used without having to perform any modification at the server side or client application.

Moreover, in decoupling service lifecycle, every microservice can adopt a Docker Compose template to allow itself to be independently deployed and updated. Only a virtual domain is required to achieve dynamic mutual binding.

## Orchestration instance

```
restserver: # Simulate the rest service
image: nginx
labels:
aliyun.routing.port_80: restserver.local # Use the local domain and the container only in the cluster can access this
domain
aliyun.scale: "2" # Scale 2 instances, to simulate the Server Load Balancer
aliyun.probe.url: "http://container:80" # Define the policy of container health check as http through Port 80.
aliyun.probe.initial_delay_seconds: "2" # The health check starts 2 seconds after the container is enabled
aliyun.probe.timeout_seconds: "2" # Timeout for the health check. A container is deemed unhealthy if no result is
returned in two seconds.
restclient: # Simulate rest service consumer
image: registry.aliyuncs.com/acs-sample/alpine:3.3
command: "sh -c 'apk update; apk add curl; while true; do curl --head restserver.local; sleep 1; done'" # Access the
rest service, and test the Server Load Balancer
tty: true
external_links:
- "restserver.local" # Specify link service domain. Make sure that you set external_links; otherwise, the access might
fail.
```

The following restclient service logs show that the HTTP request of restclient curl is routed to the containers of different rest services. The container ID is respectively 053cb232fdfbcb5405ff791650a0746ab77f26cce74fea2320075c2af55c975f and b8c36abca525ac7fb02d2a9fcaba8d36641447a774ea956cd93068419f17ee3f.

```

internal-loadbalance_restclient_1 | 2016-07-01T06:43:49.066803626Z Server: nginx/1.11.1
internal-loadbalance_restclient_1 | 2016-07-01T06:43:49.066814507Z Date: Fri, 01 Jul 2016 06:43:49 GMT
internal-loadbalance_restclient_1 | 2016-07-01T06:43:49.066821392Z Content-Type: text/html
internal-loadbalance_restclient_1 | 2016-07-01T06:43:49.066829291Z Content-Length: 612
internal-loadbalance_restclient_1 | 2016-07-01T06:43:49.066835259Z Last-Modified: Tue, 31 May 2016 14:40:22
GMT
internal-loadbalance_restclient_1 | 2016-07-01T06:43:49.066841201Z ETag: "574da256-264"
internal-loadbalance_restclient_1 | 2016-07-01T06:43:49.066847245Z Accept-Ranges: bytes
internal-loadbalance_restclient_1 | 2016-07-01T06:43:49.066853137Z Set-Cookie:
CONTAINERID=053cb232fdfbcb5405ff791650a0746ab77f26cce74fea2320075c2af55c975f; path=/

internal-loadbalance_restclient_1 | 2016-07-01T06:43:50.080502413Z HTTP/1.1 200 OK
internal-loadbalance_restclient_1 | 2016-07-01T06:43:50.082548154Z Server: nginx/1.11.1
internal-loadbalance_restclient_1 | 2016-07-01T06:43:50.082559109Z Date: Fri, 01 Jul 2016 06:43:50 GMT
internal-loadbalance_restclient_1 | 2016-07-01T06:43:50.082589299Z Content-Type: text/html
internal-loadbalance_restclient_1 | 2016-07-01T06:43:50.082596541Z Content-Length: 612
internal-loadbalance_restclient_1 | 2016-07-01T06:43:50.082602580Z Last-Modified: Tue, 31 May 2016 14:40:22
GMT
internal-loadbalance_restclient_1 | 2016-07-01T06:43:50.082608807Z ETag: "574da256-264"
internal-loadbalance_restclient_1 | 2016-07-01T06:43:50.082614780Z Accept-Ranges: bytes
internal-loadbalance_restclient_1 | 2016-07-01T06:43:50.082621152Z Set-Cookie:
CONTAINERID=b8c36abca525ac7fb02d2a9fcaba8d36641447a774ea956cd93068419f17ee3f; path=/
```

# Expose services using custom Server Load Balancer

## Expose HTTP or HTTPS services

Routing is recommended for exposing HTTP or HTTPS services. If you want to build routes, activate an intranet-based or Internet-based Server Load Balancer instance with a route destined for the VM port (which can be implemented using the label method aliyun.lb.port_$container_port), and then configure the host-container mapping used to route requests.

**Application scenario:**

In Layer-7 load balancing, you can define a route for each service. When a traditional architecture is migrated to a container architecture, the services of non-container clusters access the services of containers deployed in clusters.

## Expose TCP or UDP services

To expose TCP services, you need to configure a Server Load Balancer instance or a public IP address, and configure the mapping between host port and container port (which can be implemented using the label method aliyun.lb.port_$container_port).

**Application scenario:**

In Layer-4 load balancing, you can define a route for each service. When a traditional architecture is migrated to a container architecture, the services of non-container clusters access the services of containers deployed in clusters.

**Example:**

Use a custom Server Load Balancer instance to expose the Redis service in a container cluster to the Python project outside the container cluster.

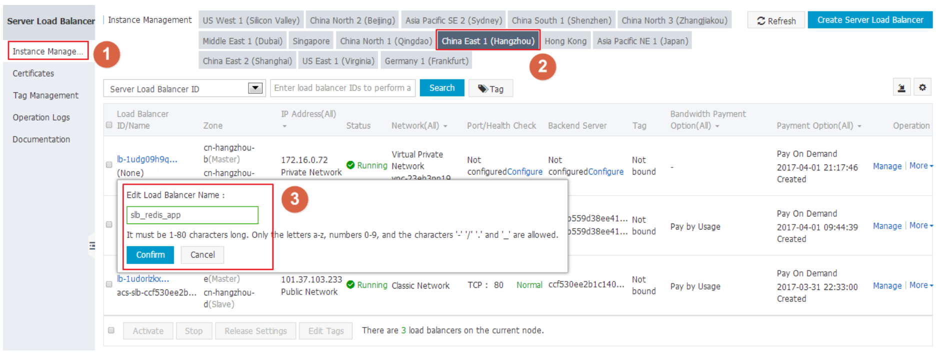
On the Server Load Balancer console, buy and create a Server Load Balancer instance used for routing (click **Create Server Load Balancer** in the upper-right corner).

An Internet-based instance is used as an example. You can select an Internet-based or intranet-based instance according to your needs.

**Note:** As Server Load Balancer does not support cross-region deployment, you should select the same region as the Container Service cluster that you intend to use.

Return to the Server Load Balancer console and name the created Server Load Balancer instance as **slb_redis_app**. This name is used by the Container Service to reference the instance.

Click **Instance Management** in the left navigation pane > select the region of the instance > select the target instance > edit the instance name and click **Confirm**.



| Load Balancer ID/Name | Zone | IP Address(All) | Status | Network(All) | Port/Health Check | Backend Server | Tag | Bandwidth Payment Option(All) | Payment Option(All) | Operation |
|---|---|---|---|---|---|---|---|---|---|---|
| lb-1udg09h9q... (None) | cn-hangzhou-b(Master) cn-hangzhou- | 172.16.0.72 Private Network | Running | Virtual Private Network | Not configuredConfigure | Not configuredConfigure | Not bound | - | Pay On Demand 2017-04-01 21:17:46 Created | Manage \| More |
| | | | | | | b559d38ee41... b559d38ee41... | Not bound | Pay by Usage | Pay On Demand 2017-04-01 09:44:39 Created | Manage \| More |
| lb-1udorlzkx... acs-slb-ccf530ee2b... | e(Master) cn-hangzhou-d(Slave) | 101.37.103.233 Public Network | Running | Classic Network | TCP : 80 Normal | ccf530ee2b1c140... | Not bound | Pay by Usage | Pay On Demand 2017-03-31 22:33:00 Created | Manage \| More |

Edit Load Balancer Name :

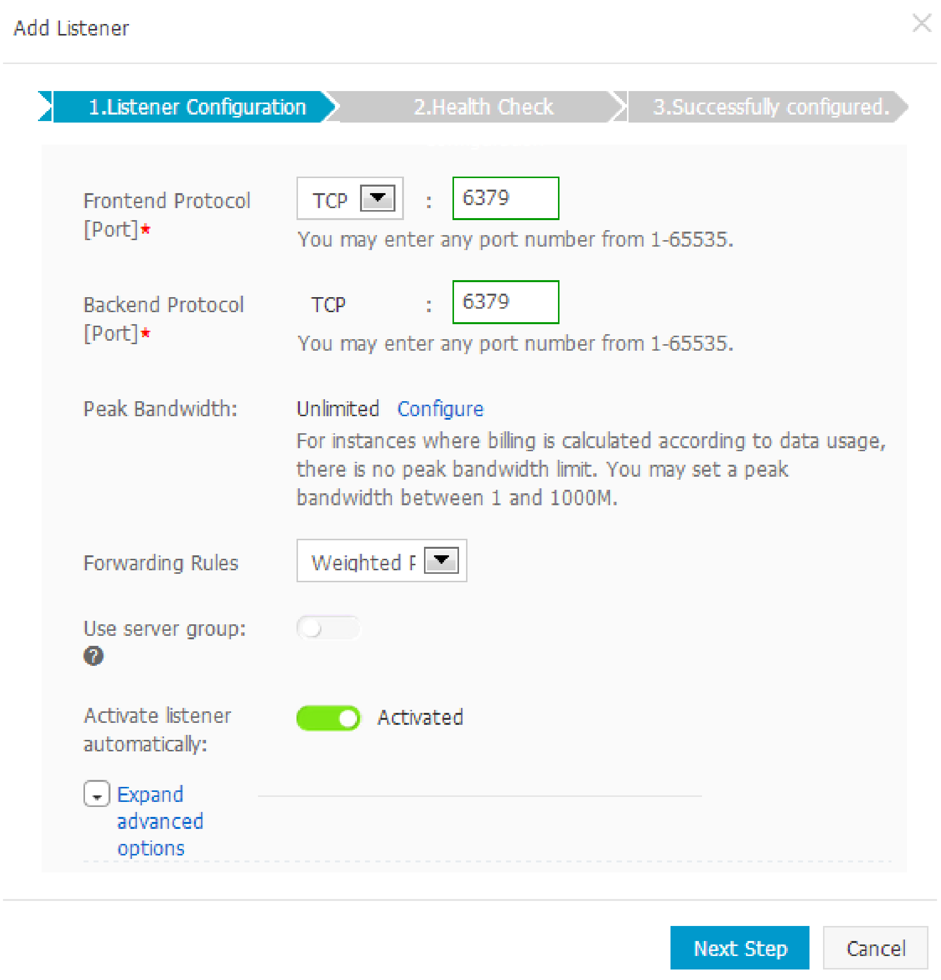
slb_redis_app

It must be 1-80 characters long. Only the letters a-z, numbers 0-9, and the characters '-' '/' '.' and '_' are allowed.

There are 3 load balancers on the current node.

Create a listening port.

Click **Manage** at the right of the instance > click **Listening** in the left navigation pane > click **Create Listener** > set the configurations. The listening port must be a TCP port with 6379:6379 mapping.



Frontend Protocol [Port]* TCP : 6379
You may enter any port number from 1-65535.

Backend Protocol [Port]* TCP : 6379
You may enter any port number from 1-65535.

Peak Bandwidth: Unlimited
For instances where billing is calculated according to data usage, there is no peak bandwidth limit. You may set a peak bandwidth between 1 and 1000M.

Forwarding Rules: Weighted F

Use server group:

Activate listener automatically: Activated

Log on to the Container Service console. Select an existing cluster, create an application named **redis-demo**, and click **Create with Image**.

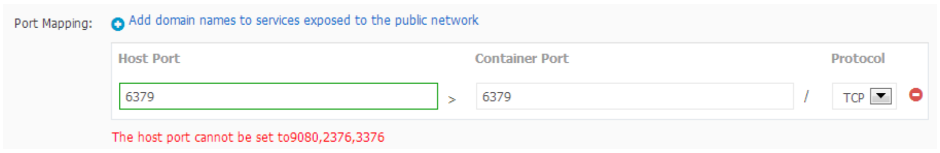
For information about how to create an application, refer to Create an application.

**Note:** As Server Load Balancer does not support cross-region deployment, the Container Service cluster should be in the same region as the Server Load Balancer instance you created.

Select the Redis image and set the **Port Mapping** information.

**Note:** The Redis image only enables Port 6379 on the container. To add a route destined for Port 6379 to the created Server Load Balancer instance, you need to obtain the host:port mapping of the Redis image.

To do this, in **Port Mapping**, specify **Host Port** as 6379 which is the backend port bound to the Server Load Balancer instance and specify **Protocol** as TCP.

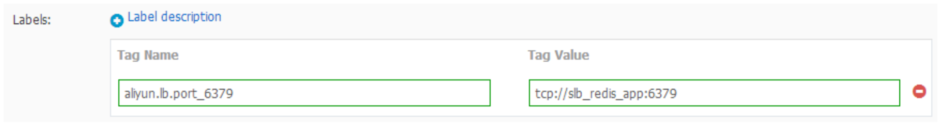

Port Mapping: Add domain names to services exposed to the public network

| Host Port | | Container Port | | Protocol |
|---|---|---|---|---|
| 6379 | > | 6379 | / | TCP |

The host port cannot be set to9080,2376,3376

To configure custom load balancing, you need to inject the Server Load Balancer information to the Redis service either by adding a label or by defining **Load Balancer**.

Add a label. In this example, the label is aliyun.lb.port_6379: tcp://slb_redis_app:6379.


Labels: Label description

| Tag Name | Tag Value |
|---|---|
| aliyun.lb.port_6379 | tcp://slb_redis_app:6379 |

The label format is shown as follows (the variable with $ is a placeholder).

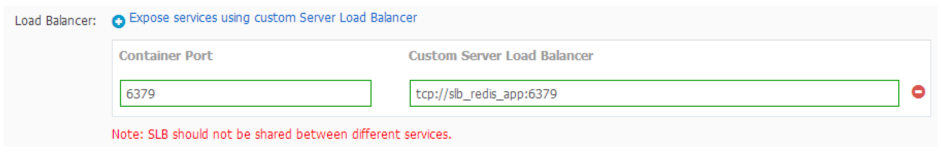
```
aliyun.lb.port_$container_port:$scheme://$[slb_name|slb_id]:$front_port
```

- $container_port indicates the port exposed by the container.
- $scheme indicates the protocol supported by the listening port of the Server Load Balancer instance, which may 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 exposed by the Server Load Balancer instance.

For more details, refer to aliyun.lb.port_$container_port.

In the **Create Application** page, click the plus icon at the right of **Load Balancer** and set the Server Load Balancer information.

This setting is equivalent to the label aliyun.lb.port_6379: tcp://slb_redis_app:6379.



Load Balancer: Expose services using custom Server Load Balancer

| Container Port | Custom Server Load Balancer |
|---|---|
| 6379 | tcp://slb_redis_app:6379 |

Note: SLB should not be shared between different services.

This example sets the destination container port to Port 6379, references the Server Load Balancer instance named **slb_redis_app**, sets the protocol used by the listening port to TCP corresponding to the protocol used by the host:container port mapping, and sets the frontend port of the Server Load Balancer instance to Port 6379.

> **Note:** In this example, the frontend port and backend port (host port) of the Server Load Balancer instance as well as the container port are all set to Port 6379, you can set a different frontend port and host port based on your needs.

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.

When the Redis application is ready, log on to the Server Load Balancer console to view the status of the **slb_redis_app** Server Load Balancer instance.

Click **Instance Management** in the left navigation pane > select the region of the instance > select the target instance > click **Manage** at the right of the instance > click **Server** > **Backend server**)

The health status shows that the Server Load Balancer instance is correctly bound to the backend host.

You can view the IP address of the Server Load Balancer instance in the **Instance Management** page of the Server Load Balancer console, and use the command line tool telnet $Server_Load_Balancer_IP_address 6379 to check port accessibility.

To test the configurations, start a simple Python project locally to access Redis in the container cluster through the **slb_redis_app** Server Load Balancer instance.

> **Note:** The Redis host address is the same as 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)
```

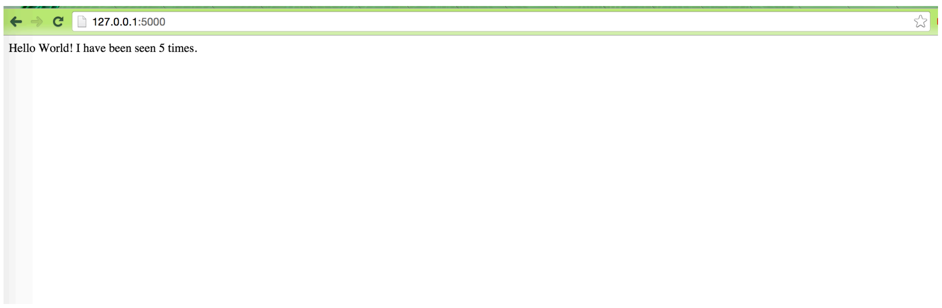
### requirements.txt

```
 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.



Hello World! I have been seen 5 times.

# Service discovery between containers

The Container Service provides several methods of service discovery for the services and containers within the cluster: by container name, by link, and by hostname.

## By container name

The Container Service can be accessed either through the container’ s IP address or the name of another container in the network. In the example described in the Container network interconnection , the service is accessed using the container name cross-host-network-test1 in the cross-host-network-test2 container.

If the container_name is not specified in the orchestration file, the container name by default is {project-name}_{service-name}_{container-index}. Therefore, the service can be accessed using the container name of another service after connecting to the management terminal.

## By link

The Container Service supports the link between services in application template. The link between services can link the container of one service to the container of another service, and a particular container can also access the container it depends on using the alias of the linked container. In addition, any change in the IP address of a container depended on by other containers can be dynamically updated to the alias. For example, the WordPress orchestration can be referenced in the Container Service orchestration instance. When the Web service in WordPress links db:mysql service to the container, the container can access the container of the db service using the MySQL domain.

## By hostname

If hostname configuration is defined in the orchestration template, the container can be accessed 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 for this service with xxserver, and if this service contains several containers, using the access with this domain allows Server Load Balancer to take effect.

If the service is not configured with a hostname, the Container Service will by default take the container name as its hostname. If an application needs to know its own container name for service registration, such as Eureka Client, a reachable address needs to be registered to the Eureka Server. The process in the container can obtain the container name for service registration and enable other service callers to access each other using this container name.

# Custom routing-Using guide

The custom proxy image is inherited from the image dockercloud/haproxy through FROM dockercloud/haproxy. It dynamically detects the container status, and realizes backend container load balancing proxy and service discovery. The feature is that all configurations of the HAProxy Server Load Balancer are parameterized for you to conveniently customize your configurations according to your need.

This image is mainly applicable in scenarios in which the default routing service of Alibaba Cloud Container Service cannot meet your needs. It helps you customize the HAProxy in a convenient way.

The acs/proxy and HAProxy mentioned in this document refer to this image and the HAProxy of this image respectively.

## Working principle of dynamic load balancing proxy and service discovery

Based on the container’ s environment variables, the image acs/proxy determines the global and default load balancing configurations.

The image acs/proxy listens to events in the cluster, such as changes to the container status, and re-obtains information about containers to which changes have occurred, so as to determine the latest load balancing configuration.

The image acs/proxy reloads the latest load balancing configuration to bring the configuration into effect.

## How to determine the backend container of Server Load Balancer

The range is determined based on the environment variable ADDITIONAL_SERVICES of acs/proxy.

- ADDITIONAL_SERVICES: "*" indicates that the range is the whole cluster.
- ADDITIONAL_SERVICES: "project_name1:service_name1,project_name2:service_name2" indicates that the range is the current application and specified services in the specified application.
- If ADDITIONAL_SERVICES is null or left blank, the range is the containers of the current application.

The image determines whether to add the container as acs/proxy backend container based on the container’ s label.

- aliyun.proxy.VIRTUAL_HOST: "www.toolchainx.com" indicates adding the container as backend container and the domain name is www.toolchainx.com.
- aliyun.proxy.required: "true" indicates adding the container as backend container and use it as the default backend container.

## How to bind a Server Load Balancer instance to the front-end

You can use a custom load balancing label, such as aliyun.lb.port_80: 'tcp://proxy:80'.

For details about how to use the custom load balancing label, refer to lb.

## Sample template

```
 lb:
image: registry.aliyuncs.com/acs/proxy:0.5
ports:
- '80:80'
restart: always
labels:
# With the use of addon, the proxy image can function as a subscription registry center and dynamically load the service route.
aliyun.custom_addon: "proxy"
# A proxy image container is deployed for each VM.
aliyun.global: "true"
# Use a custom load balancing label to bind a Server Load Balancer instance to the front-end.
aliyun.lb.port_80: tcp://proxy_test:80
environment:
# It indicates the range of backend containers that support route loading: "*" indicates the whole cluster. The default value is services in an application.
ADDITIONAL_SERVICES: "*"
appone:
expose: # For proxied services, use expose or ports to tell proxy containers which port should be exposed.
- 80/tcp
image: 'nginx:latest'
labels:
# HTTP, HTTPS, WS, and WSS are available in this example.
# Use your own domain name instead of the domain name for testing provided by the Container Service.
aliyun.proxy.VIRTUAL_HOST: "http://appone.example.com"
restart: always
```

## Configuration instructions

### Set the global and default configurations through the environment variables of

## the image acs/proxy

**Note:** Settings in this part is immutable, you have to redeploy HAProxy service to make the changes take effects.

| Environment Variable | Default | Description |
|---|---|---|
| ADDITIONAL_SERVICES | | List of additional services to balance (for example, prj1:web,prj2:sql).<br>Discovery will be based on com.docker.compose.[project&#124;service] container labels.<br>This environment variable only works on compose V2, and the referenced services must be on a network resolvable and accessible to this containers. |
| BALANCE | roundrobin | Load balancing algorithm to use.<br>Possible values include: roundrobin, static-rr, source, leastconn.<br>See HAProxy:balance. |
| CA_CERT_FILE | | Path of a ca-cert file.<br>This allows you to mount your ca-cert file directly from a volume instead of from environment variable.<br>If set, CA_CERT environment variable will be ignored.<br>Possible value: /cacerts/cert0.pem |
| CA_CERT | | CA cert for haproxy to verify the client.<br>Use the same format as DEFAULT_SSL_CERT. |
| CERT_FOLDER | | Path of certificates.<br>This allows you to mount your certificate files directly from a volume instead of from environment variables.<br>If set, DEFAULT_SSL_CERT and SSL_CERT from linked services are ignored.<br>Possible value: /certs/. |
| DEFAULT_SSL_CERT | | Default SSL certificate.<br>A pem file content with private key followed by public certificate, '\n' (two |

| | | |
|---|---|---|
| | | chars) as the line separator. The content should be formatted as one line. See SSL Termination. |
| EXTRA_BIND_SETTINGS | | Comma-separated string (<port>:<setting>) of extra settings, and each part will be appended to the related port bind section in the configuration file.<br>To escape comma, use \,.<br>Possible value: 443:accept-proxy, 80:name http. |
| EXTRA_DEFAULT_SETTINGS | | Comma-separated string of extra settings, and each part will be appended to DEFAULT section in the configuration file.<br>To escape comma, use \,. |
| EXTRA*FRONTEND_SETTINGS* \<PORT> | | Comma-separated string of extra settings, and each part will be appended frontend section with the port number specified in the name of the environment variable.<br>To escape comma, use \,.<br>For example, EXTRA_FRONTEND_SETTINGS_80=balance source, maxconn 2000. |
| EXTRA_GLOBAL_SETTINGS | | Comma-separated string of extra settings, and each part will be appended to GLOBAL section in the configuration file.<br>To escape comma, use \,.<br>Possible value: tune.ssl.cachesize 20000, tune.ssl.default-dh-param 2048. |
| EXTRA_ROUTE_SETTINGS | | A string which is appended to each backend route after the health check, and can be overwritten in the linked services.<br>Possible value: “send-proxy”. |
| EXTRA_SSL_CERTS | | List of extra certificate names separated by comma, for example, CERT1, CERT2, CERT3.<br>You also need to specify each certificate as a separate environment variable like so: |

| | | |
|---|---|---|
| | | CERT1="<cert-body1>", CERT2="<cert-body2>", CERT3="<cert-body3>". |
| HEALTH_CHECK | check | Sets health check on each backend route.<br>Possible value: “check inter 2000 rise 2 fall 3” .<br>See: HAProxy:check. |
| HTTP_BASIC_AUTH | | A comma-separated list of credentials (<user>:<pass>) for HTTP basic authentication, which applies to all the backend routes.<br>To escape comma, use \,.<br>**Note:** Do not rely on this for authentication in production. |
| MAXCONN | 4096 | Sets the maximum per-process number of concurrent connections. |
| MODE | http | Mode of load balancing for HAProxy.<br>Possible values include: http, tcp, health. |
| MONITOR_PORT | | Port number where monitor_uri should be added to.<br>Use together with MONTIOR_URI.<br>Possible value: 80. |
| MONITOR_URI | | The exact URI which we want to intercept to return HAProxy’ s health status instead of forwarding the request.<br>See: http://cbonte.github.io/haproxy-dconv/configuration-1.5.html#4-monitor-uri.<br>Possible value: /ping. |
| OPTION | redispatch | Comma-separated list of HAProxy option entries to the default section. |
| RSYSLOG_DESTINATION | 127.0.0.1 | The rsyslog destination to where HAProxy logs are sent. |
| SKIP_FORWARDED_PROTO | | If set to any value, HAProxy will not add an X-Forwarded-headers.<br>This can be used when combining HAProxy with another load balancer. |
| SSL_BIND_CIPHERS | | Explicitly sets which SSL |

| | | |
|---|---|---|
| | | ciphers will be used for the SSL server.<br>This sets the HAProxy ssl-default-bind-ciphers configuration setting. |
| SSL_BIND_OPTIONS | no-sslv3 | Explicitly sets which SSL bind options will be used for the SSL server.<br>This sets the HAProxy ssl-default-bind-options configuration setting.<br>The default will allow only TLSv1.0+ to be used on the SSL server. |
| STATS_AUTH | stats:stats | Username and password required to access the Haproxy stats. |
| STATS_PORT | 1936 | Port for the HAProxy stats section.<br>If this port is published, stats can be accessed at http://<host-ip>:<STATS_PORT>/. |
| TIMEOUT | connect 5000, client 50000, server 50000 | Comma-separated list of HAProxy timeout entries to the default section. |

## Set the backend service configurations by using the the corresponding service image labels

Configure backend service by adding labels to the backend service image. These configurations are written to the template section of the proxied service.

Settings here can overwrite the settings in HAProxy, which are only applied to the linked services. If run in Docker Cloud, when the service redeploys, joins or leaves HAProxy service, HAProxy service will automatically update itself to apply the changes.

| Labels | Description |
|---|---|
| aliyun.proxy.APPSESSION | Sticky session option.<br>Possible value JSESSIONID len 52 timeout 3h.<br>See: HAProxy:appsession. |
| aliyun.proxy.BALANCE | Load balancing algorithm to use.<br>Possible values include: roundrobin, static-rr, source, leastconn.<br>See:HAProxy:balance. |
| aliyun.proxy.COOKIE | Sticky session option.<br>Possible value: SRV insert indirect nocache.<br>See: HAProxy:cookie. |
| aliyun.proxy.DEFAULT_SSL_CERT | Similar to SSL_CERT, but stores the pem file at |

| | |
|---|---|
| | /certs/cert0.pem as the default SSL certificates.<br>If multiple DEFAULT_SSL_CERT are specified in linked services and HAProxy, the behavior is undefined. |
| aliyun.proxy.EXCLUDE_PORTS | Comma-separated port numbers (for example, 3306, 3307).<br>By default, HAProxy will add all the ports exposed by the application services to the backend routes.<br>You can exclude the ports that you don’t want to be routed, like database port. |
| aliyun.proxy.EXTRA_ROUTE_SETTINGS | A string which is appended to each backend route after the health check.<br>Possible value: “send-proxy”. |
| aliyun.proxy.EXTRA_SETTINGS | Comma-separated string of extra settings, and each part will be appended to either related backend section or listen session in the configuration file.<br>To escape comma, use \,.<br>Possible value: balance source. |
| aliyun.proxy.FORCE_SSL | If set(any value) together with SSL termination enabled, HAProxy will redirect HTTP request to HTTPS request. |
| aliyun.proxy.GZIP_COMPRESSION_TYPE | Enables gzip compression.<br>The value of this environment variable is a list of MIME types that will be compressed.<br>Possible value: text/html text/plain text/css. |
| aliyun.proxy.HEALTH_CHECK | Set health check on each backend route.<br>Possible value: “check inter 2000 rise 2 fall 3”.<br>See: HAProxy:check. |
| aliyun.proxy.HSTS_MAX_AGE | Enables HSTS.<br>It is an integer representing the max age of HSTS in seconds.<br>Possible value: 31536000. |
| aliyun.proxy.HTTP_CHECK | Enables HTTP protocol to check on the servers health.<br>Possible value: “OPTIONS * HTTP/1.1\r\nHost:\ www”.<br>See: HAProxy:httpchk. |
| aliyun.proxy.OPTION | Comma-separated list of HAProxy option entries.<br>option specified here will be added to related backend or listen part, and overwrite the OPTION settings in the HAProxy container. |
| aliyun.proxy.SSL_CERT | SSL certificate.<br>A pem file with private key followed by public certificate, ‘\n’(two chars) as the line separator. |

| | |
|---|---|
| aliyun.proxy.TCP_PORTS | Comma-separated ports (for example, 9000, 9001, 2222/ssl).<br>The port listed in TCP_PORTS will be load-balanced in TCP mode.<br>Port ends with /ssl indicates that port needs SSL termination. |
| aliyun.proxy.VIRTUAL_HOST_WEIGHT | An integer of the weight of an virtual host, used together with VIRTUAL_HOST.<br>The default value is 0.<br>It affects the order of ACL rules of the virtual hosts. The higher weight one virtual host has, the more priority that ACL rules applies. |
| aliyun.proxy.VIRTUAL_HOST | Specify virtual host and virtual path.<br>Format: [scheme://]domain[:port][/path], ....<br>Wildcard * can be used in domain and path part. |

Check the HAProxy configuration manual for more information on the above.

## Virtual host and virtual path

Both virtual host and virtual path can be specified in environment variable VIRTUAL_HOST, which is a set of comma-separated URLs with the format of [scheme://]domain[:port][/path].

| Item | Default | Description |
|---|---|---|
| scheme | http | Possible values: http, https, wss. |
| domain | | Virtual host.<br>* can be used as the wildcard. |
| port | 80/433 | Port number of the virtual host.<br>When the scheme is https wss, the default port will be to 443. |
| /path | | Virtual path, starts with /.<br>* can be used as the wildcard. |

### Examples of matching

| Virtual host | Match | Not match |
|---|---|---|
| http://domain.com | domain.com | www.domain.com |
| domain.com | domain.com | www.domain.com |
| domain.com:90 | domain.com:90 | domain.com |
| https://domain.com | https://domain.com | domain.com |

| | | |
|---|---|---|
| https://domain.com:444 | https://domain.com:444 | https://domain.com |
| *.domain.com | www.domain.com | domain.com |
| *domain.com | www.domain.com, domain.com, anotherdomain.com | www.abc.com |
| www.e*e.com | www.domain.com, www.exxe.com | www.axxa.com |
| www.domain.* | www.domain.com, www.domain.org | domain.com |
| * | any website with HTTP | |
| https://* | any website with HTTPS | |
| */path | domain.com/path, domain.org/path?u=user | domain.com/path/ |
| */path/ | domain.com/path/, domain.org/path/?u=user | domain.com/path, domain.com/path/abc |
| */path/* | domain.com/path/, domain.org/path/abc | domain.com/abc/path/ |
| */*/path/* | domain.com/path/, domain.org/abc/path/, domain.net/abc/path/123 | domain.com/path |
| */*.js | domain.com/abc.js, domain.org/path/abc.js | domain.com/abc.css |
| */*.do/ | domain.com/abc.do/, domain.org/path/abc.do/ | domain.com/abc.do |
| */path/*.php | domain.com/path/abc.php | domain/abc.php, domain.com/root/abc.php |
| *.domain.com/*.jpg | www.domain.com/abc.jpg, abc.domain.com/123.jpg | domain.com/abc.jpg |
| */path, */path/ | domain.com/path, domain.org/path/ | |
| domain.com:90, https://domain.com | domain.com:90, https://domain.com | |

**Note**:

- The sequence of the ACL rules generated based on VIRTUAL_HOST are random. In HAProxy, when an ACL rule with a wide scope (for example, *.domain.com) is put before a rule with narrow scope (for example, web.domain.com), the narrow rule will never be reached. As a result, if the virtual hosts you set have overlapping scopes, you need to use VIRTUAL_HOST_WEIGHT to manually set the order of acl rules, namely, giving the narrow virtual host a higher weight than the wide one.
- Every service that has the same VIRTUAL_HOST environment variable setting will be

considered and merged into one single service. It may be useful for some testing scenario.

## SSL termination

acs/proxy supports SSL termination on multiple certificates. For each application that you want SSL terminates, simply set SSL_CERT and VIRTUAL_HOST. HAProxy, then, reads the certificate from the link environment and sets the SSL termination up.

**Note:** There was a bug that if an environment variable value contains “=”, which is common in the SSL_CERT, Docker skips that environment variable. As a result, multiple SSL termination only works on Docker 1.7.0 or higher.

SSL termination is enabled when:

- At least one SSL certificate is set.
- Either VIRTUAL_HOST is not set, or it is set with https as the scheme.

To set SSL certificate, you can either:

- Set DEFAULT_SSL_CERT in acs/proxy.
- Set aliyun.proxy.SSL_CERT and/or DEFAULT_SSL_CERT in the application services linked to HAProxy.

The difference between aliyun.proxy.SSL_CERT and DEFAULT_SSL_CERT is that, the multiple certificates specified by SSL_CERT are stored in as cert1.pem, cert2.pem, ..., whereas the one specified by DEFAULT_SSL_CERT is always stored as cert0.pem. In that case, HAProxy will use cert0.pem as the default certificate when there is no SNI match. However, when multiple DEFAULT_SSL_CERTIFICATE is provided, only one of the certificates can be stored as cert0.pem, others are discarded.

### PEM Files

The certificate specified in acs/proxy or in the linked application services is a pem file, containing a private key followed by a public certificate (private key must be put before the public certificate and any extra Authority certificates, order matters). You can run the following script to generate a self-signed certificate.

```
openssl req -x509 -newkey rsa:2048 -keyout key.pem -out ca.pem -days 1080 -nodes -subj '/CN=*/O=My Company Name LTD./C=US'
cp key.pem cert.pem
cat ca.pem >> cert.pem
```

Once you have the pem file, you can run this command to convert the file correctly to one line.

```
awk 1 ORS='\\n' cert.pem
```

Copy the output and set it as the value of aliyun.proxy.SSL_CERT or DEFAULT_SSL_CERT.

## Affinity and session stickiness

There are three methods to set up affinity and sticky session.

- Set aliyun.proxy.BALANCE=source in your application service. When setting source method of balance, HAProxy will hash the client IP address and make sure that the same IP always goes to the same server.
- Set aliyun.proxy.APPSESSION=<value>. Use application session to determine which server a client should connect to. Possible value of <value> could be JSESSIONID len 52 timeout 3h.
- Set aliyun.proxy.COOKIE=<value>. Use application cookie to determine which server a client should connect to. Possible value of <value> could be SRV insert indirect nocache.

Check HAProxy:appsession and HAProxy:cookie for more information.

## TCP load balancing

By default, acs/proxy runs in http mode. If you want a linked service to run in a tcp mode, you can specify the environment variable TCP_PORTS, which is a comma-separated ports (for example, 9000, 9001).

For example, if you run:

```
docker --name app-1 --expose 9000 --expose 9001 -e TCP_PORTS="9000, 9001" your_app
docker --name app-2 --expose 9000 --expose 9001 -e TCP_PORTS="9000, 9001" your_app
docker run --link app-1:app-1 --link app-2:app-2 -p 9000:9000, 9001:9001 acs/proxy
```

Then, Haproxy balances the load between app-1 and app-2 in both port 9000 and 9001 respectively.

Moreover, If you have more exposed ports than TCP_PORTS, the rest of the ports will be balancing using http mode.

For example, if you run:

```
docker --name app-1 --expose 80 --expose 22 -e TCP_PORTS=22 your_app
docker --name app-2 --expose 80 --expose 22 -e TCP_PORTS=22 your_app
docker run --link app-1:app-2 --link app-2:app-2 -p 80:80 -p 22:22 acs/proxy
```

Then, Haproxy balances in http mode at port 80 and balances in tcp on port at port 22.

In this way, you can do the load balancing both in tcp and in http at the same time.

In TCP_PORTS, if you set port that ends with /ssl, for example 2222/ssl, HAProxy will set SSL termination on port 2222.

**Note:**

- You are able to set VIRTUAL_HOST and TCP_PORTS at the same time, giving more control on

http mode.
- Be careful that, the load balancing on tcp port is applied to all the services. If you link two (or more) different services using the same TCP_PORTS, acs/proxy considers them coming from the same service.

## WebSocket support

There are two ways to enable the support of websocket.

- As websocket starts using HTTP protocol, you can use virtual host to specify the scheme using ws or wss. For example, `-e VIRTUAL_HOST=”ws://ws.domain.com, wss://wss.domain.com”.
- Websocket itself is a TCP connection, you can also try the TCP load balancing mentioned in the previous section.

## Use case scenarios

### My webapp container exposes port 8080 (or any other port), and I want the proxy to listen in port 80

Use the following:

```
docker run -d --expose 80 --name webapp dockercloud/hello-world
docker run -d --link webapp:webapp -p 80:80 acs/proxy
```

### My webapp container exposes port 80 and database ports 8083/8086, and I want the proxy to listen in port 80 without my database ports added to haproxy

```
docker run -d -e EXCLUDE_PORTS=8803,8806 --expose 80 --expose 8033 --expose 8086 --name webapp
dockercloud/hello-world
docker run -d --link webapp:webapp -p 80:80 acs/proxy
```

### My webapp container exposes port 8080 (or any other port), and I want the proxy to listen in port 8080

Use the following:

```
docker run -d --expose 8080 --name webapp your_app
docker run -d --link webapp:webapp -p 8080:80 acs/proxy
```

### I want the proxy to terminate SSL connections and forward plain HTTP requests to my webapp to port 8080 (or any port)

Use the following:

```
docker run -d -e SSL_CERT="YOUR_CERT_TEXT" --name webapp dockercloud/hello-world
docker run -d --link webapp:webapp -p 443:443 -p 80:80 acs/proxy
```

or

```
docker run -d --link webapp:webapp -p 443:443 -p 80:80 -e DEFAULT_SSL_CERT="YOUR_CERT_TEXT" acs/proxy
```

The certificate in YOUR_CERT_TEXT is a combination of private key followed by public certificate. Remember to put \n between each line of the certificate. A way to do this, assuming that your certificate is stored in ~/cert.pem, is running the following:

```
docker run -d --link webapp:webapp -p 443:443 -p 80:80 -e DEFAULT_SSL_CERT="$(awk 1 ORS='\\n' ~/cert.pem)"
acs/proxy
```

### I want the proxy to terminate SSL connections and redirect HTTP requests to HTTPS

Use the following:

```
docker run -d -e FORCE_SSL=yes -e SSL_CERT="YOUR_CERT_TEXT" --name webapp dockercloud/hello-world
docker run -d --link webapp:webapp -p 443:443 acs/proxy
```

### I want to load my SSL certificate from volume instead of passing it through environment variable

You can use CERT_FOLDER environment variable to specify which folder the certificates are mounted in the container, using the following:

```
docker run -d --name webapp dockercloud/hello-world
docker run -d --link webapp:webapp -e CERT_FOLDER="/certs/" -v $(pwd)/cert1.pem:/certs/cert1.pem -p 443:443
acs/proxy
```

### I want to set up virtual host routing by domain

Virtual hosts can be configured by the proxy reading linked container environment variables (VIRTUAL_HOST). Here is an example:

```
docker run -d -e VIRTUAL_HOST="www.webapp1.com, www.webapp1.org" --name webapp1 dockercloud/hello-
world
docker run -d -e VIRTUAL_HOST=www.webapp2.com --name webapp2 your/webapp2
docker run -d --link webapp1:webapp1 --link webapp2:webapp2 -p 80:80 acs/proxy
```

In the example above, when you access http://www.webapp1.com or http://www.webapp1.org, it will show the service running in container webapp1, and http://www.webapp2.com will go to container webapp2.

If you use the following:

```
docker run -d -e VIRTUAL_HOST=www.webapp1.com --name webapp1 dockercloud/hello-world
docker run -d -e VIRTUAL_HOST=www.webapp2.com --name webapp2-1 dockercloud/hello-world
docker run -d -e VIRTUAL_HOST=www.webapp2.com --name webapp2-2 dockercloud/hello-world
docker run -d --link webapp1:webapp1 --link webapp2-1:webapp2-1 --link webapp2-2:webapp2-2 -p 80:80
acs/proxy
```

When you access http://www.webapp1.com, it will show the service running in container webapp1, and http://www.webapp2.com will go to both containers webapp2-1 and webapp2-2 using round robin (or whatever is configured in BALANCE).

## I want all my *.node.io domains point to my service

```
docker run -d -e VIRTUAL_HOST="*.node.io" --name webapp dockercloud/hello-world
docker run -d --link webapp:webapp -p 80:80 acs/proxy
```

## I want web.domain.com go to one service and *.domain.com go to another service

```
docker run -d -e VIRTUAL_HOST="web.domain.com" -e VIRTUAL_HOST_WEIGHT=1 --name webapp
dockercloud/hello-world
docker run -d -e VIRTUAL_HOST="*.domain.com" -e VIRTUAL_HOST_WEIGHT=0 --name app dockercloud/hello-
world
docker run -d --link webapp:webapp --link app:app -p 80:80 acs/proxy
```

## I want all the requests to path /path point to my service

```
docker run -d -e VIRTUAL_HOST="*/path, */path/*" --name webapp dockercloud/hello-world
docker run -d --link webapp:webapp -p 80:80 acs/proxy
```

## I want all the static HTML request point to my service

```
docker run -d -e VIRTUAL_HOST="*/*.htm, */*.html" --name webapp dockercloud/hello-world
docker run -d --link webapp:webapp -p 80:80 acs/proxy
```

## I want to see stats of HAProxy

```
docker run -d --link webapp:webapp -e STATS_AUTH="auth:auth" -e STATS_PORT=1936 -p 80:80 -p 1936:1936
acs/proxy
```

### I want to send all my logs to papertrailapp

Replace <subdomain> and <port> with your values matching your papertrailapp account:

```
docker run -d --name web1 dockercloud/hello-world
docker run -d --name web2 dockercloud/hello-world
docker run -it --env RSYSLOG_DESTINATION='<subdomain>.papertrailapp.com:<port>' -p 80:80 --link web1:web1
--link web2:web2 acs/proxy
```

## Topologies using virtual hosts

Alibaba Cloud Container Service service proxy topologies:

```
                                                        |---- container_a1
|----- service_a ----- |---- container_a2
| (virtual host a) |---- container_a3
internet --- SLB -- acs/proxy ----- |
| |---- container_b1
|----- service_b ----- |---- container_b2
(virtual host b) |---- container_b3
```

## Manually reload haproxy

In most cases, acs/proxy will configure itself automatically when the linked services change, you don’t need to reload it manually. But for some reason, if you have to do so, here is how:

docker exec <haproxy_id> /reload.sh, if you are on the node where acs/proxy deploys.

# 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 this example.

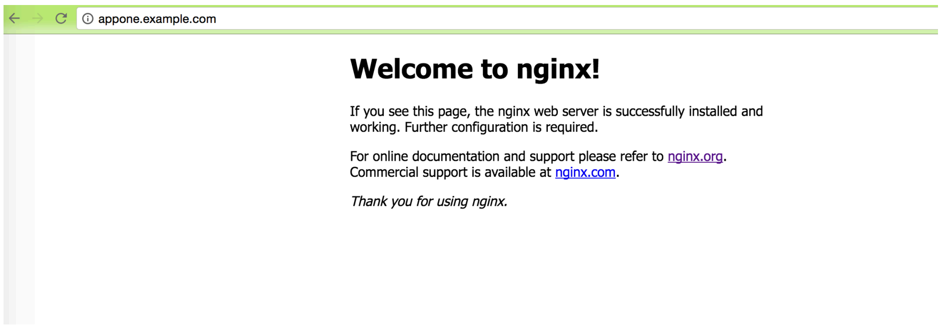
## Basic example

The compose template is shown as follows.

```
lb:
image: registry.aliyuncs.com/acs/proxy:0.5
ports:
```

```
- '80:80'
restart: always
labels:
# With the use of Addon, the proxy image can function as a subscription registry center and dynamically load the service route.
aliyun.custom_addon: "proxy"
# A proxy image container is deployed for each VM.
aliyun.global: "true"
# A Server Load Balancer instance is bound to the front-end.
aliyun.lb.port_80: tcp://proxy_test:80
environment:
# It indicates the range of backend containers that support route loading: "*" indicates the whole cluster. By default, it indicates services in an application.
ADDITIONAL_SERVICES: "*"
appone:
expose: # For proxied services, use expose or ports to tell proxy containers which port should be exposed.
- 80/tcp
image: 'nginx:latest'
labels:
# http/https/ws/wss are available. Use your own domain name instead of the test domain name provided by the Container Service.
aliyun.proxy.VIRTUAL_HOST: "http://appone.example.com"
restart: always
```

The following figure shows the page that is displayed after a successful startup.



**Welcome to nginx!**

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

*Thank you for using nginx.*

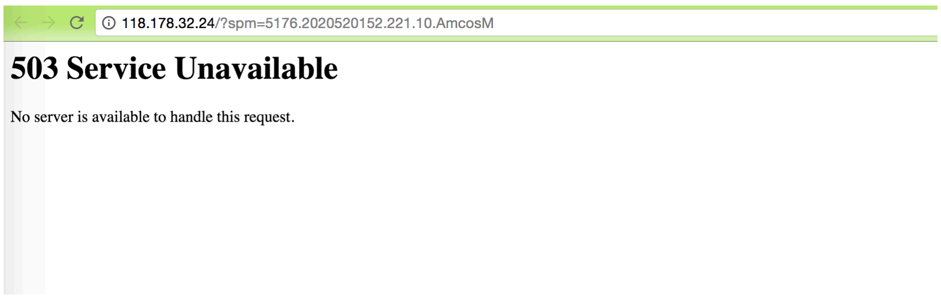
## Enable session persistence

```
lb:
image: registry.aliyuncs.com/acs/proxy:0.5
ports:
- '80:80'
restart: always
labels:
# With the use of Addon, the proxy image can function as a subscription registry center and dynamically load the service route.
aliyun.custom_addon: "proxy"
# A proxy image container is deployed for each VM.
aliyun.global: "true"
# A Server Load Balancer instance is bound to the front-end.
aliyun.lb.port_80: tcp://proxy_test:80
environment:
```

```
# It indicates the range of backend containers that support route loading: "*" indicates the whole cluster. By default, it indicates services in an application.
ADDITIONAL_SERVICES: "*"
appone:
ports:
- 80/tcp
- 443/tcp
image: 'nginx:latest'
labels:
# http/https/ws/wss are available.
aliyun.proxy.VIRTUAL_HOST: "http://appone.example.com"
# Session persistence is enabled, the cookie method is applied, and the key is CONTAINERID.
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 input, the 503 error page is returned, as shown in the following figure.



503 Service Unavailable

No server is available to handle this request.

If you want to add information to the 503 page, add the /errors folder to the VM where the container is located and add the /errors/503.http file with the following content.

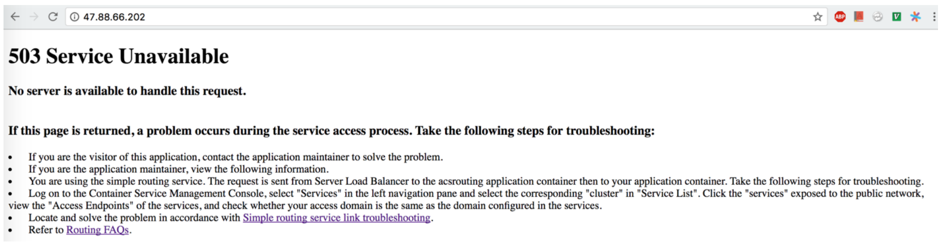
```
HTTP/1.0 503 Service Unavailable
Cache-Control: no-cache
Connection: close
Content-Type: text/html;charset=UTF-8

<html><body><h1>503 Service Unavailable</h1>

<h3>No server is available to handle this request.</h3>

<h3>If this page is returned, a problem occurs during the service access process. Take the following steps for troubleshooting:</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. Take the following steps for troubleshooting.</li>
<li>Log on to the Container Service Management Console, select "Services" in the left navigation pane and select the corresponding "cluster" in the "Service List". Click the "Services" exposed to the public network, view the "Access Endpoints" of the services, and check whether your access domain is the same as the domain configured in the services.</li>
<li>Locate and solve the problem in accordance with <a href="https://www.alibabacloud.com/help/faq-detail/42660.htm?spm=a3c0i.p39867en1.a3.14.idXIm7">Simple routing service link troubleshooting</a>.</li>
```

```
<li>Refer to <a href="https://www.alibabacloud.com/help/faq-
detail/42658.htm?spm=a3c0i.p39867en1.a3.12.pyONJ2">Routing FAQs</a>.</li>
</body></html>
```

You can modify the error page according to your need. The compose template is modified as follows:

```


lb:
image: registry.aliyuncs.com/acs/proxy:0.5
ports:
- '80:80'
restart: always
labels:
# With the use of Addon, the proxy image can function as a subscription registry center and dynamically load the
service route.
aliyun.custom_addon: "proxy"
# A proxy image container is deployed for each VM.
aliyun.global: "true"
# A Server Load Balancer instance is bound to the front-end.
aliyun.lb.port_80: tcp://proxy_test:80
environment:
# It indicates the range of backend containers that support route loading: "*" indicates the whole cluster. By default,
it indicates services in an application.
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 available.
aliyun.proxy.VIRTUAL_HOST: "http://appone.example.com"
restart: always
```

The following figure shows the 503 page that is returned after the VIP address of the Server Load Balancer instance is entered.



47.88.66.202

503 Service Unavailable

No server is available to handle this request.

If this page is returned, a problem occurs during the service access process. Take the following steps for troubleshooting:

- If you are the visitor of this application, contact the application maintainer to solve the problem.
- If you are the application maintainer, view the following information.
- 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. Take the following steps for troubleshooting.
- Log on to the Container Service Management Console, select "Services" in the left navigation pane and select the corresponding "cluster" in "Service List". Click the "services" exposed to the public network, view the "Access Endpoints" of the services, and check whether your access domain is the same as the domain configured in the services.
- Locate and solve the problem in accordance with Simple routing service link troubleshooting.
- Refer to Routing FAQs.

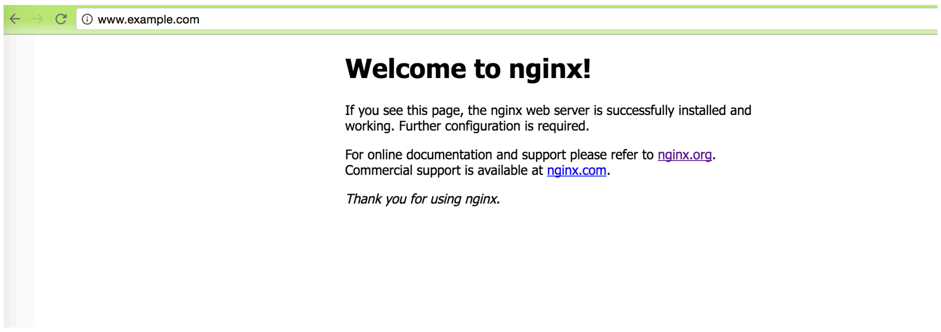
## Wildcard domain support

Modify the configuration as follows to enable the backend of Nginx to support wildcard domain names (that is, the Nginx homepage can be accessed through appone.example.com and

*.example.com).

```
lb:
image: registry.aliyuncs.com/acs/proxy:0.5
ports:
- '80:80'
restart: always
labels:
# With the use of Addon, the proxy image can function as a subscription registry center and dynamically load the service route.
aliyun.custom_addon: "proxy"
# A proxy image container is deployed for each VM.
aliyun.global: "true"
# A Server Load Balancer instance is bound to the front-end.
aliyun.lb.port_80: tcp://proxy_test:80
environment:
# It indicates the range of backend containers that support route loading: "*" indicates the whole cluster. By default, it indicates services in an application.
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 available.
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 shown in the following figure.



www.example.com

**Welcome to nginx!**

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

*Thank you for using nginx.*

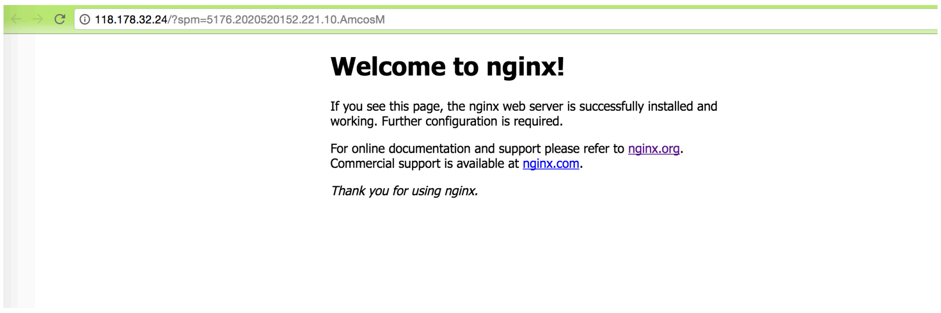
## Default backend configuration

Remove the URL configuration and modify the configuration as follows to enable access to Nginx at the backend through an IP address.

```
lb:
```

```
image: registry.aliyuncs.com/acs/proxy:0.5
ports:
- '80:80'
restart: always
labels:
# With the use of Addon, the proxy image can function as a subscription registry center and dynamically load the
service route.
aliyun.custom_addon: "proxy"
# A proxy image container is deployed for each VM.
aliyun.global: "true"
# A Server Load Balancer instance is bound to the front-end.
aliyun.lb.port_80: tcp://proxy_test:80
environment:
# It indicates the range of backend containers that support route loading: "*" indicates the whole cluster. By default,
it indicates services in an application.
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:
# It indicates that the service must be proxied.
aliyun.proxy.required: "true"
restart: always
```

The following figure shows the Nginx homepage after the VIP address of the Server Load Balancer instance is entered.


118.178.32.24/?spm=5176.2020520152.221.10.AmcosM

**Welcome to nginx!**

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

*Thank you for using nginx.*

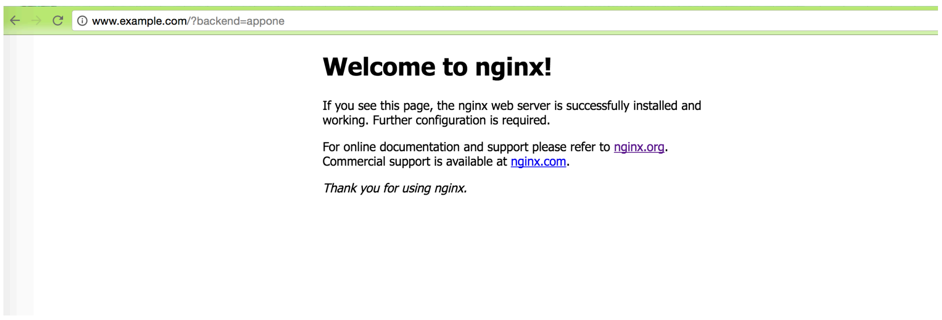
## Backend selection based on URL parameter values

You can use different backend proxies based on different URL parameter values.

The following example shows how to access the appone service, that is, Nginx homepage, through http://www.example.com?backend=appone and how to access the apptwo service, that is, hello world homepage, through http://www.example.com?backend=apptwo. The application template code is as follows:

```
lb:
image: registry.aliyuncs.com/acs/proxy:0.5
ports:
- '80:80'
restart: always
labels:
# With the use of Addon, the proxy image can function as a subscription registry center and dynamically load the service route.
aliyun.custom_addon: "proxy"
# A proxy image container is deployed for each VM.
aliyun.global: "true"
# A Server Load Balancer instance is bound to the front-end.
aliyun.lb.port_80: tcp://proxy_test:80
environment:
# It indicates the range of backend containers that support route loading: "*" indicates the whole cluster. By default, it indicates services in an application.
ADDITIONAL_SERVICES: "*"
# Obtain the value of the "backend" parameter in the URL and modify the HOST header 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 available.
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 available.
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 shown in the following figure.


www.example.com/?backend=appone

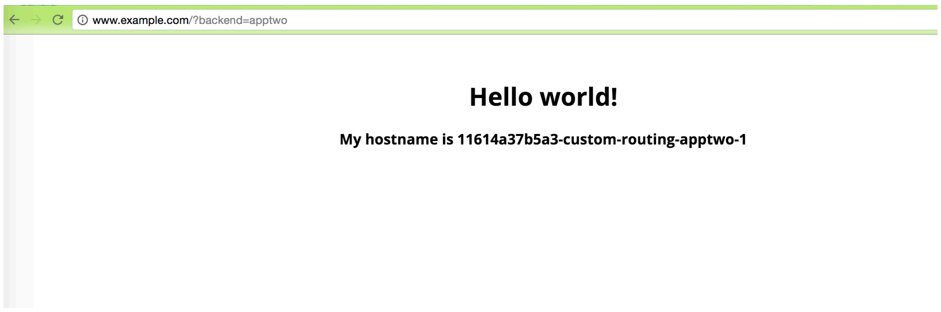
**Welcome to nginx!**

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

*Thank you for using nginx.*

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 shown in the following figure.



Hello world!

My hostname is 11614a37b5a3-custom-routing-apptwo-1

## Access log recording

```
lb:
image: registry.aliyuncs.com/acs/proxy:0.5
ports:
- '80:80'
restart: always
labels:
# With the use of Addon, the proxy image can function as a subscription registry center and dynamically load the service route.
aliyun.custom_addon: "proxy"
# A proxy image container is deployed for each VM.
aliyun.global: "true"
# A Server Load Balancer instance is bound to the front-end.
aliyun.lb.port_80: tcp://proxy_test:80
environment:
# It indicates the range of backend containers that support route loading: "*" indicates the whole cluster. By default, it indicates services in an application.
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 available.
aliyun.proxy.VIRTUAL_HOST: "http://appone.example.com"
restart: always
```

Logs are printed directly to the standard output of the rsyslog container. The access log of custom routes can be viewed using docker logs $rsyslog_container_name.

## Server Load Balancer between services

The following template is used to create a Server Load Balancer service named lb and an application service named appone to provide services externally with the domain name appone.example.com.

```
lb:
image: registry.aliyuncs.com/acs/proxy:0.5
hostname: proxy # The domain name of the service is proxy, which is resolved to all containers with the image.
ports:
- '80:80'
restart: always
labels:
# With the use of Addon, the proxy image can function as a subscription registry center and dynamically load the
service route.
aliyun.custom_addon: "proxy"
# A proxy image container is deployed for each VM.
aliyun.global: "true"
# A Server Load Balancer instance is bound to the front-end.
aliyun.lb.port_80: tcp://proxy_test:80
environment:
# It indicates the range of backend containers that support route loading: "*" indicates the whole cluster. By default,
it indicates services in an application.
ADDITIONAL_SERVICES: "*"
appone:
ports:
- 80/tcp
- 443/tcp
image: 'nginx:latest'
labels:
# http/https/ws/wss are available.
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.example.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 alias 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.

# Custom routing-Support TCP Protocol

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 node (ECS), the client cannot access the local server through Server Load Balancer due to limitation. In this document, common TCP-based Redis is used as an example to describe how to solve the problem through custom routing acs/proxy.

## Solution 1: Deploy client and server containers on different nodes through scheduling containers.

The following is a sample application template (the lb label and swarm filter are applied):

NOTE:

- If the scheduling operation fails, go to the Container Service Management Console, and click **Services** in the left navigation pane to go to the service list page. Select the service you want to schedule, and click **Reschedule**. Click **Force Reschedule** in the pop-up dialog box, and then click **OK**.
- Clicking **Force Reschedule** will discard the volume of the existing container. Back up and migrate data in the container before doing so.

## Solution 2: Clients inside the container cluster access the server through links, while clients outside access the server through 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 through Custom routing (which is based on HAProxy and serves as a proxy server), while clients outside access the server through

## Server Load Balancer

The following is a sample application template (the lb label and Custom routing image are used):

```
lb:
image: registry.aliyuncs.com/acs/proxy:0.5
ports:
- '6379:6379/tcp'
restart: always
labels:
# With the use of Addon, the proxy image can function as a subscription registry center and dynamically load the
service route.
aliyun.custom_addon: "proxy"
# A proxy image container is deployed for each VM.
aliyun.global: "true"
# The front-end is bound to Server Load Balancer, and an lb label is used.
aliyun.lb.port_6379: tcp://proxy_test:6379
# Indicate that the custom route must be started after the master Redis and slave Redis, and the custom route
depends on the master Redis and slave Redis.
aliyun.depends: redis-master,redis-slave
environment:
# It indicates the range of backend containers that support route loading: "*" indicates the whole cluster. By default,
it indicates services in an application.
ADDITIONAL_SERVICES: "*"
EXTRA_DEFAULT_SETTINGS: "log rsyslog local0,log global,option httplog"
# Configure 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:
# Indicate that Port 6379 should be exposed for the custom route.
aliyun.proxy.TCP_PORTS: "6379"
# Indicate that the system route should be added to the custom route.
aliyun.proxy.required: "true"
redis-slave:
ports:
- 6379/tcp
image: 'redis:alpine'
links:
- redis-master
labels:
# Indicate that Port 6379 should be exposed for the custom route.
aliyun.proxy.TCP_PORTS: "6379"
# Indicate that the system route should be added to the custom route.
aliyun.proxy.required: "true"
# Indicate that the slave Redis depends on the master Redis and should 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 using a Custom routing image, so the Container Server becomes highly available.

## Custom routing-Support multiple HTTPS certificates

In this example, an acs/proxy image is used.

```
lb:
image: registry.aliyuncs.com/acs/proxy:0.5
ports:
- '80:80'
- '443:443' # The port for HTTPS must be exposed.
restart: always
labels:
# With the use of Addon, the proxy image can function as a subscription registry center and dynamically load the service route.
aliyun.custom_addon: "proxy"
# A proxy image container is deployed for each VM.
aliyun.global: "true"
# A Server Load Balancer instance is bound to the front-end.
aliyun.lb.port_80: tcp://proxy_test:80
aliyun.lb.port_443:tcp://proxy_test:443
environment:
# It indicates the range of backend containers that support route loading: "*" indicates the whole cluster. By default, it indicates services in an application.
ADDITIONAL_SERVICES: "*"

appone:
expose: # For proxied services, use expose or ports to tell proxy containers which port should be exposed.
- 80/tcp
image: 'nginx:latest'
labels:
# You can specify paths when configuring URLs. In this example, http/https/ws/wss are available.
aliyun.proxy.VIRTUAL_HOST: "https://appone.example.com"

# Configure the appone certificate
restart: always
apptwo:
expose: # For proxied services, use expose or ports to tell proxy containers which port should 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 are available.
aliyun.proxy.VIRTUAL_HOST: "https://apptwo.example.com"

# Configure the apptwo certificate
restart: always
```

As shown above, the domain names of the appone and apptwo services are specified through aliyun.proxy.VIRTUAL_HOST. If you need to configure the certificate, set the protocol to https. Then specify the certificate content through aliyun.proxy.SSL_CERT. The method of configuring the certificate content is described below.

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 and private key files).

```
$ 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, refer to the templates mentioned above and corresponding Documentation sample.

# Release policy

## 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 the route weights, you can switch traffic between different service versions. After verifying that there are no errors, you can use the release confirmation method to delete the old service version. If the new version does not pass verification, the release is rolled back and the new version is deleted.

## Prerequisite

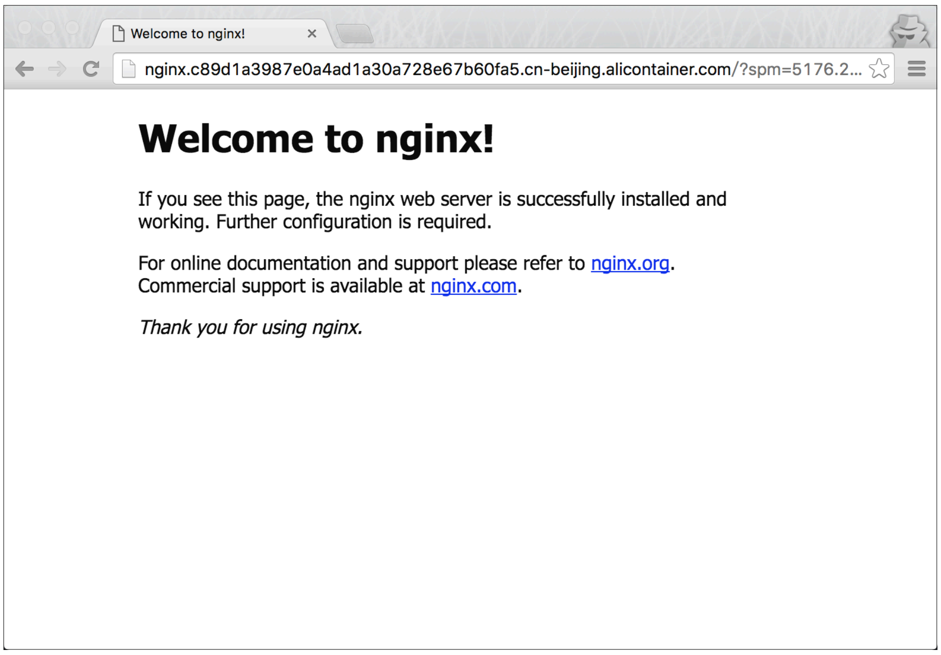
The routing service must be upgraded to the latest version. For details, refer to Upgrade system services.

## Scenario

In the following example, assume that you want to perform a blue-green release for a 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 deployment, the page is as follows.


Welcome to nginx!

nginx.c89d1a3987e0a4ad1a30a728e67b60fa5.cn-beijing.alicontainer.com/?spm=5176.2...

**Welcome to nginx!**

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

*Thank you for using nginx.*

## Operating procedure

Log on to the Container Service console.

Click **Applications** in the left navigation pane.

Select the cluster of the desired application.

Locate the application and click **Update**.

Set the release mode and the configuration of the new service version.

**Note:**

- The new and old versions cannot share the same name.
- To ensure that the application does not experience downtime when switching versions, the weight of the new service version is set to 0 by default. In 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
```

Click **OK** to release the changed version.

The release process goes through two statuses:

- Blue-green release in progress: Indicates that the new service version has not been launched.

  Blue-green release awaiting confirmation: Indicates that the new service version has been launched. At this point, you need to confirm the release or roll it back, in order to perform another release.

  In the application details page (click the application name), you can see that the new and old application versions coexist. Here, blue designates the old service version and green designates the new service version. If a service does not change from one version to the other, a yellow label is displayed. This indicates that this application will not change during the blue-green release.

In the application details page, click **Routes** and click **Set Service Weight**.

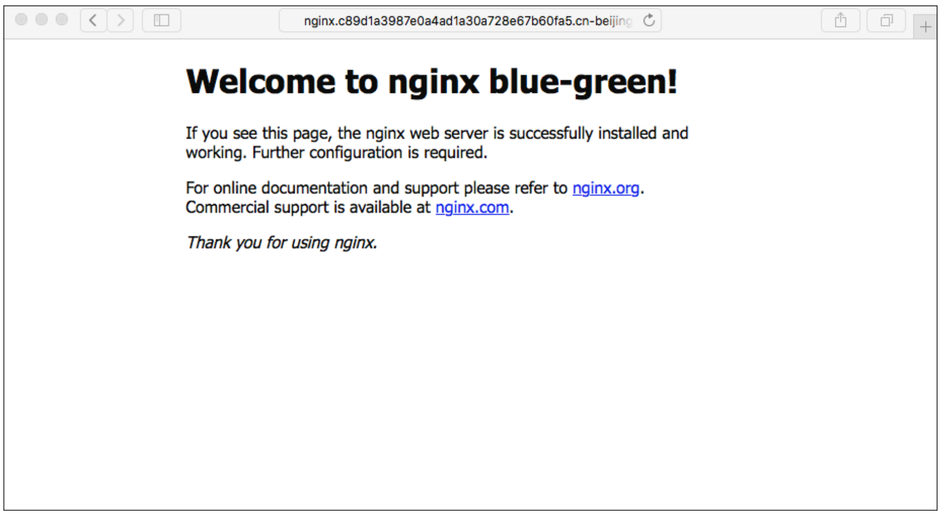
The old version has a weight of 100 and the new version has a weight of 0.

To realize zero downtime update, you need to first set the weight of the new version to 100; at this point, the new version and old version accounts for 50% of the weight respectively and they both 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, you must adjust the weights in two steps and only adjust the weight of one version in each step. For example, first, adjust the weight of the new version from 0 to 100; and then, adjust the weight of the old version from 100 to 0.

Then, adjust the old service version weight to 0 and the new version weight to 100.

Because the route service will hold the session by default, you can open a new browser window to access the new version. The results are shown below.


nginx.c89d1a3987e0a4ad1a30a728e67b60fa5.cn-beijing

**Welcome to nginx blue-green!**

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

*Thank you for using nginx.*

After the entire release process has been verified, click **Confirm Release Completion** in the **Application List** page and click **Confirm** in the pop-up dialog box 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.

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 verifying that there are no errors, you can use the release confirmation method to delete the old service version. If the new version does not pass verification, the release is rolled back and the new version is deleted.

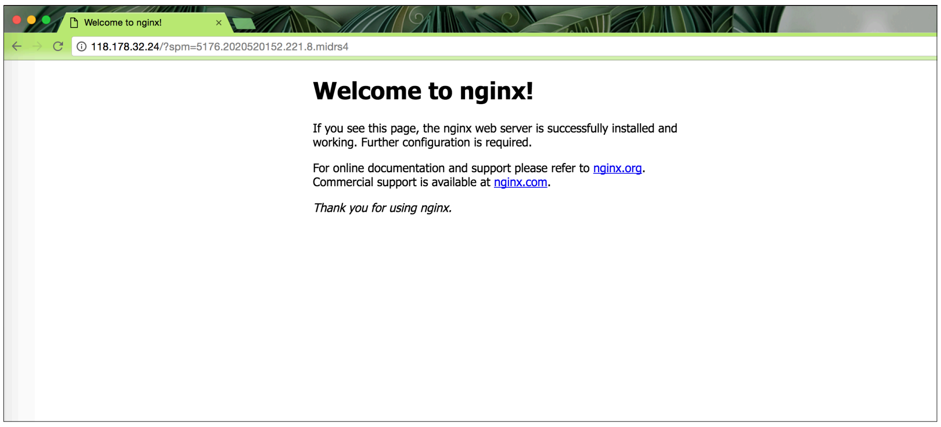
## Scenario

Assume that you want to perform a blue-green release for a 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 deployment, the page is as follows.



Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org. Commercial support is available at nginx.com.

*Thank you for using nginx.*

## Use instructions

As all containers need to publish host ports, when you perform Server Load Balancer route blue-green release, make sure that the number of containers of a service is equal to or less than half of the number of hosts in the cluster; otherwise, port conflict might occur.

It is recommended that you perform scaling in to reduce the number of containers of a service to half of the number of hosts in the cluster before performing blue-green release; and perform scaling out to increase the number of containers to the original amount after the blue-green release is completed.

## Operating procedure

Log on to the Container Service console.

Click **Applications** in the left navigation pane.

Select the cluster of the desired application.

Locate the application and click **Update**.

Set the release mode and the configuration of the new service version.

**Note:**

- The new and old versions cannot share the same name.
- To ensure that the application does not experience downtime when switching versions, the Server Load Balancer weight of the new service version is set to 0 by default. In 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
```

Click **OK** to release the changed version.

The release process goes through two statuses:

- Blue-green release in progress: Indicates that the new service version has not been launched.

  Blue-green release awaiting confirmation: Indicates that the new service version has been launched. At this time, you need to confirm the release or roll it back, to perform another release.

  In the application details page, you can see that the new and old application versions coexist. Here, blue designates the old service version and green designates the new service version. If a service does not change from one version to the other, a yellow label is displayed. This indicates that this application will not change during the blue-green release.

Click the application name, click **Routes** and click **Set Service Weight**.

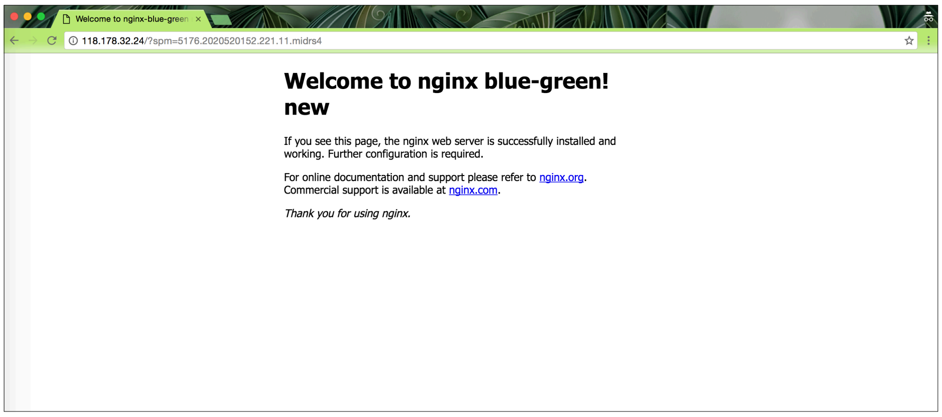
The Server Load Balancer weight of the old service is 100 and the Server Load Balancer weight of the new version is 0.

To realize zero downtime update, you need to first set the weight of the new version to 100; at this point, the new version and old version accounts for 50% of the weight respectively and they both 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, you must adjust the weights in two steps and only adjust the weight of one version in each step. For example, first, adjust the weight of the new version from 0 to 100; and then, adjust the weight of the old version from 100 to 0.

Then, adjust the old service version weight to 0 and the new version weight to 100.

Open a new browser window to access the new version. The results are shown below.


Welcome to nginx-blue-green
118.178.32.24/?spm=5176.2020520152.221.11.midrs4

**Welcome to nginx blue-green!**
**new**

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

*Thank you for using nginx.*

After the entire release process has been verified, click **Confirm Release Completion** in the **Application List** page and click **Confirm** in the pop-up dialog box 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.