

Alibaba Cloud Container Service for Kubernetes

User Guide

Issue: 20181008

Legal disclaimer

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

1. You shall download and obtain this document from the Alibaba Cloud website or other Alibaba Cloud-authorized channels, and use this document for your own legal business activities only. The content of this document is considered confidential information of Alibaba Cloud. You shall strictly abide by the confidentiality obligations. No part of this document shall be disclosed or provided to any third party for use without the prior written consent of Alibaba Cloud.
2. No part of this document shall be excerpted, translated, reproduced, transmitted, or disseminated by any organization, company, or individual in any form or by any means without the prior written consent of Alibaba Cloud.
3. The content of this document may be changed due to product version upgrades, adjustments, or other reasons. Alibaba Cloud reserves the right to modify the content of this document without notice and the updated versions of this document will be occasionally released through Alibaba Cloud-authorized channels. You shall pay attention to the version changes of this document as they occur and download and obtain the most up-to-date version of this document from Alibaba Cloud-authorized channels.
4. This document serves only as a reference guide for your use of Alibaba Cloud products and services. Alibaba Cloud provides the document in the context that Alibaba Cloud products and services are provided on an "as is", "with all faults" and "as available" basis. Alibaba Cloud makes every effort to provide relevant operational guidance based on existing technologies. However, Alibaba Cloud hereby makes a clear statement that it in no way guarantees the accuracy, integrity, applicability, and reliability of the content of this document, either explicitly or implicitly. Alibaba Cloud shall not bear any liability for any errors or financial losses incurred by any organizations, companies, or individuals arising from their download, use, or trust in this document. Alibaba Cloud shall not, under any circumstances, bear responsibility for any indirect, consequential, exemplary, incidental, special, or punitive damages, including lost profits arising from the use or trust in this document, even if Alibaba Cloud has been notified of the possibility of such a loss.
5. By law, all the content of the Alibaba Cloud website, including but not limited to works, products, images, archives, information, materials, website architecture, website graphic layout, and webpage design, are intellectual property of Alibaba Cloud and/or its affiliates. This intellectual property includes, but is not limited to, trademark rights, patent rights, copyrights, and trade

secrets. No part of the Alibaba Cloud website, product programs, or content shall be used, modified, reproduced, publicly transmitted, changed, disseminated, distributed, or published without the prior written consent of Alibaba Cloud and/or its affiliates. The names owned by Alibaba Cloud shall not be used, published, or reproduced for marketing, advertising, promotion, or other purposes without the prior written consent of Alibaba Cloud. The names owned by Alibaba Cloud include, but are not limited to, "Alibaba Cloud", "Aliyun", "HiChina", and other brands of Alibaba Cloud and/or its affiliates, which appear separately or in combination, as well as the auxiliary signs and patterns of the preceding brands, or anything similar to the company names, trade names, trademarks, product or service names, domain names, patterns, logos, marks, signs, or special descriptions that third parties identify as Alibaba Cloud and/or its affiliates).

6. Please contact Alibaba Cloud directly if you discover any errors in this document.

Generic conventions

Table -1: Style conventions

Style	Description	Example
	This warning information indicates a situation that will cause major system changes, faults, physical injuries, and other adverse results.	 Danger: Resetting will result in the loss of user configuration data.
	This warning information indicates a situation that may cause major system changes, faults, physical injuries, and other adverse results.	 Warning: Restarting will cause business interruption. About 10 minutes are required to restore business.
	This indicates warning information, supplementary instructions, and other content that the user must understand.	 Note: Take the necessary precautions to save exported data containing sensitive information.
	This indicates supplemental instructions, best practices, tips, and other content that is good to know for the user.	 Note: You can use Ctrl + A to select all files.
>	Multi-level menu cascade.	Settings > Network > Set network type
Bold	It is used for buttons, menus, page names, and other UI elements.	Click OK .
Courier font	It is used for commands.	Run the <code>cd /d C:/windows</code> command to enter the Windows system folder.
<i>Italics</i>	It is used for parameters and variables.	<code>bae log list --instanceid Instance_ID</code>
[] or [a b]	It indicates that it is a optional value, and only one item can be selected.	<code>ipconfig [-all -t]</code>
{ } or {a b}	It indicates that it is a required value, and only one item can be selected.	<code>swich {stand / slave}</code>

Contents

Legal disclaimer.....	I
Generic conventions.....	I
1 Kubernetes cluster.....	1
1.1 Introduction.....	1
1.1.1 Overview.....	1
1.1.2 Alibaba Cloud Kubernetes vs. self-built Kubernetes.....	2
1.2 Authorizations.....	4
1.2.1 Use sub-accounts.....	4
1.2.2 Sub-account Kubernetes permission configuration guide.....	6
1.3 Clusters.....	9
1.3.1 View cluster overview.....	9
1.3.2 Create a Kubernetes cluster.....	10
1.3.3 Plan Kubernetes CIDR blocks under VPC.....	20
1.3.4 Connect to a Kubernetes cluster by using kubectl.....	24
1.3.5 Access Kubernetes clusters by using SSH.....	25
1.3.6 Access Kubernetes clusters by using SSH key pairs.....	27
1.3.7 Upgrade a cluster.....	28
1.3.8 Scale out or in a cluster.....	30
1.3.9 Cluster auto scaling.....	31
1.3.10 Delete a cluster.....	39
1.4 Nodes.....	41
1.4.1 Add an existing ECS instance.....	41
1.4.2 View node list.....	45
1.4.3 Node monitoring.....	46
1.4.4 Manage node labels.....	48
1.4.5 Set node scheduling.....	51
1.5 Namespaces.....	52
1.5.1 Create a namespace.....	52
1.5.2 Configure resource quotas for namespaces.....	53
1.5.3 Update a namespace.....	57
1.5.4 Delete a namespace.....	59
1.6 Applications.....	60
1.6.1 Create an application by using an image.....	60
1.6.2 Create an application in Kubernetes dashboard.....	68
1.6.3 Create an application by using an orchestration template.....	71
1.6.4 Manage applications by using commands.....	76
1.6.5 Simplify Kubernetes application deployment by using Helm.....	76
1.6.6 Create a service.....	84
1.6.7 Service scaling.....	86
1.6.8 View services.....	88
1.6.9 Update a service.....	89

1.6.10 Delete a service.....	90
1.6.11 Use an application trigger.....	91
1.6.12 View pods.....	93
1.6.13 Change container configurations.....	94
1.6.14 Schedule a pod to a specified node.....	95
1.6.15 View image list.....	99
1.7 Server Load Balancer and Ingress.....	100
1.7.1 Overview.....	100
1.7.2 Access services by using Server Load Balancer.....	100
1.7.3 Support for Ingress.....	107
1.7.4 Configure Ingress monitoring.....	112
1.7.5 Ingress configurations.....	115
1.7.6 Create an Ingress in Container Service console.....	118
1.7.7 Update an Ingress.....	128
1.7.8 View Ingress details.....	130
1.7.9 Delete an Ingress.....	131
1.8 Config maps and Secrets.....	132
1.8.1 Create a config map.....	132
1.8.2 Use a config map in a pod.....	136
1.8.3 Update a config map.....	140
1.8.4 Delete a config map.....	144
1.8.5 Create a secret.....	145
1.8.6 View secret details.....	147
1.8.7 Update a secret.....	148
1.8.8 Delete a secret.....	149
1.9 Storage.....	150
1.9.1 Overview.....	151
1.9.2 Install the plug-in.....	151
1.9.3 Use Alibaba Cloud cloud disks.....	155
1.9.4 Use Alibaba Cloud NAS.....	161
1.9.5 Use Alibaba Cloud OSS.....	168
1.9.6 Create a persistent storage volume claim.....	172
1.9.7 Using persistent storage volume claim.....	175
1.10 Logs.....	178
1.10.1 Overview.....	178
1.10.2 View cluster logs.....	178
1.10.3 Configure Log4jAppender for Kubernetes and Log Service.....	179
1.10.4 A solution to log collection problems of Kubernetes clusters by using log-pilot, Elasticsearch, and Kibana.....	184
1.11 Monitoring.....	190
1.11.1 Integration and usage with CloudMonitor.....	190
1.11.2 Use Grafana to display monitoring data.....	195
1.11.3 Use an HPA auto scaling container.....	202
1.11.4 Monitor resources and send alarm notifications by using resource groups...	206

1.12 Manage a release.....	212
1.12.1 Manage a Helm-based release.....	212
1.12.2 Use batch release on Alibaba Cloud Container Service for Kubernetes.....	216
1.13 Template.....	220
1.13.1 Create an orchestration template.....	220
1.13.2 Edit an orchestration template.....	222
1.13.3 Save an existing orchestration template as a new one.....	225
1.13.4 Download an orchestration template.....	226
1.13.5 Delete an orchestration template.....	227
1.14 App catalog.....	228
1.14.1 App catalog overview.....	228
1.14.2 View app catalog list.....	229
1.15 Service catalog.....	230
1.15.1 Overview.....	230
1.15.2 Enable service catalog function.....	231

1 Kubernetes cluster

1.1 Introduction

1.1.1 Overview

Kubernetes is a popular open-source container orchestration technology. To allow you to use Kubernetes to manage container applications in Alibaba Cloud, Alibaba Cloud Container Service provides support for Kubernetes clusters.

You can create a safe and high-availability Kubernetes cluster in the Container Service console. The Kubernetes cluster integrates with the virtualization, storage, network, and security capabilities of Alibaba Cloud to provide scalable, high-performance container application management, simplify cluster creation and expansion, and focus on the development and management of containerized applications.

Kubernetes supports the deployment, expansion, and management of containerized applications, and provides the following features:

- Elastic expansion and self-reparation.
- Service discovery and server load balancing.
- Service release and rollback.
- Secret and configuration management.

Limits

- Currently, Kubernetes clusters only support Linux containers. The support for Kubernetes Windows containers is in the works.
- Currently, Kubernetes clusters only support the network type Virtual Private Cloud (VPC). You can select to create a VPC or use an existing VPC when creating the Kubernetes cluster.
- Currently, all resources (for example, the Elastic Compute Service (ECS) instances and Server Load Balancer instances) created in the Kubernetes cluster are Pay-As-You-Go resources. You can change the Pay-As-You-Go instances to monthly or yearly subscription instances in the console. Monthly or yearly subscription resources will be supported in the future.

Related open-source projects

- Alibaba Cloud Kubernetes Cloud Provider: <https://github.com/AliyunContainerService/kubernetes>.

- Alibaba Cloud VPC network drive for Flannel: <https://github.com/coreos/flannel/blob/master/Documentation/alibabacloud-vpc-backend.md>.

If you have any questions or suggestions regarding a specific project, you are welcome to raise an issue or pull a request in the community.

1.1.2 Alibaba Cloud Kubernetes vs. self-built Kubernetes

Advantages of Alibaba Cloud Kubernetes

Convenient

- Supports creating Kubernetes clusters with one click in the Container Service console.
- Supports upgrading Kubernetes clusters with one click in the Container Service console.

You may have to deal with self-built Kubernetes clusters of different versions at the same time, including version 1.8.6, 1.9.4, and 1.10 in the future. Upgrading clusters each time brings you great adjustments and Operation & Maintenance (O&M) costs. Container Service upgrade solution performs rolling update by using images and uses the backup policy of complete metadata, which allows you to conveniently roll back to the previous version.

- Supports expanding or contracting Kubernetes clusters conveniently in the Container Service console.

Container Service Kubernetes clusters allow you to expand or contract the capacity vertically with one click to respond to the peak of the data analysis business quickly.

Strong

Function	Description
Network	<ul style="list-style-type: none">• High-performance Virtual Private Cloud (VPC) network plug-in.• Supports network policy and flow control. <p>Container Service can provide you with continuous network integration and the best network optimization.</p>
Server Load Balancer	<p>Supports creating Internet or intranet Server Load Balancer instances.</p> <p>If your self-built Kubernetes clusters are implemented by using the self-built Ingress, publishing the business frequently may cause the Ingress to have pressure about configuration and higher error probabilities. The Server</p>

Function	Description
	Load Balancer solution of Container Service supports Alibaba Cloud native high-availability Server Load Balancer, and can automatically modify and update the network configurations. This solution has been used by a large number of users for a long time, which is more stable and reliable than self-built Kubernetes.
Storage	Container Service integrates with Alibaba Cloud cloud disk, NAS, and EBS, and provides the standard FlexVolume drive. Self-built Kubernetes clusters cannot use the storage resources on the cloud. Alibaba Cloud Container Service provides the best seamless integration.
O&M	<ul style="list-style-type: none"> Integrates with Alibaba Cloud Log Service and CloudMonitor. Supports auto scaling.
Image repository	<ul style="list-style-type: none"> High availability. Supports high concurrency. Supports speeding up the pull of images. Supports P2P distribution. <p>The self-built image repository may crash if you pull images from millions of clients at the same time. Enhance the reliability of the image repository by using the Apsara Stack version of Container Service image repository , which reduces the O&M burden and upgrade pressure.</p>
Stable	<ul style="list-style-type: none"> Special teams to guarantee the stability of containers. Each Linux version and Kubernetes version are provided to you after the strict test. <p>Container Service provides the Docker CE to reveal all the details and promotes the repair capabilities of Docker. If you have issues such as Docker Engine hang, network problems, and kernel compatibility, Container Service provides you with the best practices.</p>
High availability	<ul style="list-style-type: none"> Supports multiple zones.

Function	Description
	<ul style="list-style-type: none">• Supports backup and disaster recovery.
Technical support	<ul style="list-style-type: none">• Provides the Kubernetes upgrade capabilities. Supports upgrading a Kubernetes cluster to the latest version with one click.• Alibaba Cloud container team is responsible for solving problems about containers in your environment.

Costs and risks of self-built Kubernetes

- Building clusters is complicated

You must manually configure the components, configuration files, certificates, keys, plug-ins, and tools related to Kubernetes. It takes several days or weeks for professional personnel to build the cluster.

- For public cloud, it takes you significant costs to integrate with cloud products.

You must devote your own money to integrate with other products of Alibaba Cloud, such as Log Service, monitoring service, and storage management.

- The container is a systematic project, involving network, storage, operating system, orchestration, and other technologies, which requires the devotion of professional personnel.
- The container technology is continuously developing and the version iteration is fast, which requires continuous upgrade and test.

1.2 Authorizations

1.2.1 Use sub-accounts

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

Step 1 Create sub-accounts and enable console logon

1. Log on to the [RAM console](#).
2. Click **Users** in the left-side navigation pane. Click **Create User** in the upper-right corner.
3. Enter the username of the sub-account and then click **OK**.
4. On the User Management page, click **Manage** at the right of the created sub-account.
5. Click **Enable Console Logon** in the **Web Console Logon Management** section.

6. Enter the logon password in the appeared dialog box and click **OK**.

Step 2 Grant sub-accounts permissions to access Container Service

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

RAM	User Management Create User Refresh				
Dashboard	<div> <div>User Name</div> <div>Search by User Name</div> <div>Search</div> </div>				
Users					
Groups					
Policies					
Roles					
Settings					
	User Name/Display Name	Description	Created At	Actions	
	zhongqinwei	zhongqinwei	2017-11-01 11:26:17	Manage	Authorize
	AliyunOSSTokenGeneratorUser		2017-11-27 11:55:21	Manage	Authorize
	yangjiale	yangjiale	2017-11-01 11:24:54	Manage	Authorize

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

Edit User-Level Authorization

Members added to this group have all the permissions of this group. A member cannot be added to the same group more than once.

Available Authorization Policy Names	Type
CS	
EcsRamRoleDocument...	
AliyunACSResourcesAccess_yangx...	Custom
aliyun container s...	
AliyunCSReadOnlyAccess	System
Provides read-only...	
AliyunCSFullAccess	System
Provides full acce...	

Selected Authorization Policy Name	Type
AdministratorAccess	System
Provides full acce...	
AliyunACSResourcesAccess_xingy...	Custom
aliyun container s...	

OK

Close

You can use the following system default authorization policies:

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

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

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

Log on to the [Container Service console](#) with a sub-account.

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

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

Cloud Resource Access Authorization

Note: If you need to modify role permissions, please go to the RAM Console. [Role Management](#). If you do not configure it correctly, the following role: CS will not be able to obtain the required permissions. X

CS needs your permission to access your cloud resources.
The system has created roles for the following user. These roles can be found below. CS. After authorization, CS will have access to your cloud resources.

AliyunCSDefaultRole Description: The Container Service will use this role to access your resources in other services. Permission Description: The policy for AliyunCSDefaultRole.	✓
AliyunCSClusterRole Description: The clusters of Container Service will use this role to access your resources in other services. Permission Description: The policy for AliyunCSClusterRole.	✓

[Confirm Authorization Policy](#) [Cancel](#)

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

1.2.2 Sub-account Kubernetes permission configuration guide

This document helps you understand how to configure the Kubernetes Resource Access Management (RAM) cluster permissions for sub-accounts and the corresponding Kubernetes RBAC application permissions within the cluster through the Container Service console .

Prerequisite

- The sub-account authorization page is visible only to the main account. You have an Alibaba Cloud main account and one or several sub-accounts.
- Due to the security restrictions of Alibaba Cloud RAM, when you modify the sub-account RAM authorization after you pass the authorization of the Container Service console, manually perform authorization on the RAM console for the target sub-account according to the reference policy content and operation instructions on the page.

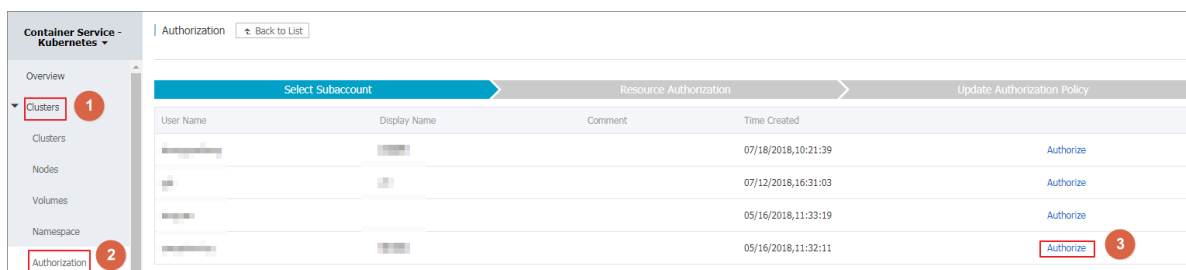
Procedure



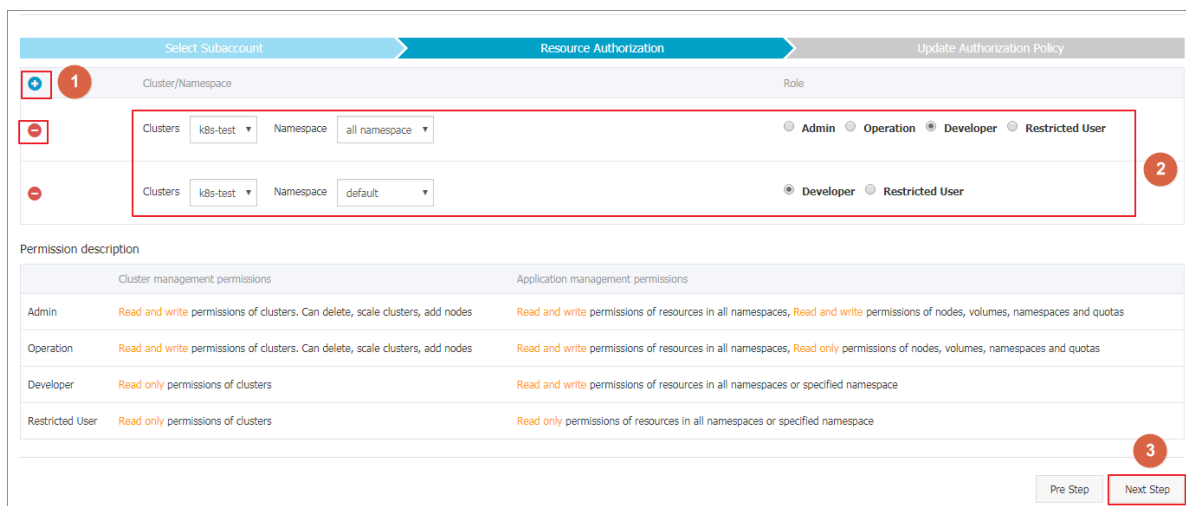
Note:

Use your main account to log on to the Container Service console because the sub-account authorization page is visible only to the main account.

1. Log on to the [Container Service console](#).
2. Click Kubernetes **Clusters** > **Authorization** in the left-side navigation pane to go to the Authorization page.
3. In the sub-account list, select the sub-account that requires authorization, and click **Authorize**.



4. On the Resource Authorization page, you can add a cluster or namespace level permission configuration by clicking the plus sign in the upper left corner of the table, and select an role. You can also click the minus sign at the beginning of the configuration line to remove the target role.



For information about definitions of roles in clusters and namespaces, see the permission description below:

Table 1-1: Role permission description

	Cluster management permissions	Application management permissions
Admin	Read and write permissions of clusters. Delete clusters, scale clusters, and add nodes .	Read and write permissions of resources in all namespaces. Read and write permissions of nodes, volumes, namespaces, and quotas.
Operation	Read and write permissions of clusters. delete clusters, scale clusters, and add nodes .	Read and write permissions of resources in all namespaces, Read only permissions of nodes, volumes, namespaces, and quotas.
Developer	Read only permissions of clusters.	Read and write permissions of resources in all namespaces or specified namespace.
Restricted user	Read only permissions of clusters.	Read only permissions of resources in all namespaces or specified namespace.

5. After the configuration is completed, if you change the target sub-account RAM cluster permission, the Kubernetes cluster RAM permission reference configuration corresponding to the configuration item is displayed on the Update Authorization Policy page. You can complete the sub-account authorization update in the RAM console according to the instructions on the page.

Additional instructions

To avoid affecting the normal use of currently accessible Kubernetes clusters by the sub-accounts , the Container Service console is temporarily compatible with the old cluster access permission control. In a period of time, the original sub-account can access the Kubernetes cluster without RBAC application permission check. If you are a sub-account user, please contact the main account in time for authorization according to the cluster type and compatibility method below.

For the existing cluster created by the current sub-account, you can complete the automatic upgrade of the cluster application permissions by clicking **Upgrade current cluster authorization information** on the cluster details page.

After the end of the notice period, if a sub-account obtains no authentication from the main account or gets no permission management update, the sub-account is banned from accessing the application console corresponding to the cluster.

Table 1-2: Compatible cluster description

Compatible cluster type	Compatibility method
Clusters created by an existing sub-account	Prompt permission management upgrade notice and provide one-click upgrade link. The sub-account can complete application authorization by clicking the upgrade link.
Clusters with access authorized by an existing RAM	Prompt the permission management upgrade notice: Please contact the main account to complete the application authorization.
Clusters with access authorized by a newly created RAM	Prompt the permission management upgrade notice: Please contact the main account to complete the application authorization.

1.3 Clusters

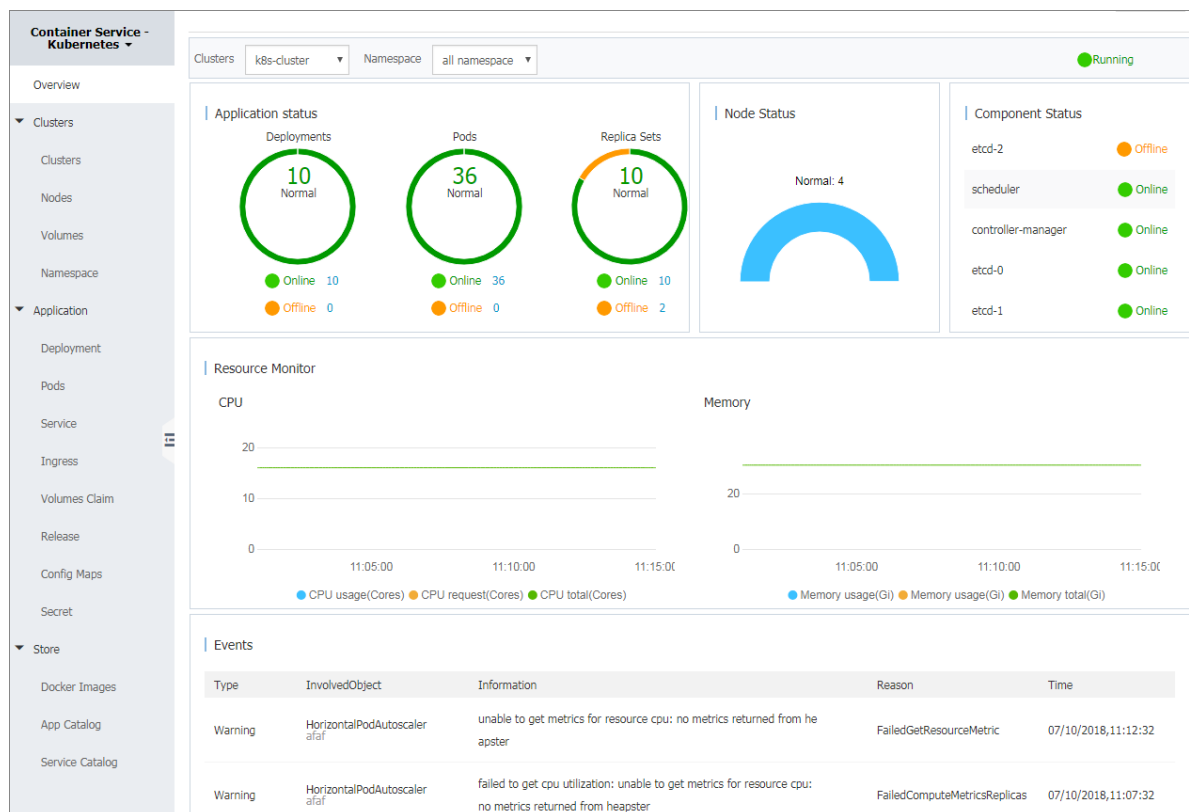
1.3.1 View cluster overview

You can view the application status, component status, and resource monitoring charts on the Overview page of Alibaba Cloud Container Service Kubernetes clusters, which allows you to quickly understand the health status of clusters.

Procedure

1. Log on to the [Container Service console](#).
2. Under Kubernetes, click **Overview** in the left navigation bar to enter the Kubernetes cluster overview page.
3. Select the cluster and namespace from the Clusters and Namespace drop-down lists. You can view the application status, component status, and resource monitoring charts.

- **Application status:** The status of deployments, pods, and replica sets that are currently running. Green indicates the normal status and orange indicates an exception.
- **Node status:** Displays the node status of the current cluster.
- **Component status:** The components of Kubernetes clusters are generally deployed under the kube-system namespace, including the core components such as scheduler, controller-manager, and etcd.
- **Resource monitor:** Provides the monitoring charts of CPU and memory. CPU is measured in cores and is accurate to three decimal places. The minimum unit is millicores, that is, one thousandth of one core. Memory is measured in G and is accurate to three decimal places. For more information, see [Meaning of CPU](#) and [Meaning of memory](#).
- **Event:** Displays event information of the cluster, such as warnings and error events.



1.3.2 Create a Kubernetes cluster

You can create a Kubernetes cluster quickly and easily in the Container Service console.

Instructions

During cluster creation, Container Service performs the following operations:

- Create Elastic Compute Service (ECS) instances and configure to log on to other nodes from management nodes with the SSH public key. Install and configure the Kubernetes cluster by using CloudInit.
- Create a security group. This security group allows the Virtual Private Cloud (VPC) inbound access of all the ICMP ports.
- Create a new VPC and VSwitch if you do not use the existing VPC, and then create SNAT for the VSwitch.
- Create VPC routing rules.
- Create a NAT Gateway and a shared bandwidth package or Elastic IP (EIP).
- Create a Resource Access Management (RAM) user and the AccessKey. This RAM user has the permissions of querying, creating, and deleting ECS instances, the permissions of adding and deleting cloud disks, and all permissions for Server Load Balancer (SLB), CloudMonitor, VPC, Log Service, and Network Attached Storage (NAS). The Kubernetes cluster dynamically creates SLB instances, cloud disks, and VPC routing rules according to your configurations.
- Create an intranet SLB instance and expose the port 6443.
- Create an Internet SLB instance and expose the port 6443. (If you select to enable the SSH logon for Internet when creating the cluster, port 22 is exposed. Otherwise, port 22 is not exposed.)

Prerequisites

Activate the following services: Container Service, Resource Orchestration Service (ROS), and Resource Access Management (RAM).

Log on to the [Container Service console](#), [ROS console](#), and [RAM console](#) to activate the corresponding services.



Note:

The deployment of Container Service Kubernetes clusters depends on the application deployment capabilities of Alibaba Cloud ROS. Therefore, activate ROS before creating a Kubernetes cluster.

Limits

- The SLB instance created with the cluster supports only the Pay-As-You-Go billing method.
- Kubernetes clusters support only the VPC network type.

- By default, each account has a certain quota for the cloud resources it can create. The cluster fails to be created if the quota is exceeded. Make sure you have enough quota before creating the cluster. To increase your quota, open a ticket.
- By default, each account can create at most five clusters in all regions and add up to 40 worker nodes to each cluster. To create more clusters or nodes, open a ticket.

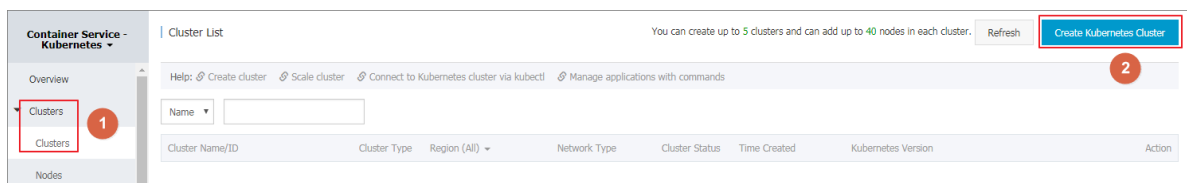
**Note:**

In a Kubernetes cluster, the maximum number of default VPC routes is 48, that is, the Kubernetes cluster has up to 48 nodes by default when using VPC. To increase the number of nodes, first open a ticket for the target VPC so as to increase the number of VPC routes, and then open a ticket for Container Service.

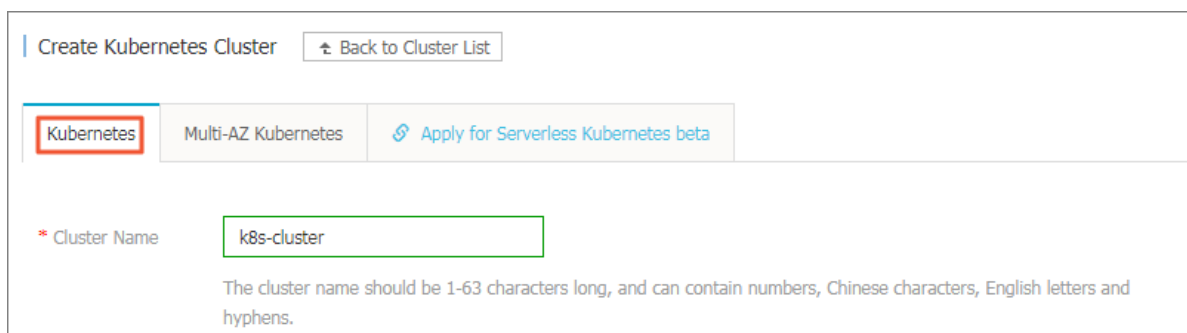
- By default, each account can create at most 100 security groups.
- By default, each account can create at most 60 Pay-As-You-Go SLB instances.
- By default, each account can create at most 20 EIPs.
- Limits for ECS instances are as follows:
 - Only the Centos operating system is supported.
 - Creating Pay-As-You-Go and Subscription ECS instances is supported.

Procedure

1. Log on to the [Container Service console](#).
2. Under Kubernetes, click **Clusters** in the left-side navigation pane.
3. Click **Create Kubernetes Cluster** in the upper-right corner.



By default, the **Create Kubernetes Cluster** page is displayed.



4. Enter the cluster name.

The cluster name can be 1–63 characters long and contain numbers, Chinese characters, English letters, and hyphens (-).

5. Select the region and zone where the cluster resides.

Region	China North 2 (Beijing)	China North 3 (Zhangjiakou)	China East 1 (Hangzhou)	China East 2 (Shanghai)	China South 1 (Shenzhen)	Hong Kong	Asia Pacific SE 1 (Singapore)	Asia Pacific SE 3 (Kuala Lumpur)	Asia Pacific SE 5 (Jakarta)	Asia Pacific SOU 1 (Mumbai)
	US East 1 (Virginia)	US West 1 (Silicon Valley)	Middle East 1 (Dubai)	EU Central 1 (Frankfurt)						
Zone	China North 2 Zone A									

6. Set the cluster network type. Kubernetes clusters support only the VPC network type.

VPC: You can select **Auto Create** to create a VPC together with the Kubernetes cluster, or select **Use Existing** to use an existing VPC. By selecting **Use Existing**, you can select a VPC and VSwitch from the two displayed drop-down lists.

- **Auto Create:** The system automatically creates a NAT gateway for your VPC when the cluster is created.
- **Use Existing:** If the selected VPC has a NAT gateway, Container Service uses the NAT gateway. Otherwise, the system automatically creates a NAT gateway by default. If you do not want the system to automatically create a NAT gateway, clear the **Configure SNAT for VPC** check box.



Note:

If you select to not automatically create a NAT gateway, configure the NAT gateway on your own to implement the VPC public network environment with secure access, or manually configure the SNAT. Otherwise, instances in the VPC cannot access public network normally, which leads to cluster creation failure.

VPC	Auto Create	Use Existing
	VPC123 (vpc-2zercq4pyanzsficlyl)	VSwitch123 (vsw-2zeydh15uwh1ej522lauo) ZoneA

7. Configure the node type, Pay-As-You-Go and Subscription types are supported.

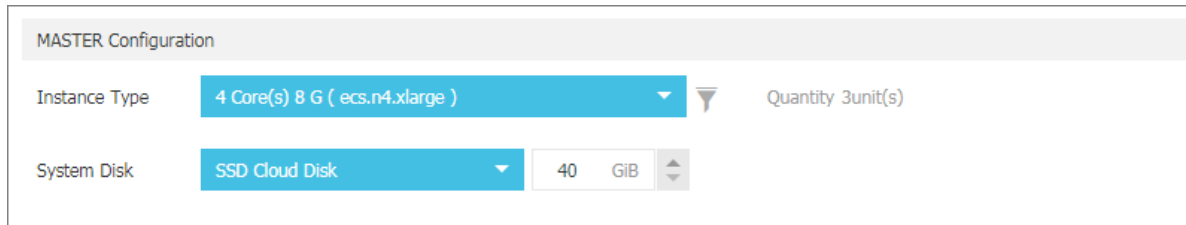
8. Configure the master nodes.

Select the instance type for the master nodes.



Note:

- Currently, master nodes support only the CentOS operating system.
- Currently, you can create only three master nodes.
- System disks are attached to the master node by default. Available system disks are SSD Cloud Disks and Ultra Cloud Disks.



MASTER Configuration

Instance Type 4 Core(s) 8 G (ecs.n4.xlarge) Quantity 3 unit(s)

System Disk SSD Cloud Disk 40 GiB

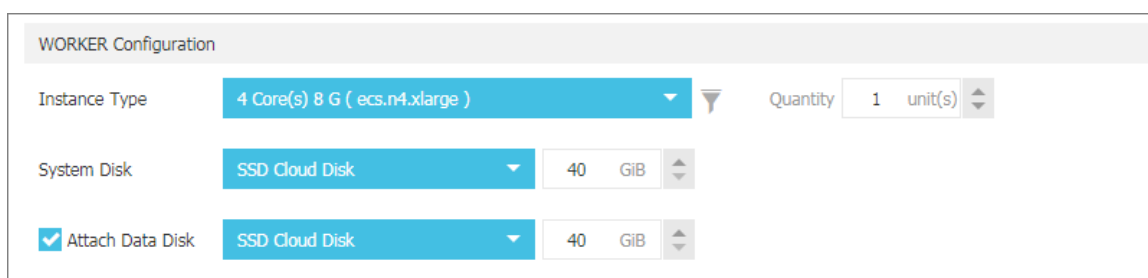
9. Configure the worker nodes. Select to create a worker node or add existing ECS instances.



Note:

- Currently, worker nodes support only the CentOS operating system.
- Each cluster can contain up to 37 worker nodes. To create more nodes, open a ticket.
- System disks are attached to the worker node by default. Available system disks are SSD Cloud Disks and Ultra Cloud Disks.
- You can also choose to manually attach a disk to the worker node, which can be an SSD Cloud Disk, Ultra Cloud Disk, or Basic Disk.

- a. To create worker nodes, select the instance type and set the number worker nodes. In this example, create one worker node.



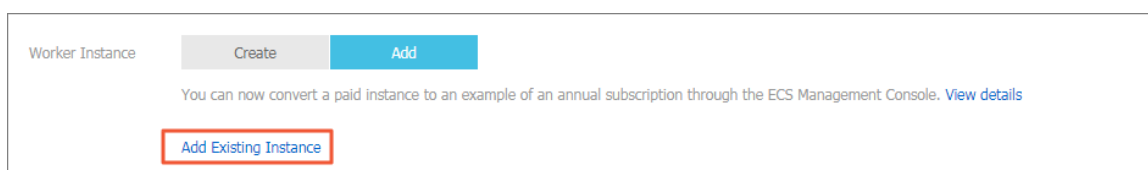
WORKER Configuration

Instance Type 4 Core(s) 8 G (ecs.n4.xlarge) Quantity 1 unit(s)

System Disk SSD Cloud Disk 40 GiB

☒ Attach Data Disk SSD Cloud Disk 40 GiB

- b. To add existing ECS instances, you must create ECS instances in the current region in advance.



Worker Instance Create Add

You can now convert a paid instance to an example of an annual subscription through the ECS Management Console. [View details](#)

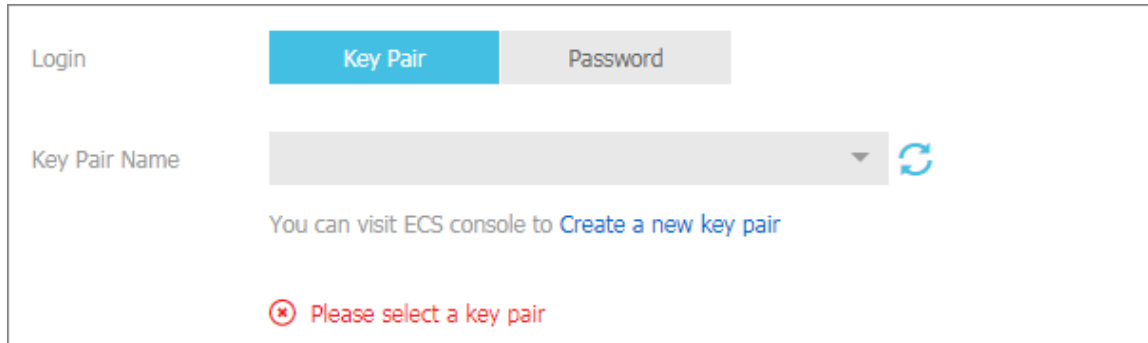
[Add Existing Instance](#)

10. Configure the logon mode.

- Configure the key pair.

Select the key pair logon mode when creating the cluster and click **Create a new key pair**.

In the ECS console, create a key pair, see [Create an SSH key pair](#). After the key pair is created, set the key pair as the credentials for logging on to the cluster.



Login

Key Pair Password

Key Pair Name

You can visit ECS console to [Create a new key pair](#)

⊗ Please select a key pair

- Configure the password.

— **Logon Password**: Configure the node logon password.

— **Confirm Password**: Confirm your node logon password.

11. Configure the **Pod Network CIDR** and **Service CIDR**.

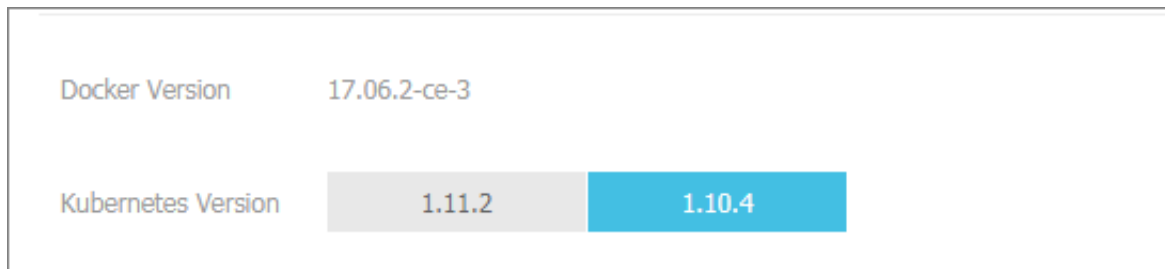


Note:

These two parameters are available only when you select to **Use Existing VPC**.

Specify **Pod Network CIDR** and **Service CIDR**. Both of them cannot overlap with the Classless Inter-Domain Routing (CIDR) block used by VPC and the existing Kubernetes clusters in VPC. The values cannot be modified after the cluster is created. Besides, Service CIDR cannot overlap with Pod Network CIDR. For more information about how to plan Kubernetes CIDR blocks, see [Plan Kubernetes CIDR blocks under VPC](#).

12. Available Docker versions and Kubernetes versions are displayed. You can view the versions and select a version according to your needs.



Docker Version 17.06.2-ce-3

Kubernetes Version 1.11.2 1.10.4

13. Set whether to configure a SNAT gateway for VPC.

**Note:**

SNAT must be configured if you select **Auto Create** VPC. If you select to **Use Existing** VPC, you can select whether to automatically configure SNAT gateway. If you choose not to automatically configure SNAT, you can configure the NAT gateway to implement VPC security access to the public network or you can configure SNAT manually. Otherwise, instances in the VPC cannot access the public network, which causes cluster creation failure.

Configure SNAT

☒ Configure SNAT for VPC

If the VPC you choose does not have access to Internet, NAT gateway and EIP will be used to configure SNAT for the VPC. During this period, NAT gateway, EIP, and other resources may be created.

14. Set whether to enable **Expose API SERVER with public SLB.**

API server provides add, delete, edit, check, watch, and other HTTP Rest interfaces for a variety of resource objects (such as pods and services).

1. If you choose to enable this option, the Internet SLB is created and the port 6443 of master nodes is exposed. The port corresponds to API Server. Then you can use kubeconfig to connect to and operate the cluster in the network outside the VPC.
2. If you choose not to enable the option, the Internet SLB is not created. You use kubeconfig to connect to and operate the cluster inside the VPC.

Public SLB

☒ Expose API SERVER with public SLB

When the selection is not open, the cluster API SERVER can not be accessed out of the VPC.

15. Select whether to enable SSH access for Internet.**Note:**

To enable SSH access for Internet, you must enable **Expose API SERVER with public SLB**.

- With this check box selected, you can access the cluster by using SSH.
- If this check box is not selected, you cannot access the cluster by using SSH or connect to the cluster by using kubectl. To access a cluster instance by using SSH, manually bind EIP to the ECS instance, configure security group rules, and open the SSH port (22). For more information, see [#unique_16](#).

Public SLB

☒ Expose API SERVER with public SLB

When the selection is not open, the cluster API SERVER can not be accessed out of the VPC.

16. Select whether to install the cloud monitoring plug-in.

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

17. Sets whether to enable Log Service. You can use an existing project or create a new project.

If you select the **Using SLS** check box, the Log Service plug-in is automatically configured in the cluster. When creating an application later, you can use Log Service quickly with simple configuration. For more information, see [Use Log Service in a Kubernetes cluster](#).

Log Service

☒ Using SLS

Select Project

Create Project

A SLS Project named k8s-log-{ClusterID} will be created automatically

18. Select whether to enable advanced configurations.

a. Select a network plug-in. Available network plug-ins are Flannel and Terway.

- Flannel: Simple and stable community Flannel cni plug-in.
- Terway: Alibaba Cloud Container Service self-developed network plug-in, which supports Alibaba Cloud flexible network card to be distributed to the container, and supports Kubernetes **NetworkPolicy** to define the inter-container access policy. Supports bandwidth limiting for the separate containers. Currently it is in the public beta.

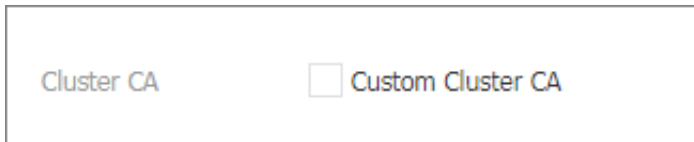
b. Set the number of pods for one node, that is, the maximum number of pods that can be run by a single node. We recommend that you maintain the default value.

Pod Number for Node

128

▼

- c. Sets whether to use **Custom Cluster CA**. With this check box selected, the CA certificate can be added to the Kubernetes cluster, which enhances the security of information exchange between the server and client.



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



Note:

Creating a Kubernetes cluster with multiple nodes lasts more than 10 minutes.

View cluster deployment results

After the cluster is successfully created, you can view the cluster in the Kubernetes cluster list of the Container Service console.

Container Service - Kubernetes

Overview

Clusters

Clusters

Nodes

Volumes

Cluster List

You can create up to 5 clusters and can add up to 40 nodes in each cluster.

Refresh

Create Kubernetes Cluster

Help: Create clusterScale clusterConnect to Kubernetes cluster via kubectlManage applications with commands

Name

Cluster Name/ID	Cluster Type	Region (All)	Network Type	Cluster Status	Time Created	Kubernetes Version	Action
k8s-cluster	Kubernetes	China East 1 (Hangzhou)	VPC vpc-bp1ik9evdj...	Running	09/26/2018,17:41:26	1.11.2	ManageView LogsDashboard Scale ClusterMore

- Click **View Logs** at the right of the cluster to view the cluster logs. To view more detailed information, click **Stack Events**.

Detailed resource deployment logs: Stack Events	
Time	Information
06/22/2018,11:13:50	c06eadb6387ec4c6080d495e243649a7b Start to DescribeK8sUserCertConfig
06/22/2018,11:03:39	c06eadb6387ec4c6080d495e243649a7b Set up k8s DNS configuration successfully
06/22/2018,11:02:30	c06eadb6387ec4c6080d495e243649a7b Stack CREATE completed successfully:o
06/22/2018,11:02:30	c06eadb6387ec4c6080d495e243649a7b Start describeStackInfo
06/22/2018,11:02:29	c06eadb6387ec4c6080d495e243649a7b Start describeStackInfo
06/22/2018,10:46:55	c06eadb6387ec4c6080d495e243649a7b Successfully to CreateStack
06/22/2018,10:46:55	c06eadb6387ec4c6080d495e243649a7b Start to wait stack ready
06/22/2018,10:46:53	c06eadb6387ec4c6080d495e243649a7b Start to create cluster task
06/22/2018,10:46:53	c06eadb6387ec4c6080d495e243649a7b Start to CreateK8sCluster
06/22/2018,10:46:53	c06eadb6387ec4c6080d495e243649a7b Start to CreateStack
06/22/2018,10:46:50	c06eadb6387ec4c6080d495e243649a7b Start to validateCIDR
06/22/2018,10:46:41	c06eadb6387ec4c6080d495e243649a7b Start create cluster certificate

- You can also click **Manage** at the right of the cluster to view the basic information and connection information of this cluster.

Basic Information	
Cluster ID: 	VPC Running Region: China East 1 (Hangzhou)
Connection Information	
API Server Internet endpoint	https:// :6443
API Server Intranet endpoint	https:// :6443
Master node SSH IP address	
Service Access Domain	 n-hangzhou.alicontainer.com
Cluster resource	
ROS	k8s-for-cs
Internet SLB	lb-
VPC	vpc-
NAT Gateway	ngw-
Connect to Kubernetes cluster via kubectl 1. Download the latest kubectl client from the Kubernetes Edition page . 2. Install and set up the kubectl client. For more information, see Installing and Setting Up kubectl 3. Configure the cluster credentials: <div> KubeConfig SSH </div>	

In the Connection Information section:

- API Server Internet endpoint:** The address and port through which the Kubernetes API server provides services for the Internet. It enables you to manage the cluster by using kubectl or other tools on your terminal.

- **API Server Intranet endpoint:** The address and port through which the Kubernetes API server provides services inside the cluster. This IP address is the address of the Server Load Balancer instance, and three master nodes in the backend provide the service.
- **Master node SSH IP address:** You can directly log on to the master nodes by using SSH to perform routine maintenance for the cluster.
- **Service Access Domain:** Provides the service in the cluster with access domain name for testing. The service access domain name suffix is `<cluster_id>.<region_id>.alicontainer.com`.

For example, you can log on to the master nodes by using SSH, and run the `kubectl get node` to view the node information of the cluster.

```
login as: root
root@1[redacted] 3's password:

Welcome to Alibaba Cloud Elastic Compute Service !

[root@iZbp1d7yvpa3j183u0url1Z ~]# kubectl get node
NAME                                STATUS    ROLES    AGE    VERSION
cn-hangzhou.i-[redacted]             Ready     <none>    17m    v1.8.4
cn-hangzhou.i-[redacted]             Ready     master    19m    v1.8.4
cn-hangzhou.i-[redacted]             Ready     master    24m    v1.8.4
cn-hangzhou.i-[redacted]             Ready     master    22m    v1.8.4
[root@iZbp1d7yvpa3j183u0url1Z ~]#
```

As shown in the preceding figure, the cluster has four nodes, including three master nodes and one worker node configured when creating the cluster.

1.3.3 Plan Kubernetes CIDR blocks under VPC

Generally, you can select to create a Virtual Private Cloud (VPC) automatically and use the default network address when creating a Kubernetes cluster in Alibaba Cloud. In some complicated scenarios, plan the Elastic Compute Service (ECS) address, Kubernetes pod address, and Kubernetes service address on your own. This document introduces what the addresses in Kubernetes under Alibaba Cloud VPC environment are used for and how to plan the CIDR blocks.

Basic concepts of Kubernetes CIDR block

The concepts related to IP address are as follows:

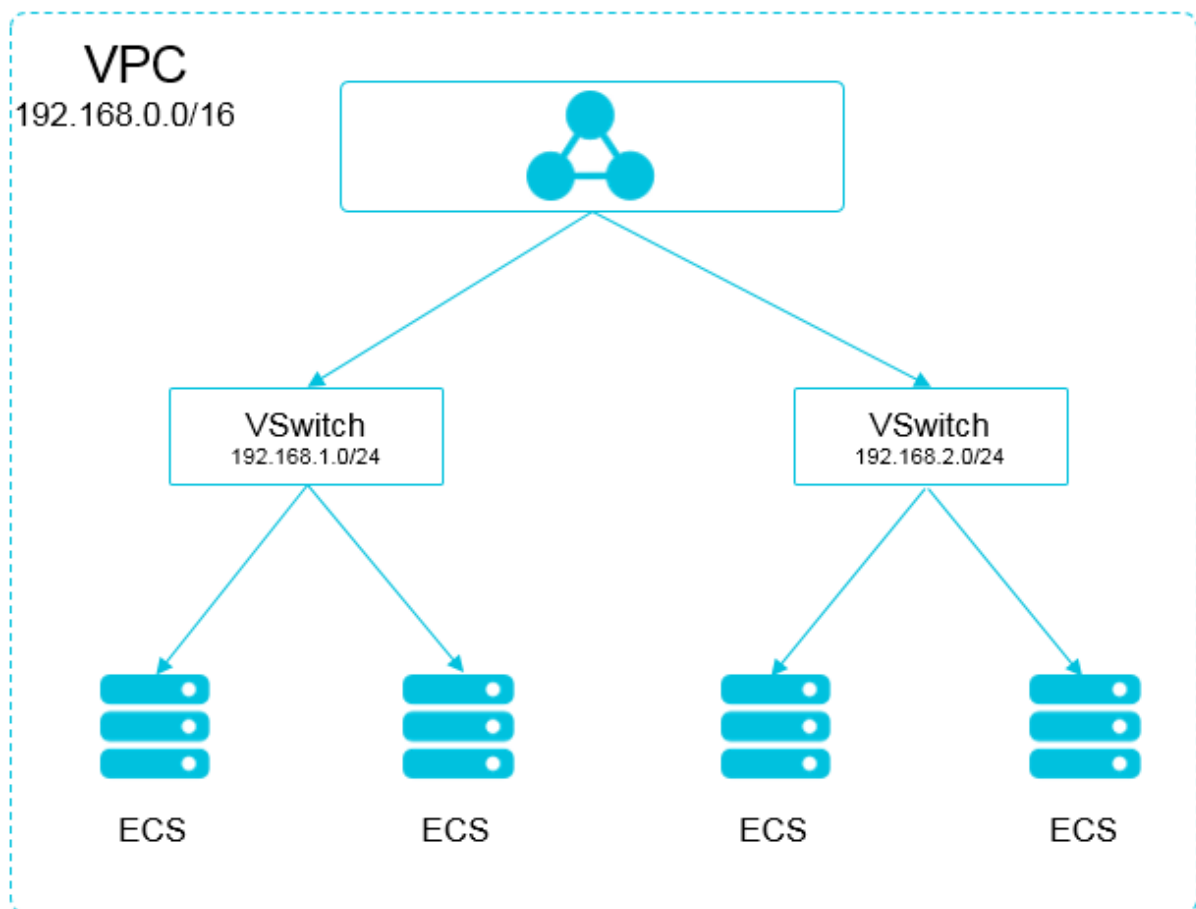
VPC CIDR block

The CIDR block selected when you create a VPC. Select the VPC CIDR block from 10.0.0.0/8, 172.16.0.0/12, and 192.168.0.0/16.

VSwitch CIDR block

The CIDR block specified when you create a VSwitch in VPC. The VSwitch CIDR block must be the subset of the current VPC CIDR block, which can be the same as the VPC CIDR block but cannot go beyond that range. The address assigned to the ECS instance under the VSwitch is obtained from the VSwitch CIDR block. Multiple VSwitches can be created under one VPC, but the VSwitch CIDR blocks cannot overlap.

The VPC CIDR block structure is as follows.



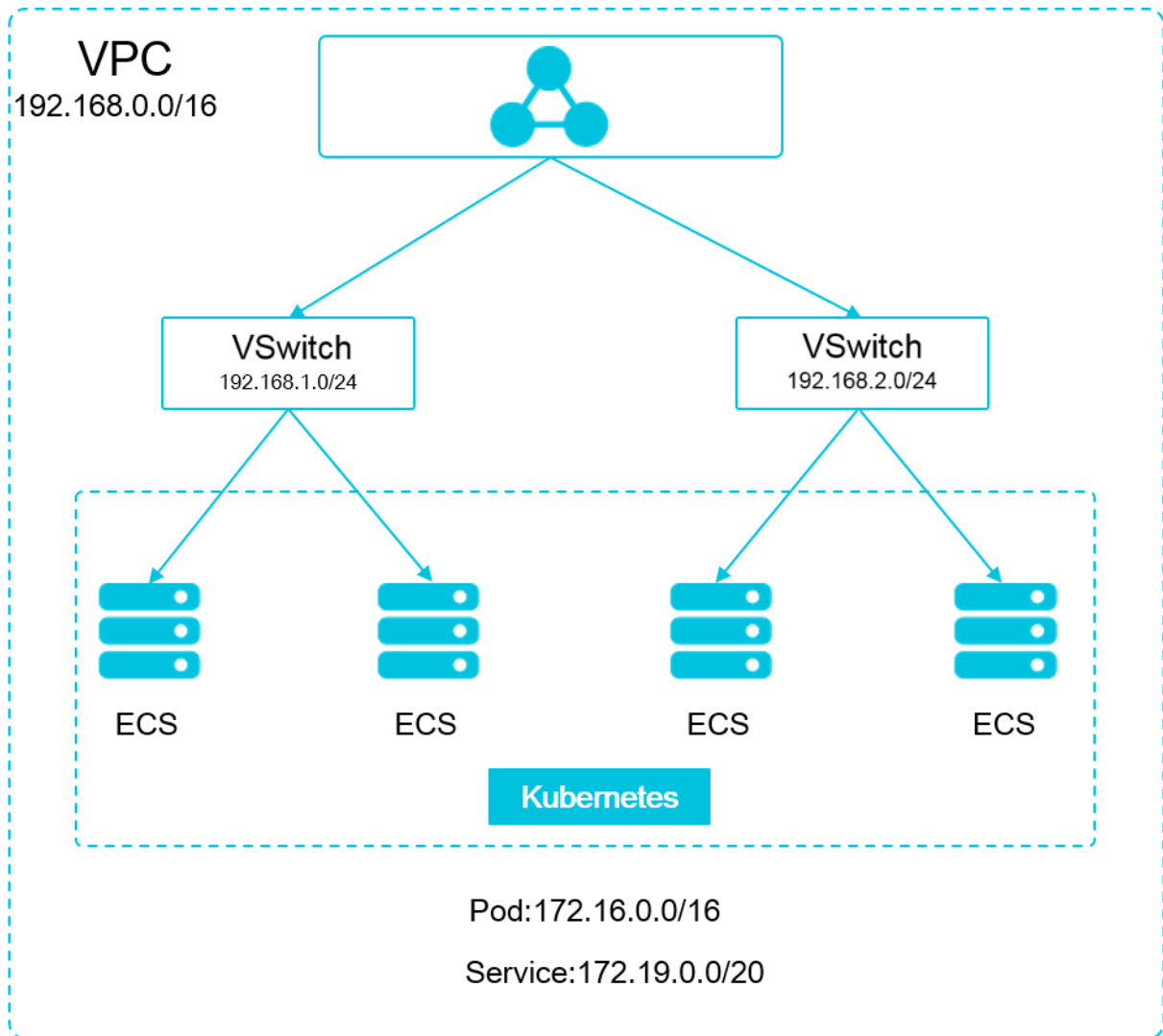
Pod CIDR block

Pod is a concept in Kubernetes. Each pod has one IP address. You can specify the pod CIDR block when creating a Kubernetes cluster in Alibaba Cloud Container Service and the pod CIDR block cannot overlap with the VPC CIDR block. For example, if the VPC CIDR block is 172.16.0.0/12, then the pod CIDR block of Kubernetes cannot use 172.16.0.0/16, 172.17.0.0/16, or any address that is included in 172.16.0.0/12.

Service CIDR block

Service is a concept in Kubernetes. Each service has its own address. The service CIDR block cannot overlap with the VPC CIDR block or pod CIDR block. The service address is only used in a Kubernetes cluster and cannot be used outside a Kubernetes cluster.

The relationship between Kubernetes CIDR block and VPC CIDR block is as follows.



How to select CIDR block

Scenario of one VPC and one Kubernetes cluster

This is the simplest scenario. The VPC address is determined when the VPC is created. Select a CIDR block different from that of the current VPC when creating a Kubernetes cluster.

Scenario of one VPC and multiple Kubernetes clusters

Create multiple Kubernetes clusters under one VPC. In the default network mode (Flannel), the pod message needs to be routed by using VPC, and Container Service automatically configures

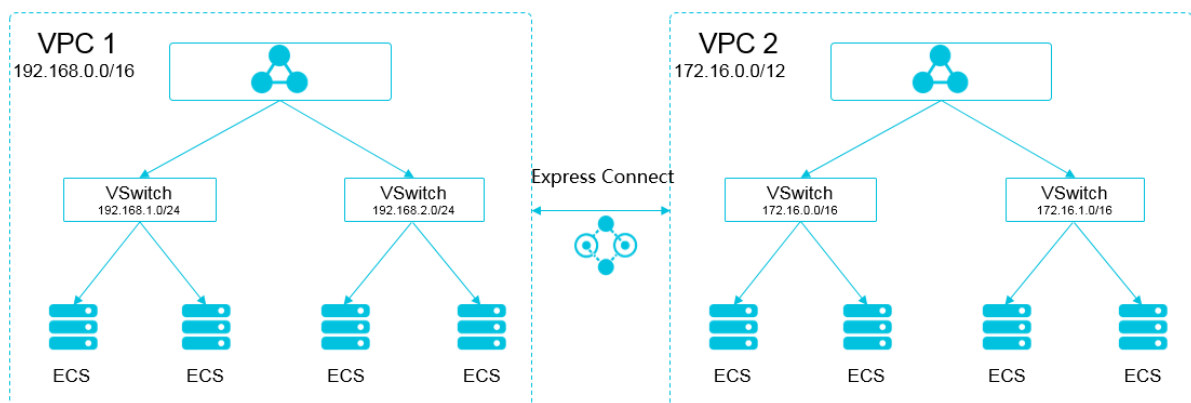
the route table to each pod CIDR block on the VPC route. The pod CIDR blocks of all the Kubernetes clusters cannot overlap, but the service CIDR blocks can overlap.

The VPC address is determined when the VPC is created. Select a CIDR block that does not overlap with the VPC address or other pod CIDR blocks for each Kubernetes cluster when creating a Kubernetes cluster.

In such a situation, parts of the Kubernetes clusters are interconnected. The pod of one Kubernetes cluster can directly access the pod and ECS instance of another Kubernetes cluster, but cannot access the service of another Kubernetes cluster.

Scenario of VPC interconnection

You can configure what messages are to be sent to the opposite VPC by using route tables when two VPCs are interconnected. Take the following scenario as an example: VPC 1 uses the CIDR block 192.168.0.0/16 and VPC 2 uses the CIDR block 172.16.0.0/12. By using route tables, specify to send the messages of 172.16.0.0/12 in VPC 1 to VPC 2.



In such a situation, the CIDR block of the Kubernetes cluster created in VPC 1 cannot overlap with VPC 1 CIDR block or the CIDR block to be routed to VPC 2. The same applies to the scenario when you create a Kubernetes cluster in VPC 2. In this example, the pod CIDR block of the Kubernetes cluster can select a sub-segment under 10.0.0.0/8.



Note:

The CIDR block routing to VPC 2 can be considered as an occupied address. Kubernetes clusters cannot overlap with an occupied address.

To access the Kubernetes pod of VPC 1 in VPC 2, configure the route to the Kubernetes cluster in VPC 2.

Scenario of VPC to IDC

Similar to the scenario of VPC interconnection, if parts of the CIDR blocks in VPC route to IDC, the pod address of Kubernetes clusters cannot overlap with those addresses. To access the pod address of Kubernetes clusters in IDC, configure the route table to leased line virtual border router (VBR) in IDC.

1.3.4 Connect to a Kubernetes cluster by using kubectl

To connect to a Kubernetes cluster from a client computer, use the Kubernetes command line client [kubectl](#).

Procedure

1. Download the latest kubectl client from the [Kubernetes release page](#).
2. Install and set the kubectl client.

For more information, see [Install and set kubectl](#).

3. Configure the cluster credentials.

You can use the `scp` command to safely copy the master node configurations from the `/etc/kubernetes/kube.conf` file on the master virtual machine of the Kubernetes cluster to the `$HOME/.kube/config` file (where the kubectl expected credentials reside) of the local computer.

- If you select Password in the Login field when creating the cluster, copy the kubectl configuration file in the following method:

```
mkdir $HOME/.kube
scp root@<master-public-ip>:/etc/kubernetes/kube.conf $HOME/.kube/
config
```

- If you select Key Pair in the Login field when creating the cluster, copy the kubectl configuration file in the following method:

```
mkdir $HOME/.kube
scp -i [the storage path of the .pem private key file on the local
machine] root@:/etc/kubernetes/kube.conf $HOME/.kube/config
```

You can check the cluster `master-public-ip` on the cluster information page.

- a) Log on to the [Container Service console](#).
- b) Under Kubernetes, click **Clusters** in the left-side navigation pane.
- c) Click **Manage** at the right of the cluster.

In the **Connection Information** section, view the Master node SSH IP address.

1.3.5 Access Kubernetes clusters by using SSH

If you select not to enable SSH access for Internet when creating the Kubernetes cluster, you cannot access the Kubernetes cluster by using SSH or connect to the Kubernetes cluster by using `kubectl`. To access the cluster by using SSH after creating the cluster, manually bind Elastic IP (EIP) to the Elastic Compute Service (ECS) instance, configure security group rules, and open the SSH port (22).

Procedure

1. Log on to the [Container Service console](#).
2. Under Kubernetes, click **Clusters** in the left-side navigation pane.
3. Click **Manage** at the right of the cluster.
4. In **Cluster resource**, click the ID of the Internet SLB. Then, you are redirected to the Instance Details page of your Internet Server Load Balancer instance.

Basic Information	
Cluster ID: c8bde9f83c0bce4ef3a7156879ed384e44	VPC ●Running Region: China East 1 (Hangzhou)
Connection Information	
API Server Internet endpoint	https://47.87.228.5:443
API Server Intranet endpoint	https://192.168.223.178:443
Master node SSH IP address	47.87.228.5
Service Access Domain	*c8bde9f83c0bce4ef3a7156879ed384e44.cn-hangzhou.aliyuncs.com
Cluster resource	
ROS	hls-hs-cs-c8bde9f83c0bce4ef3a7156879ed384e44
Internet SLB	ls-c8bce9ef0cc0a48f8gms1
VPC	vpc-6p3m8rt32t2gkdt2-w-mvict
NAT Gateway	ngw-6p32t2t3aaz373-0mev0d3c

5. Click **Listeners** in the left-side navigation pane and then click **Add Listener** in the upper-right corner.
6. Add the SSH listening rule.
 - a. **Front-end Protocol [Port]**: Select TCP and enter 22.
 - b. **Backend Protocol [Port]**: Enter 22.
 - c. Turn on the **Use Server Group** switch and select **VServer Group**.
 - d. **Server Group ID**: Select **sshVirtualGroup**.
 - e. Click **Next** and then click **Confirm** to create the listener.

Front-end Protocol [Port]:*

TCP

:

22

Port range is 1-65535.

Backend Protocol [Port]:*

TCP

:

22

Port range is 1-65535.

Peak Bandwidth:

No Limits

[Configure](#)

Instances charged by traffic are not limited by peak bandwidth. Peak bandwidth range is 1-5000.

Scheduling Algorithm:

Weighted Round Robin

Use Server Group:

☒

Server Group Type:

☒ VServer Group

☐ Master-Slave Server Group

Server Group ID:

sshVirtualGroup

Automatically Enable Listener After Creation:

☒ Enable

☐ [Show Advanced Options](#)

Next

Cancel

7. Then, you can use the Server Load Balancer instance IP address to access your cluster by using SSH.

Basic Information		
Server Load Balancer ID: lb-1ad5g9f0m2k09qast	Status: Running	
Server Load Balancer Name: kube-apiserver	Region: China East 1 (Hangzhou)	
Instance IP Type: Public IP	Zone: cn-hangzhou-b(Master)/cn-hangzhou-d(Slave)	
Network Type: Classic Network		
Billing Information		
Billing Method: Pay by Traffic	Created At: 2018-01-24 11:13:01	Billing Details Release
Instance IP Address: 114.55.111.25(Public IP)	Automatic Release Time: -	

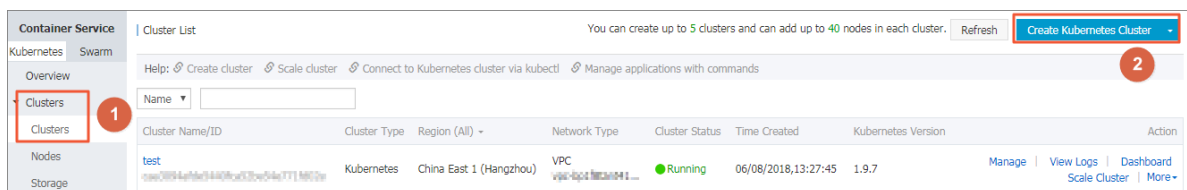
1.3.6 Access Kubernetes clusters by using SSH key pairs

Alibaba Cloud Container Service allows you to log on to clusters by using SSH key pairs, which guarantees the security of SSH remote access.

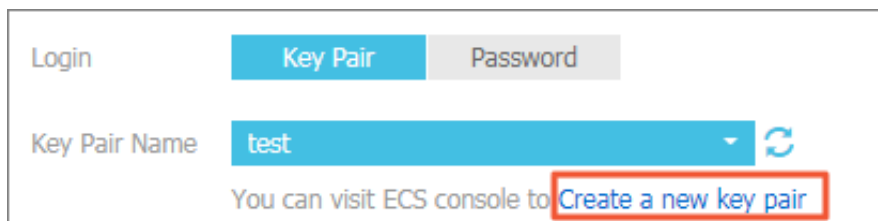
Context

Procedure

1. Log on to the [Container Service console](#).
2. Under Kubernetes, click **Clusters** in the left-side navigation pane.
3. Click **Create Kubernetes Cluster** in the upper-right corner.



4. Select Key Pair in the Login field. Complete the other configurations. For more information, see [Create a Kubernetes cluster](#). Then, click **Create**.
 1. If you have created key pairs in the Elastic Compute Service (ECS) console, select a key pair from the Key Pair Name drop-down list.
 2. If you have no key pair, click **Create a new key pair** to create one in the ECS console. For more information, see [Create an SSH key pair](#).



5. After the cluster is created, click **Manage** at the right of the cluster on the Cluster List page. View the **Master node SSH IP address** under Connection Information.

Instructions

- To upgrade the cluster, make sure your machine can access the Internet to download the necessary software packages.
- The upgrade may fail. We recommend that you back up snapshots before upgrading the cluster to guarantee your data security. For how to create a snapshot, see [Create snapshots](#).
- During the upgrade, your applications are not affected, but we recommend that you do not manage the cluster by using kubectl or the Container Service console. The upgrade lasts 5–15 minutes. The cluster status changes to Running after the upgrade.

Prerequisites

Check the health status of the cluster before upgrading the cluster. Make sure the cluster is healthy.

Log on to the master node. For more information, see [Access Kubernetes clusters by using SSH](#) and [Connect to a Kubernetes cluster by using kubectl](#).

1. Run the command `kubectl get cs`. Make sure all the modules are healthy.

```
NAME STATUS MESSAGE ERROR
scheduler Healthy ok
controller-manager Healthy ok
etcd-0 Healthy {"health": "true"}
etcd-1 Healthy {"health": "true"}
etcd-2 Healthy {"health": "true"}
```

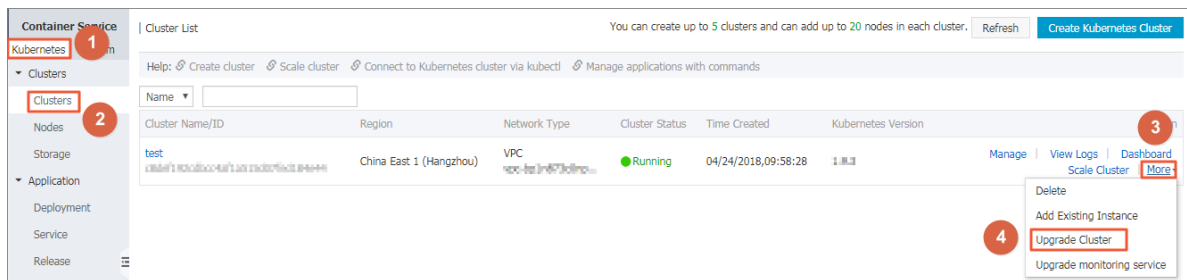
2. Run the command `kubectl get nodes`. Make sure all the nodes are in the Ready status.

```
kubectl get nodes
NAME STATUS ROLES AGE VERSION
cn-shanghai.i-xxxxxxx Ready master 38d v1.9.3
cn-shanghai.i-xxxxxxx Ready <none> 38d v1.9.3
cn-shanghai.i-xxxxxxx Ready <none> 38d v1.9.3
cn-shanghai.i-xxxxxxx Ready <none> 38d v1.9.3
cn-shanghai.i-xxxxxxx Ready master 38d v1.9.3
cn-shanghai.i-xxxxxxx Ready master 38d v1.9.3
```

If nodes are abnormal, you can fix them by yourself or open a ticket to ask Alibaba Cloud engineers to fix them.

Procedure

1. Log on to the [Container Service console](#).
2. Under Kubernetes, click **Clusters** in the left-side navigation pane.
3. Click **More > Upgrade Cluster** at the right of the cluster.



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

The system starts to upgrade the Kubernetes version.

After the upgrade, you can check the Kubernetes version of this cluster in the Kubernetes cluster list to make sure whether or not the upgrade is successful.

1.3.8 Scale out or in a cluster

In the Container Service console, you can scale out or scale in the worker nodes of a Kubernetes cluster according to your actual business requirements.

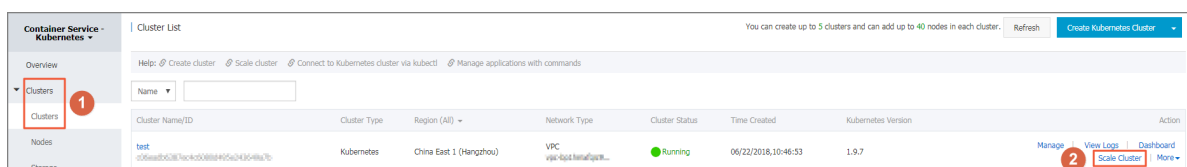
Context

Instructions

- Currently, Container Service only supports manually scaling in and out a cluster and does not support auto scaling.
- Currently, Container Service does not support scaling in and out the master nodes in a cluster.
- Container Service only supports scaling in the worker nodes that are created when you create the cluster or added after you scale out the cluster. The worker nodes that are added as existing [Add an existing ECS instance](#) when you create the cluster cannot be scaled in.
- When you scale in a cluster, the worker nodes are removed from the cluster in the order that they are added after you scale out the cluster.
- You must have more than 1 node that is not manually added to perform scaling in.

Procedure

1. Log on to the [Container Service console](#).
2. Under Container Service Kubernetes, click **Clusters** in the left-side navigation pane.
3. Click **Scale Cluster** at the right of the cluster.



4. Select **Scale out** or **Scale in** in the Scale field and then configure the number of worker nodes.

In this example, scale out the cluster and change the number of worker nodes from one to four.

Cluster Name	k8s-test	
Region	China East 1 (Hangzhou) ZoneG	
Existing	1	
Scale	<div>Scale out</div> <div>Scale in</div>	
Instance Type	<div>2 Core(s) 4 G (ecs.n4.large)</div>	
Scaling Number	<div>3 unit(s)</div>	
Number of workers after scaling:	4	
* Logon Password	<div>*****</div>	Please enter the login password used when creating the cluster
RDS Whitelist	Select RDS Instances	
<div>Submit</div>		

5. Enter the logon password of the node.

**Note:**

Make sure this password is the same as the one you entered when creating the cluster because you have to log on to the Elastic Compute Service (ECS) instance to copy the configuration information in the upgrade process.

6. Click **Submit**.

What's next

After scaling is complete, go to the Kubernetes Clusters Node List page to view that the number of worker nodes changes from one to four.

1.3.9 Cluster auto scaling

Configure a cluster to auto scale according to the cluster load.

Prerequisites

You have created a Kubernetes cluster successfully. For how to create a Kubernetes cluster, see [Create a Kubernetes cluster](#).

Background

Cluster auto scaling is different from [Scale out or in a cluster](#) that is based on resource thresholds. After auto scaling is configured for a cluster, the cluster automatically scales out or scales in when the cluster load reaches the configured value.

Procedure

Enable cluster auto scaling

1. Log on to the [Container Service console](#).
2. Under Kubernetes, click **Clusters** in the left-side navigation pane.
3. Select a cluster and click **More > Auto Scaling** in the Action column.

Cluster Name/ID	Cluster Type	Region (All)	Network Type	Cluster Status	Time Created	Kubernetes Version	Action
mymanagedk8s1 ce2de2005ab5b4c839d690d9ddda1714	ManagedKubernetes	China North 2 (Beijing)	VPC vpc-2zetk0nc8jr...	Running	09/07/2018,16:01:46	1.10.4	Manage View Logs Dashboard Scale Cluster More
myserverlessk8s1 cb767c92315b7498c8bb5bac58e4a854d	Serverless Kubernetes	China East 1 (Hangzhou)	VPC vpc-bp1bw6ox1cy...	Running	09/04/2018,20:13:20	1.9.7	Manage View Logs Delete
mytest1 c68b3c5a4273c4fba8151f06276bcc224	Kubernetes	China North 2 (Beijing)	VPC vpc-2ze7c50jcu...	Running	09/03/2018,10:06:49	1.10.4	Manage View Logs Dashboard Scale Cluster More
myk8s1 c1001d096f1bd4bebb17037d35ae12a24	Kubernetes	China East 1 (Hangzhou)	VPC vpc-bp1faadnm5o...	Running	08/26/2018,11:53:17	1.10.4	Manage Delete Add Existing Instance Upgrade Cluster Auto-scaling Addon Upgrade Upgrade monitoring service Deploy Istio

Authorization

• Activate Auto Scaling service

1. Click the first link in the displayed dialog box.

Note

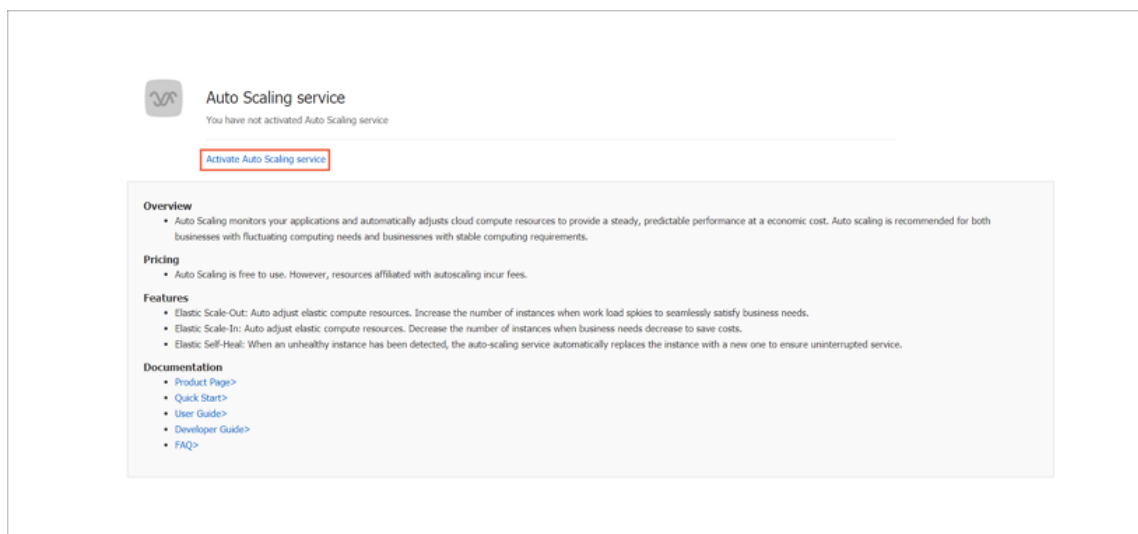
Auto-scaling relies on the ESS service. Before enabling auto-scaling, you need to:

- 1 Enable the service and complete the default role authorization: [ESS](#)
- 2 Jump to RAM to add an ESS authorization policy to the current cluster: [View detailed steps](#)
[KubernetesWorkerRole](#)

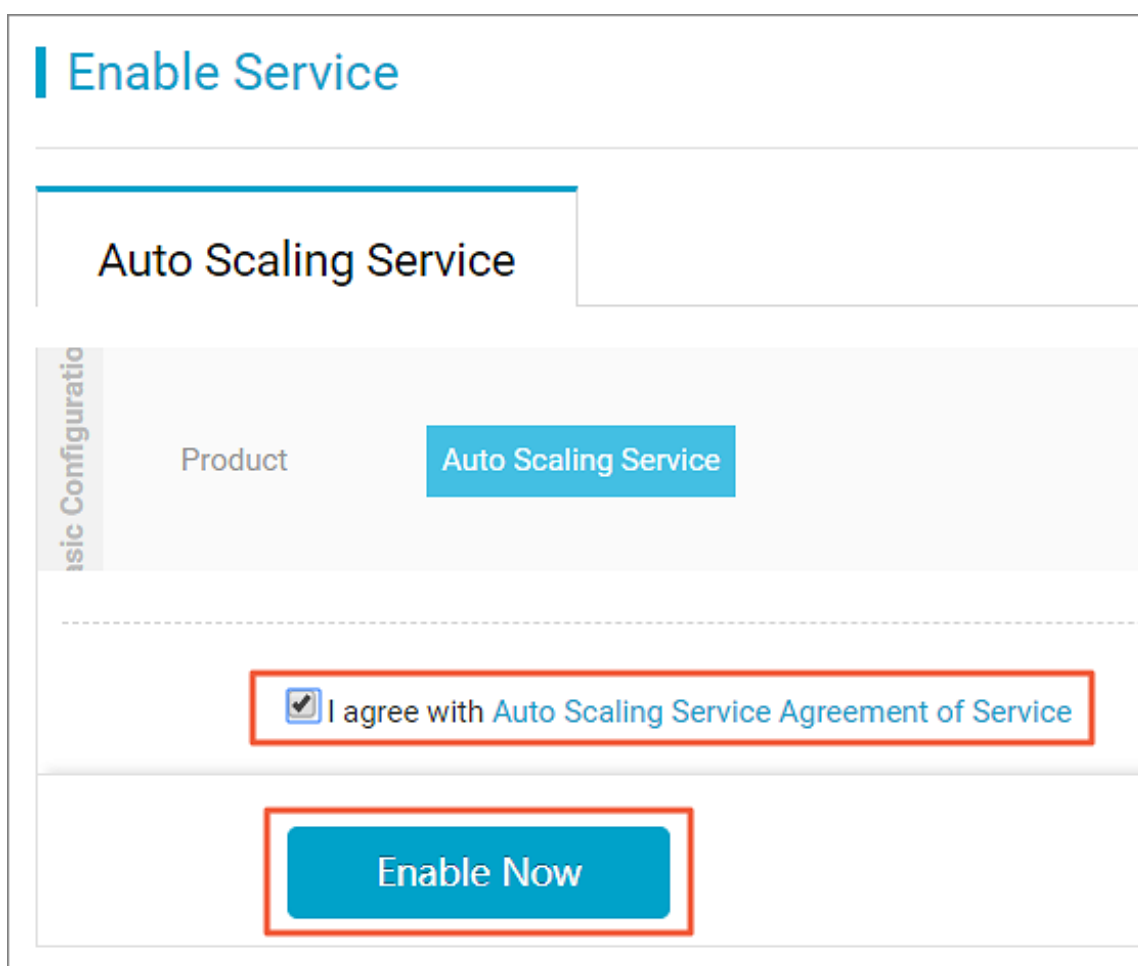
Please confirm the above steps, otherwise the Auto-scaling will not be enabled.

Confirm

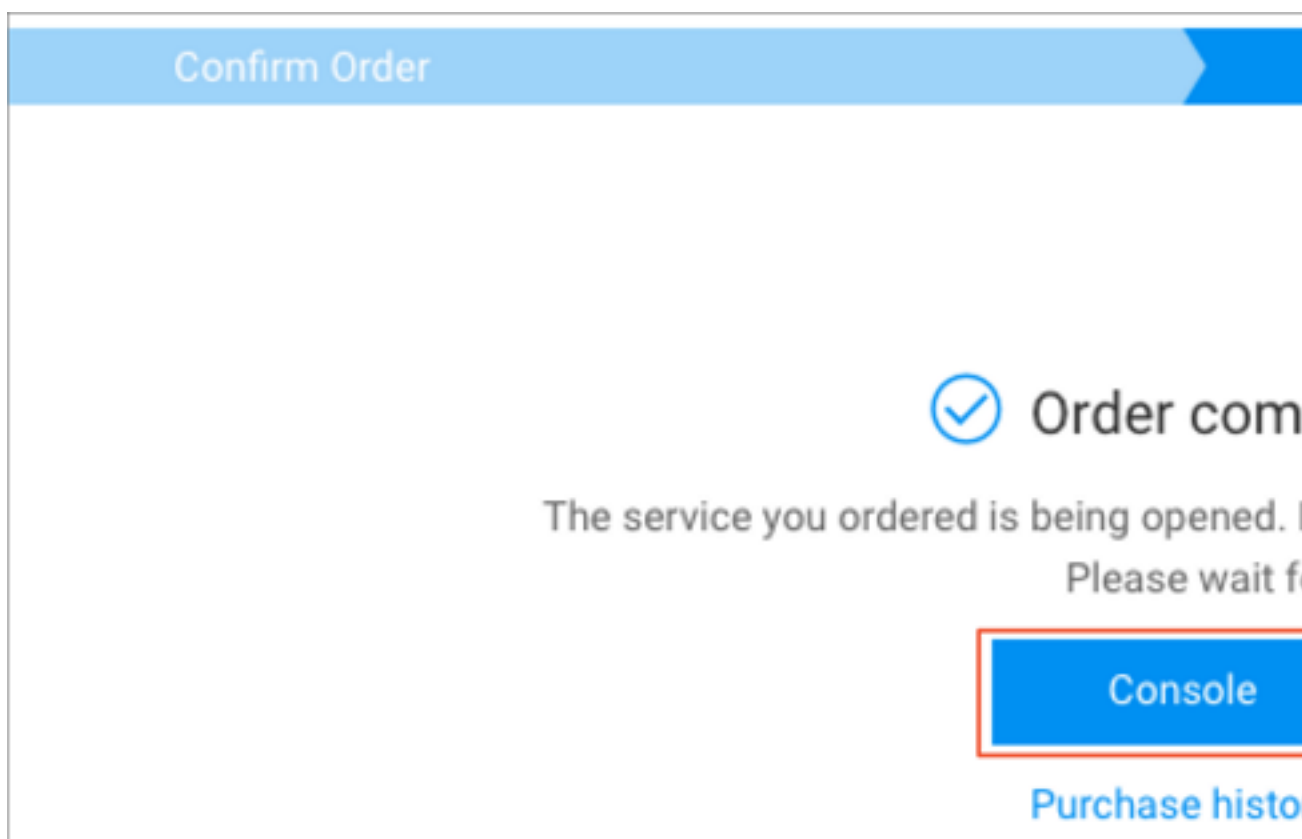
2. Click **Activate Auto Scaling service**.



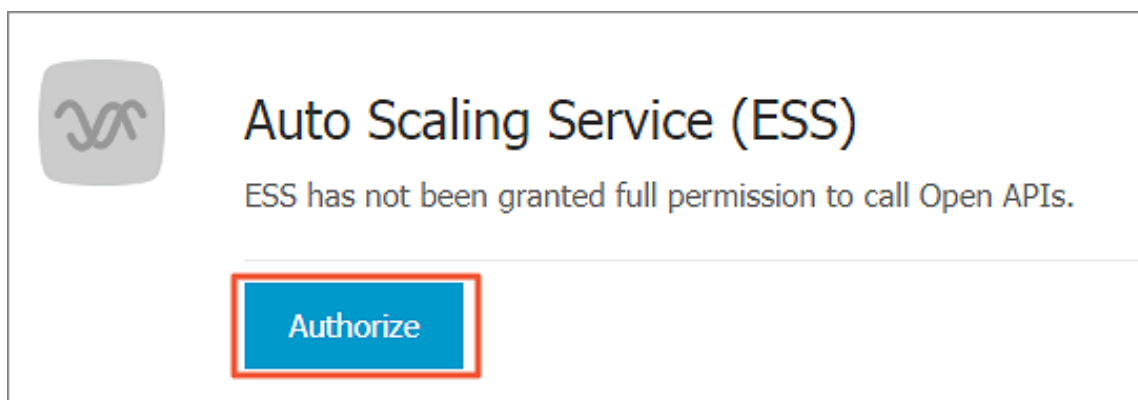
3. Select the **I agree with Auto Scaling Service Agreement of Service** check box and click **Enable Now**.



4. After the service is activated, click **Console**.



5. Click **Authorize**. Configure permissions for accessing cloud resources on the **Cloud Resource Access Authorization** page.



6. Click **Confirm Authorization Policy**.

Cloud Resource Access Authorization

Note: If you need to modify role permissions, please go to the RAM Console. [Role Management](#). If you do not configure it correctly, the following role: ESS will not be able to obtain the required permissions. ✕

ESS needs your permission to access your cloud resources.
 Authorize ESS to use the following roles to access your cloud resources.

AliyunESSDefaultRole ✓

Description: The ESS service will use this role to run ECS instances.

Permission Description: The policy for AliyunESSDefaultRole, including the permission for ECS.

[Confirm Authorization Policy](#) [Cancel](#)

Expected result

When the page automatically jumps to the elastic scaling console, the authorization succeeds.
 Closes the page and continues to configure **Authorize roles**.

- **Authorize a role.**

1. Click the second link in the displayed dialog box.



Note:

Use the primary account to log on to the console.

Note ✕

Auto-scaling relies on the ESS service. Before enabling auto-scaling, you need to:

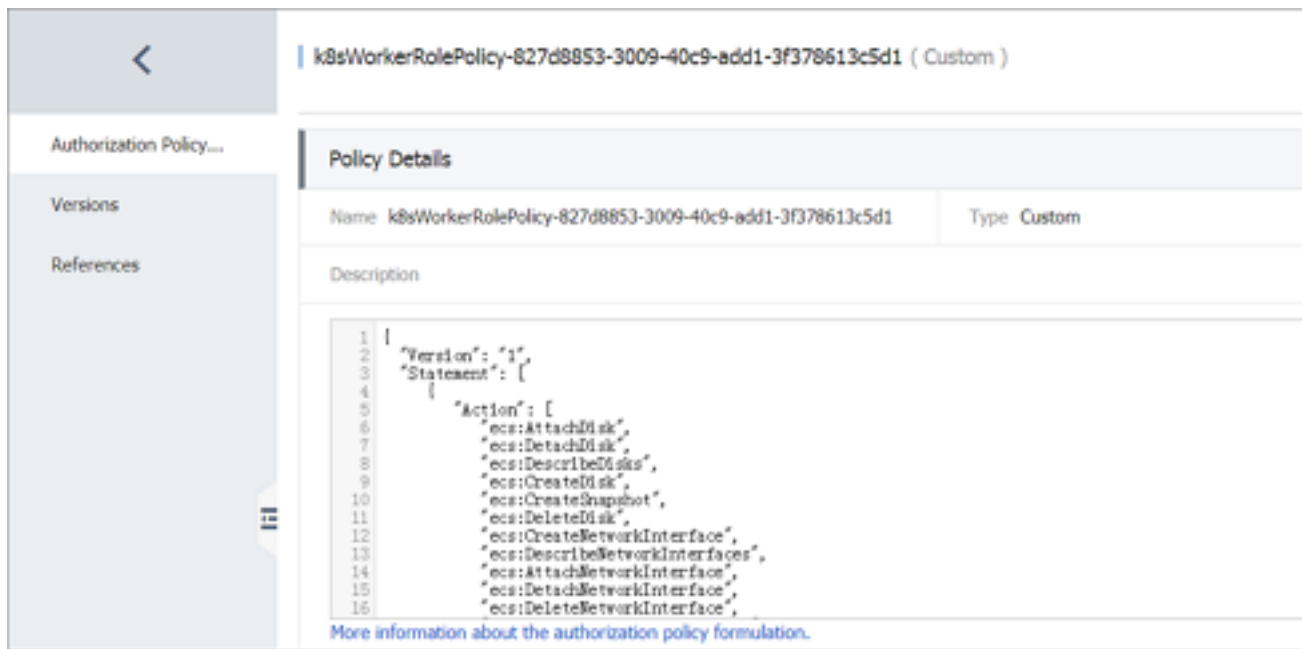
- 1 Enable the service and complete the default role authorization: [ESS](#)
- 2 Jump to RAM to add an ESS authorization policy to the current cluster: [View detailed steps](#)
[KubernetesWorkerRole](#)

Please confirm the above steps, otherwise the Auto-scaling will not be enabled.

[Confirm](#)

2. Select the target authorization policy and click **View Permissions** in the Action column.

3. Click **Modify Authorization Policy** in the upper-right corner of the page.



4. In the **Action** field of **Policy content**, enter the following:

```
"ess:Describe*",
"ess:CreateScalingRule",
"ess:ModifyScalingGroup",
"ess:RemoveInstances",
"ess:ExecuteScalingRule",
"ess:ModifyScalingRule",
"ess>DeleteScalingRule",
"ecs:DescribeInstanceTypes",
"ess:DetachInstances"
```



Note:

Add a comma (,) to the last line in the **Action** field before entering the preceding content.

5. Click **Modify Authorization**.

Configure cluster auto scaling

1. On the **Auto-scaling** page, configure the following parameters:

Configuration	Description
Cluster	The target cluster name.
Shrinkage Threshold	The ratio of the amount of resources requested by the cluster load to the amount of cluster resources. When the amount of resource requested by the cluster load is less than or equal to the configured shrinkage

Configuration	Description
	threshold, the cluster automatically shrinks. The default is 50%.
Shrinkage Trigger Delay	When the configured shrinkage threshold is reached and the configured shrinkage trigger delay expires, the cluster starts to shrinks. Unit: minute The default is 10 minutes.
Cooldown Time	After cluster expansion or shrinkage, the cooldown time starts to count. During the cooldown time, adding nodes to or removing nodes from the cluster does not trigger the cluster to expand or shrink again. The default is 10 minutes.



2. Select a resource type (CPU or GPU) to be scaled, click **Create** in the Action column.

On the **Scaling Group Config** page, configure the following parameters to create a scaling group:

Configuration	Description
Region	The region to which the created scaling group is deployed. This region must be consistent with the region of the cluster in which the scaling group resides. This region cannot be changed.
Zone	The zone of the created scaling group.
VPC	The network of the created scaling group, which must be consistent with the network of the cluster in which the scaling group resides.

Configure worker nodes

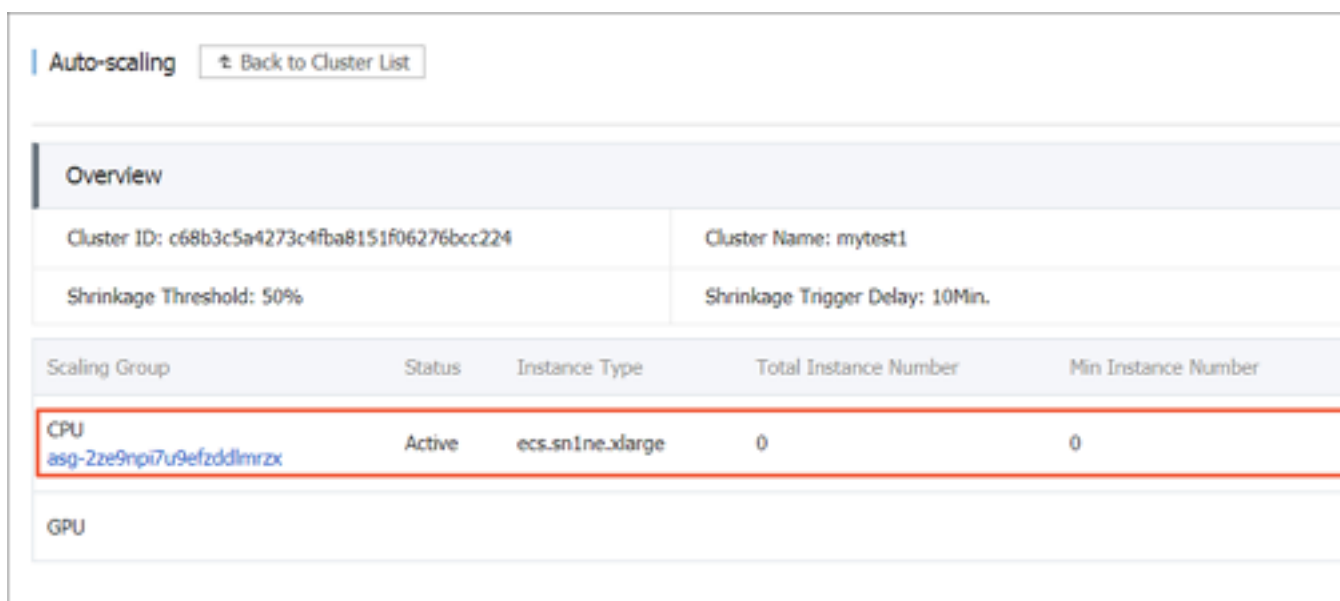
Configuration	Description
Instance Type	Types of instances in the scaling group.

Configuration	Description
System Disk	The system disk of the scaling group.
Attach Data Disk	Whether or not to mount a data disk when you create a scaling group. By default, no data disk is not mounted.
Instance Quantity	<p>The number of instances contained by the scaling group.</p> <div> Note:<ul style="list-style-type: none">Existing instances are not included.By default, the minimum number of instances is 0. When the number of instances exceeds 0, the cluster adds an instance to the scaling group and adds the instance into the Kubernetes cluster in which the scaling group resides by default.</div>
Key Pair	<p>The key pair used to log on to the scaled node. You can create a new key pair on the Elastic Compute Service (ECS) console.</p> <div> Note:<p>Only key pair logon is supported currently.</p></div>
RDS whitelist	The Relational Database Service (RDS) instance that can be accessed by a scaled node.

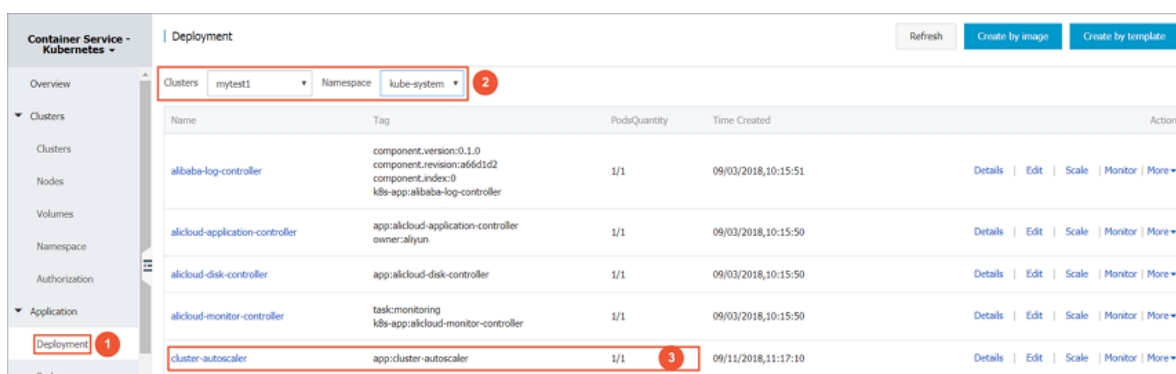
3. Click **OK** to create a scaling group.

Expected result

- A scaling group is displayed under CPU on the **Auto-scaling** page.



- 1. Click **Application > Deployment** in the left-side navigation pane.
- 2. Select the target cluster and the kube-system namespace, you can view the created component named cluster-autoscaler.

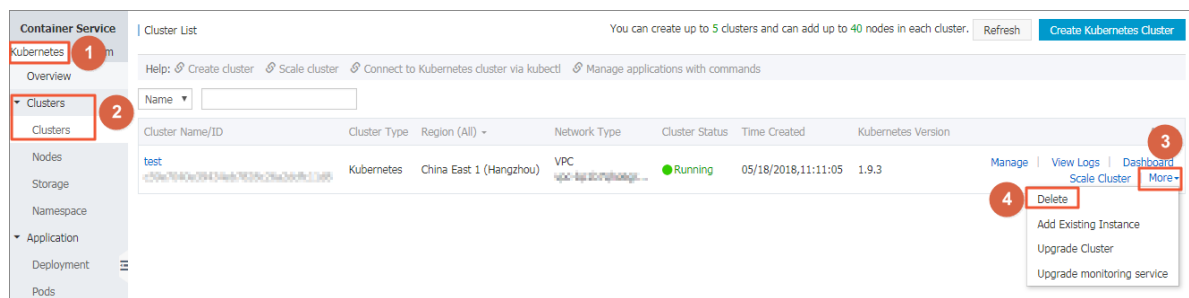


1.3.10 Delete a cluster

In the Container Service console, you can delete clusters that are no longer in use.

Procedure

1. Log on to the [Container Service console](#).
2. Under Container Service Kubernetes, click **Clusters** in the left-side navigation pane.
3. Click **More > Delete** and select the target cluster.



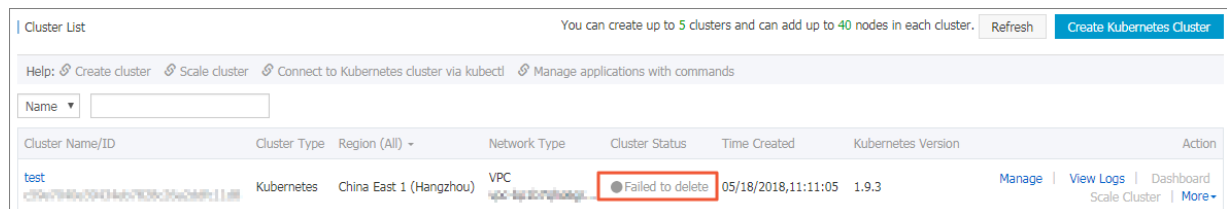
What's next

Failed to delete a cluster

If you manually add some resources under the resources created by Resource Orchestration Service (ROS), ROS does not have permissions to delete the manually added resources. For example, manually add a VSwitch under the Virtual Private Cloud (VPC) created by ROS. ROS fails to process this VPC when deleting the Kubernetes resources and then the cluster fails to be deleted.

Container Service allows you to force delete the cluster. You can force delete the cluster record and ROS stack if the cluster fails to be deleted. However, you must release the created resources manually.

The cluster status is Failed if the cluster fails to be deleted.

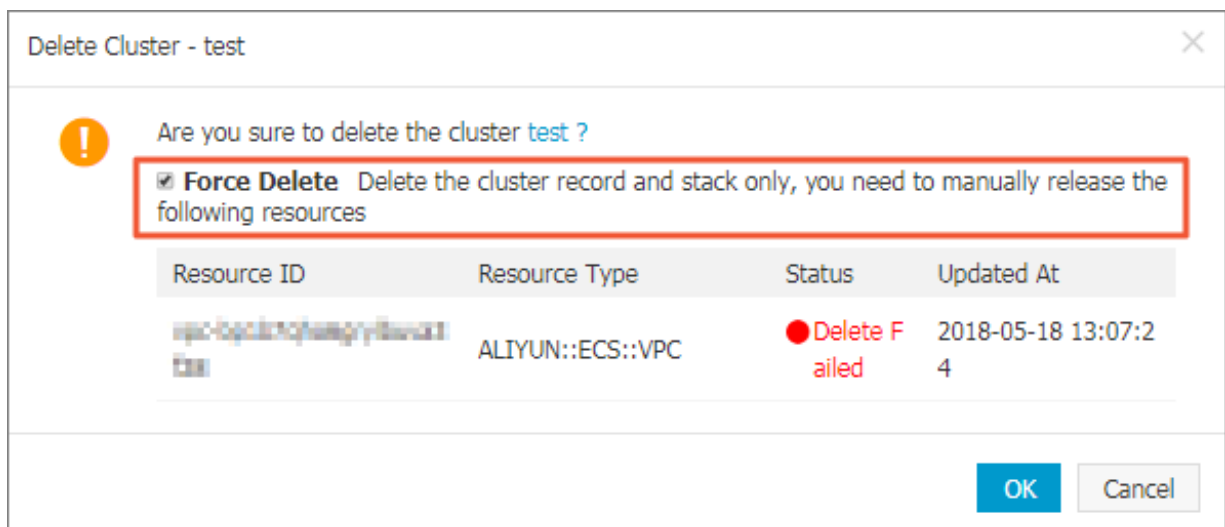


Click **More** > **Delete** in the displayed dialog box, you can see the resource that failed to delete, check force delete, and click **OK**. The cluster and ROS resource can be deleted.



Note:

You must manually release the resources that failed to be deleted. For information on how to troubleshoot the problem with resources that cannot be released, see [Failed to delete Kubernetes clusters: ROS stack cannot be deleted](#).



1.4 Nodes

1.4.1 Add an existing ECS instance

You can add existing Elastic Compute Service (ECS) instances to a created Kubernetes cluster. Currently, Kubernetes clusters only support adding worker nodes.

Prerequisites

- If you have not created a cluster before, create a cluster first. For how to create a cluster, see [Create a Kubernetes cluster](#).
- Add the ECS instance to the security group of the Kubernetes cluster first.

Context

Instructions

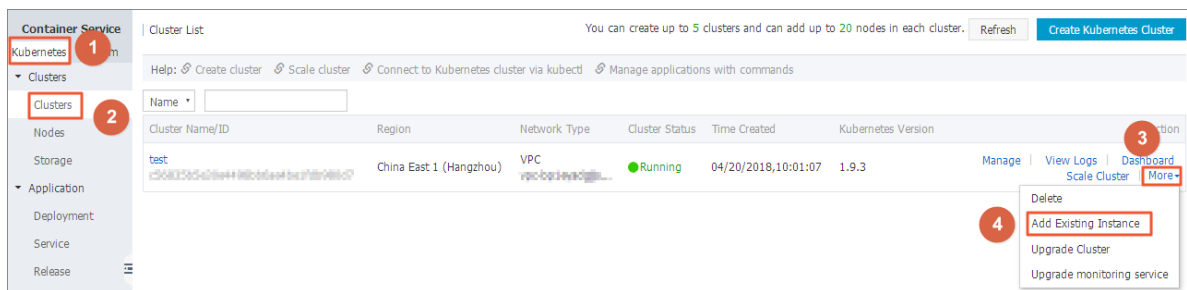
- By default, each cluster can contain up to 40 nodes. To add more nodes, open a ticket.
- The ECS instance to be added must be in the same Virtual Private Cloud (VPC) region as the cluster.
- When adding an existing instance, make sure that your instance has an Elastic IP (EIP) for the VPC network type, or the corresponding VPC is already configured with the NAT gateway. In short, make sure the corresponding node can access public network normally. Otherwise, the ECS instance fails to be added.
- The ECS instance to be added must be under the same account as the cluster.
- Only nodes with a CentOS operating system are supported.

Procedure

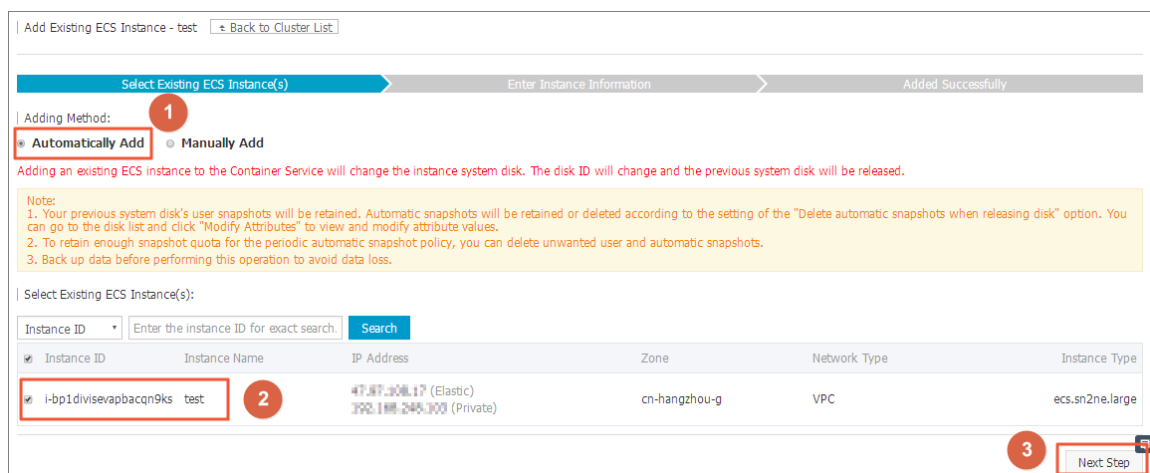
1. Log on to the [Container Service console](#).
2. Under Kubernetes, click **Clusters** in the left-side navigation pane.
3. Select the target cluster and click **More > Add Existing Instance**.

On the Add Existing ECS Instance page and you can automatically or manually add an existing instance.

If Automatically Add is selected, select the ECS instances to add them to the cluster automatically. If Manually Add is selected, you must obtain the command and then log on to the corresponding ECS instance to add the ECS instance to this cluster. You can only add one ECS instance at a time.



4. Select Automatically Add to add multiple ECS instances at a time.
 - a) In the list of existing cloud servers, select the target ECS instance, and then click **Next Step**.



- b) Enter the instance information, set the logon password, and then click **Next Step**.

选择已有云服务器实例

填写实例信息

添加完成

集群ID/名称: / test-gpu
当前要添加的集群信息

登录方式: ☒ 设置密码

* 密码:
密码为8-30个字符, 必须同时包含三项 (大、小写字母, 数字和特殊符号), 不支持\两个符号

* 确认密码:

实例信息:

实例ID	实例名称
shukun-ECS	test

上一步 下一步

- c) In the displayed dialog box, click **OK**, the selected ECS instance is automatically added to the cluster.

Confirm adding existing instance to cluster

! Are you sure you want to add the selected ECS instance to the cluster `shukun-ECS` / test?

Adding an existing ECS instance will change the system disk. The disk ID will change and the previous system disk will be released.

1. After the system disk is changed, the user snapshots on the previous system disk will be retained, but the automatic snapshots will be released with the disk.

2. To retain enough snapshot quota for the automatic snapshot policy of the new disk, you can delete unwanted snapshots.

3. Back up data before performing this operation to avoid data loss. Alibaba Cloud is not liable for any data losses caused by your failure to back up personal data in the system.

Confirm Cancel

5. You can also select **Manually Add** to manually add an existing ECS instance to the cluster.

- a) Select the ECS instance to be added and then click **Next Step**. You can only add one ECS instance at a time.

Add Existing ECS Instance - test [Back to Cluster List](#)

Select Existing ECS Instance(s) Enter Instance Information Added Successfully

Adding Method:
☐ Automatically Add ☒ **Manually Add** 1

Select Existing ECS Instance(s):

To manually add existing nodes, you can only select one ECS instance at a time.

Instance ID Enter the instance ID for exact search.

Instance ID	Instance Name	IP Address	Zone	Network Type	Instance Type
<input checked="" type="checkbox"/> i-bp1divisevabacqn9ks	test	47.101.101.17 (Elastic) 192.168.1.100 (Private)	cn-hangzhou-g	VPC	ecs.sn2ne.large

3

b) Confirm the information and then click **Next Step**.

Add Existing ECS Instance - test [Back to Cluster List](#)

Select Existing ECS Instance(s) Enter Instance Information Added Successfully

Cluster ID/Name : / test
 Information of the cluster to which to add the ECS instance(s).

Instance Information :

Instance ID	Instance Name
i-bp1divisevabacqn9ks	test

c) Go to the Add Existing ECS Instance page and copy the command.

Add Existing ECS Instance - test [Back to Cluster List](#)

Select Existing ECS Instance(s) Enter Instance Information Added Successfully

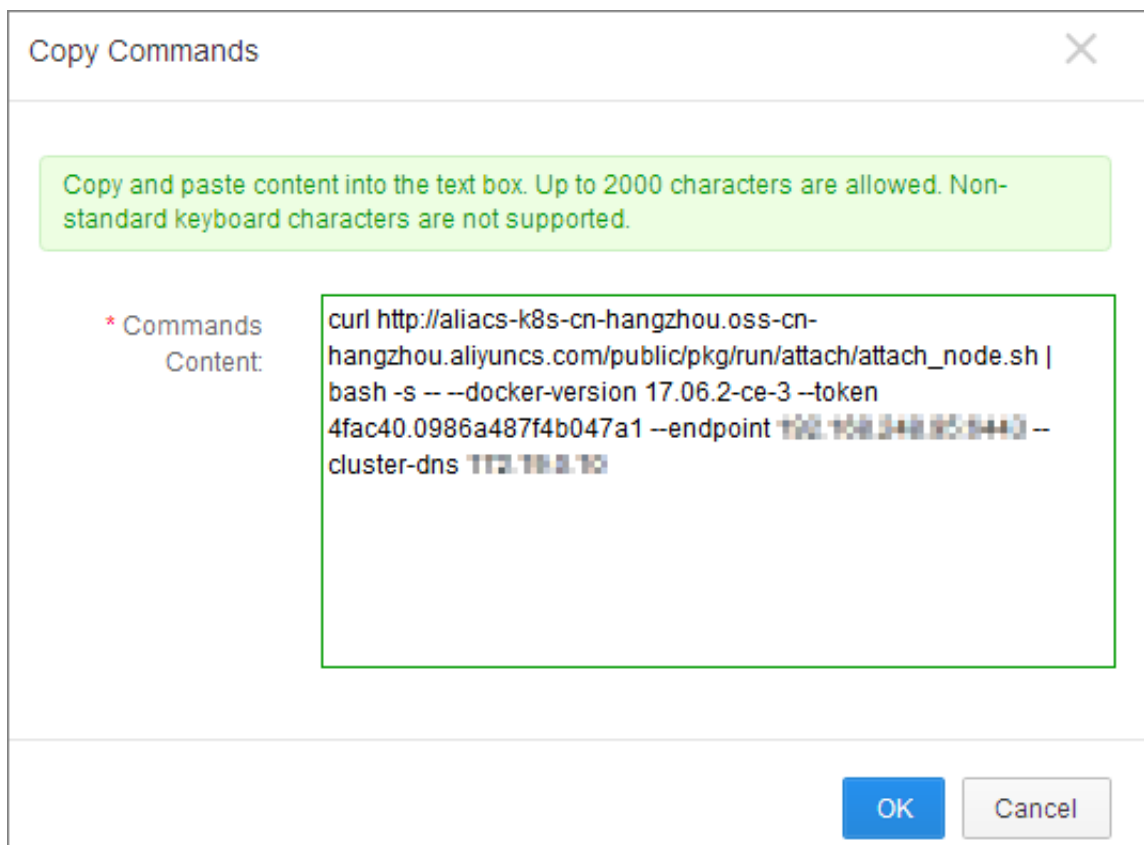
Only supports adding nodes in the same VPC with CentOS operating system

Log in to the node you want to add, execute the following command:

```
curl http://aliacs-k8s-cn-hangzhou.oss-cn-hangzhou.aliyuncs.com/public/pkg/run/attach/attach_node.sh | bash -s -- --docker-version 17.06.2-c
e-3 --token 4fac40.0986a487f4b047a1 --endpoint 192.168.1.100 --cluster-dns 172.17.0.1
```

d) Log on to the [ECS console](#). Select the region in which the cluster resides.

e) Click **Connect** at the right of the ECS instance to be added. The Enter VNC Password dialog box appears. Enter the VNC password and then click **OK**. Enter the copied command and then click **OK** to run the script.



- f) After the script is successfully run, the ECS instance is added to the cluster. You can click the cluster ID on the Cluster List page to view the node list of the cluster and check if the ECS instance is successfully added to the cluster.

1.4.2 View node list

You can view the node list of the Kubernetes cluster by using commands, in the Container Service console, or in the Kubernetes dashboard.

View node list by using commands



Note:

Before using commands to view the node list of the Kubernetes cluster, [Connect to a Kubernetes cluster by using kubectl](#) first.

After connecting to the Kubernetes cluster by using kubectl, run the following command to view the nodes in the cluster:

```
kubectl get nodes
```

Sample output:

```
$ kubectl get nodes
```

```

NAME STATUS AGE VERSION
iz2ze2n6ep53tch701yh9zz Ready 19m v1.6.1-2+ed9e3d33a07093
iz2zeaf762wibijx39e5az Ready 7m v1.6.1-2+ed9e3d33a07093
iz2zeaf762wibijx39e5bz Ready 7m v1.6.1-2+ed9e3d33a07093
iz2zef4dnn9nos8elyr32kz Ready 14m v1.6.1-2+ed9e3d33a07093
iz2zeitvvo8enoreufstkmz Ready 11m v1.6.1-2+ed9e3d33a07093

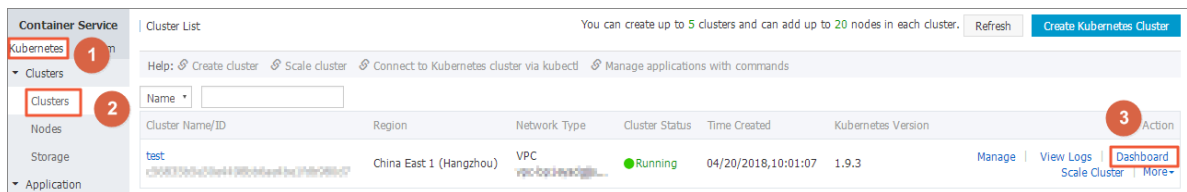
```

View node list in Container Service console

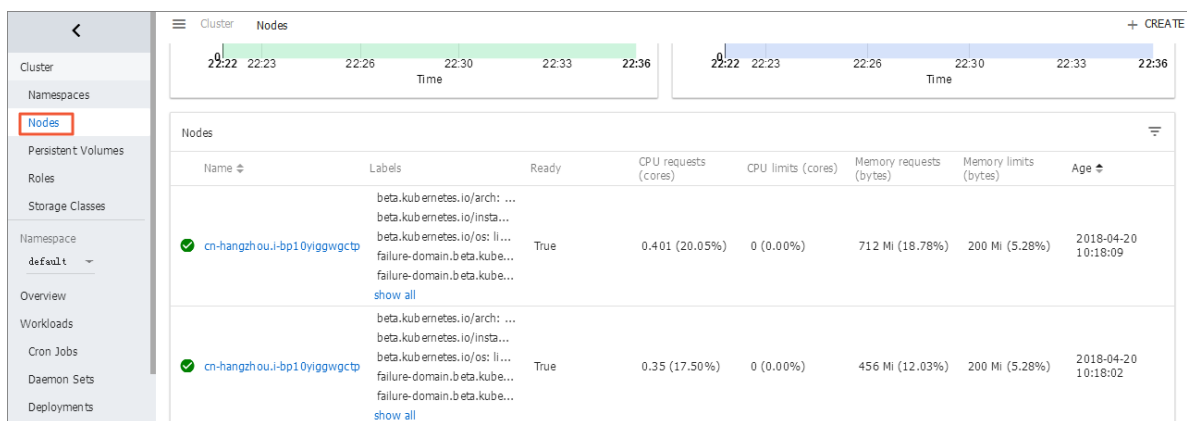
1. Log on to the [Container Service console](#).
2. Click **Kubernetes > Clusters > > Nodes** in the left-side navigation pane.
3. Select the cluster from the Cluster drop-down list and then view the node list of this cluster.

View node list in Kubernetes dashboard

1. Log on to the [Container Service console](#).
2. Click **Kubernetes > Clusters** in the left-side navigation pane.
3. Click **Dashboard** at the right of the cluster to enter the Kubernetes dashboard.



4. In the Kubernetes dashboard, click **Nodes** in the left-side navigation pane to view the node list of this cluster.



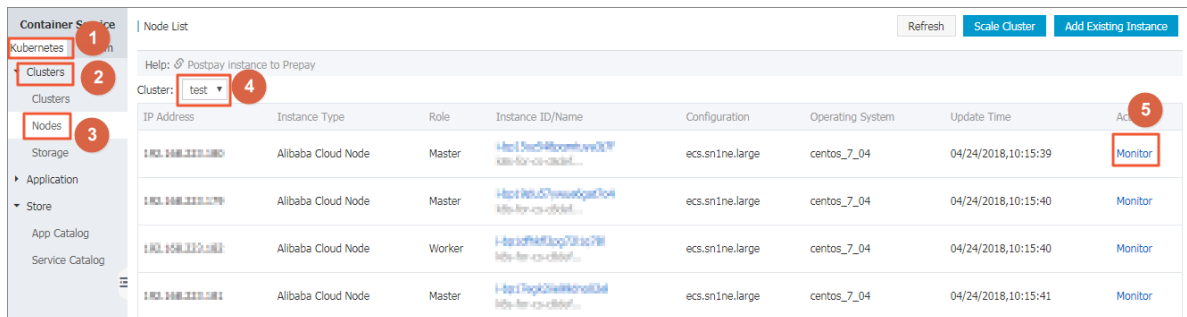
1.4.3 Node monitoring

Kubernetes clusters integrate with the Alibaba Cloud monitoring service seamlessly. You can view the monitoring information of Kubernetes nodes and get to know the node monitoring metrics of the Elastic Compute Service (ECS) instances under Kubernetes clusters.

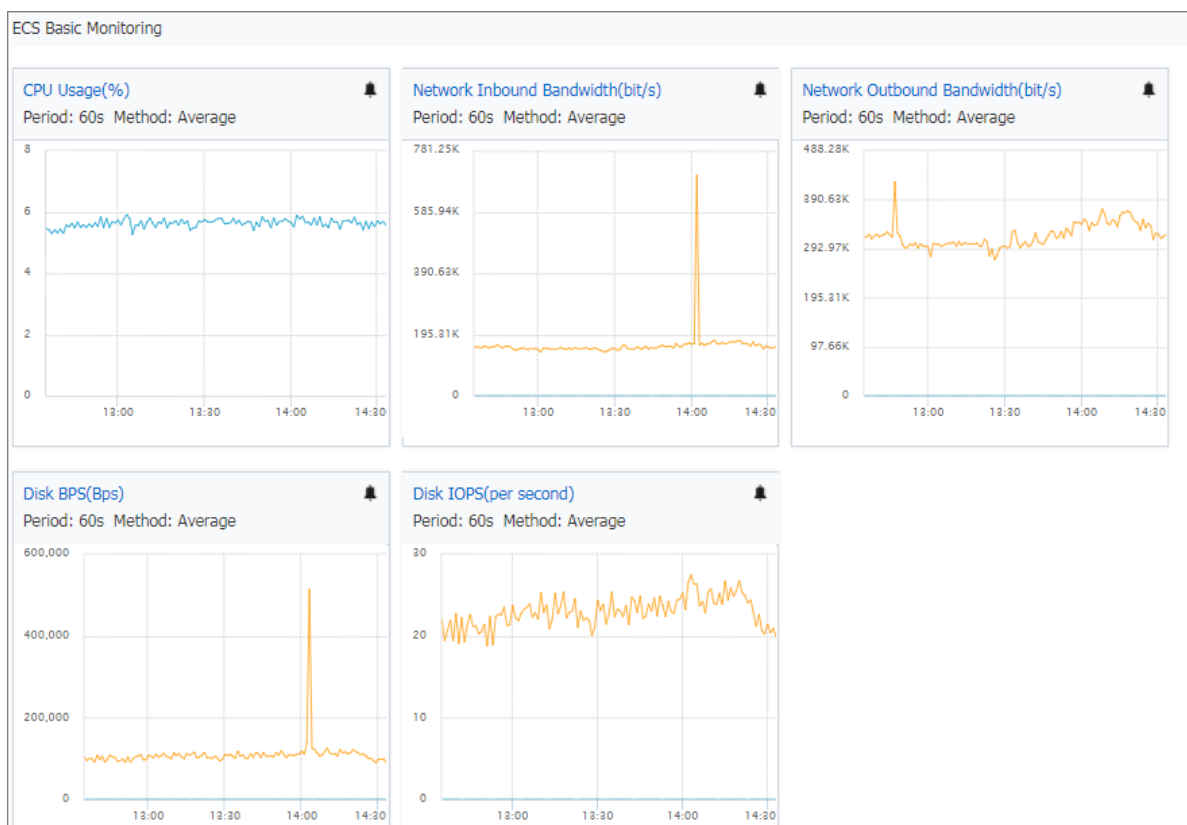
Procedure

1. Log on to the [Container Service console](#).

2. Under Kubernetes, click **Clusters** > **Nodes** to enter the Node List page.
3. Select the target cluster and node under the cluster.
4. Click **Monitor** at the right of the node to view the monitoring information of this node.



5. You are redirected to the CloudMonitor console. View the basic monitoring information of the corresponding ECS instance, including the CPU usage, network inbound bandwidth, network outbound bandwidth, disk BPS, and disk IOPS.



What's next

To view the monitoring metrics at the operating system level, install the CloudMonitor component. For more information, see [Introduction to Host monitoring](#).

Kubernetes clusters can now monitor resources by using application groups. For more information, see [Monitor resources and send alarm notifications by using resource groups](#).

1.4.4 Manage node labels

You can manage node labels in the Container Service console, including adding node labels in batches, filtering nodes by using a label, and deleting a node label quickly.

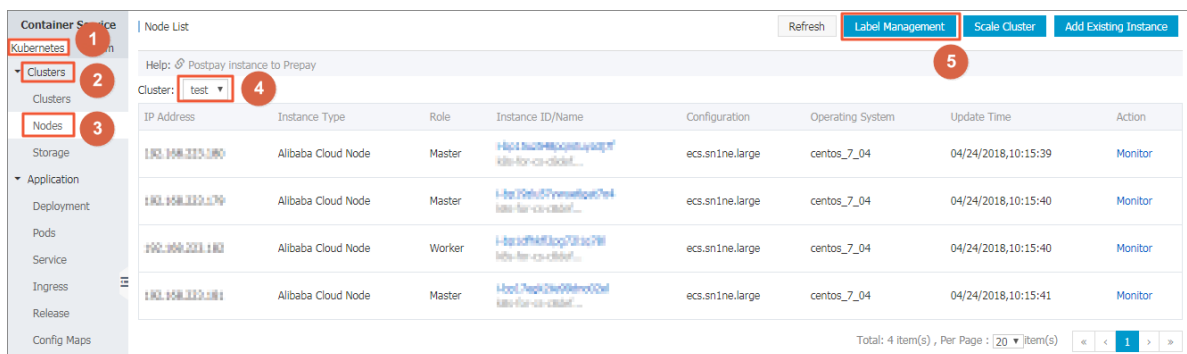
For how to use node labels to schedule pods to specified nodes, see [Schedule a pod to a specified node](#).

Prerequisite

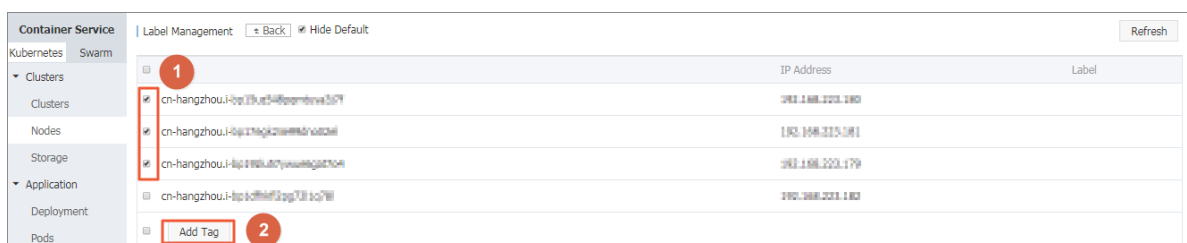
You have successfully created a Kubernetes cluster. For more information, see [Create a Kubernetes cluster](#).

Add node labels in batches

1. Log on to the [Container Service console](#).
2. Click **Kubernetes Clusters** > **Nodes** in the left-side navigation pane.
3. Select the cluster from the Clusters drop-down list and then click **Label Management** in the upper-right corner.



4. Select one or more nodes by selecting the corresponding check boxes and then click **Add Tag**.



5. Enter the name and value of the label in the displayed dialog box and then click **OK**.

Add

Name

group

Value

master

OK

Close

Nodes with the same label are displayed on the Label Management page.

<div>Container Service</div> <div>Kubernetes Swarm</div> <div>Clusters</div> <div>Nodes</div> <div>Storage</div> <div>Application</div> <div>Deployment</div> <div>Pods</div> <div>Service</div>	Label Management Back Hide Default Refresh		
	Name	IP Address	Label
	cn-hangzhou-1-bp1-5nch440pemtara27f	100.100.200.100	group : master
	cn-hangzhou-1-bp1-7nqk31w9wemo0del	100.100.200.100	group : master
	cn-hangzhou-1-bp1-8nch37vneue6gat7o4	100.100.200.100	group : master

Filter nodes by using a label

1. Log on to the [Container Service console](#).
2. Click **Kubernetes Clusters > Nodes** in the left-side navigation pane.
3. Select the cluster from the Clusters drop-down list and then click **Label Management** in the upper-right corner.

<div>Container Service</div> <div>Kubernetes</div> <div>Clusters</div> <div>Nodes</div> <div>Storage</div> <div>Application</div> <div>Deployment</div> <div>Pods</div> <div>Service</div> <div>Ingress</div> <div>Release</div> <div>Config Maps</div>	Node List Refresh Label Management Scale Cluster Add Existing Instance		
	Help: Postpay Instance to Prepay		
	Cluster: test		
	IP Address	Instance Type	Role
	100.100.200.100	Alibaba Cloud Node	Master

4. Click the label at the right of a node to filter nodes by using the label. In this example, click `group:worker`.

Nodes with the label `group:worker` are filtered.

Label Management

Back

Hide Default

group:master

Refresh

Name	IP Address	Label
cn-hangzhou-1-k8s-1-master-00000000000000000000	192.168.223.160	group : master
cn-hangzhou-1-k8s-1-master-00000000000000000000	192.168.223.161	group : master
cn-hangzhou-1-k8s-1-master-00000000000000000000	192.168.223.179	group : master

Add Tag

Delete a node label

1. Log on to the [Container Service console](#).
2. Click Kubernetes **Clusters** > **Nodes** in the left-side navigation pane.
3. Select the cluster from the Clusters drop-down list and then click **Label Management** in the upper-right corner.

Container Service

Kubernetes

Clusters

Nodes

Storage

Application

Deployment

Pods

Service

Ingress

Release

Config Maps

Node List

Refresh

Label Management

Scale Cluster

Add Existing Instance

Help: [Postpay Instance to Prepay](#)

Cluster: test

IP Address	Instance Type	Role	Instance ID/Name	Configuration	Operating System	Update Time	Action
193.148.215.180	Alibaba Cloud Node	Master	i-h41nuc84gcnuyagwtf k8s-for-ali-cloud...	ecs.sn1ne.large	centos_7_04	04/24/2018,10:15:39	Monitor
193.148.215.179	Alibaba Cloud Node	Master	i-h41nuc84gcnuyagwtf k8s-for-ali-cloud...	ecs.sn1ne.large	centos_7_04	04/24/2018,10:15:40	Monitor
193.148.215.183	Alibaba Cloud Node	Worker	i-h41nuc84gcnuyagwtf k8s-for-ali-cloud...	ecs.sn1ne.large	centos_7_04	04/24/2018,10:15:40	Monitor
193.148.215.181	Alibaba Cloud Node	Master	i-h41nuc84gcnuyagwtf k8s-for-ali-cloud...	ecs.sn1ne.large	centos_7_04	04/24/2018,10:15:41	Monitor

Total: 4 item(s), Per Page: 20 item(s)

4. Click the delete (x) button of a node label, for example, `group:worker`.

Label Management ← Back ☑ Hide Default Refresh		
Name	IP Address	Label
cn-hangzhou.i-instance-4pazv4y2g7f	192.168.223.100	group : master
cn-hangzhou.i-instance-4pazv4y2g7f	192.168.223.101	group : master
cn-hangzhou.i-instance-4pazv4y2g7f	192.168.223.102	group : master
cn-hangzhou.i-instance-4pazv4y2g7f	192.168.223.103	group : worker
Add Tag		

Click Confirm in the displayed dialog box. The node label is deleted.

Label Management Back Hide Default Refresh

Name	IP Address	Label
cn-hangzhou.i-instance-2h9gndvutj7n1	192.168.203.188	group : master
cn-hangzhou.i-instance-2h9gndvutj7n1	192.168.203.189	group : master
cn-hangzhou.i-instance-2h9gndvutj7n1	192.168.203.179	group : master
cn-hangzhou.i-instance-2h9gndvutj7n1	192.168.203.180	

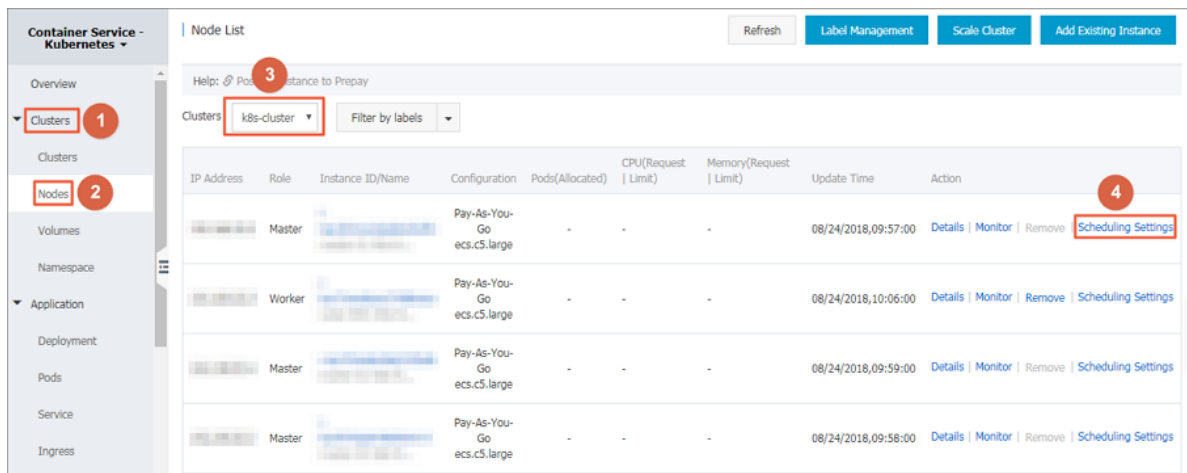
Add Tag

1.4.5 Set node scheduling

You can set node scheduling through the web interface so that you can allocate loads to each node properly.

Procedure

1. Log on to the [Container Service console](#).
2. Under Kubernetes, click **Clusters** > **Nodes** to enter the Node List page.
3. Select a cluster, select a node under the cluster, and click **Schedule Settings** on the right.

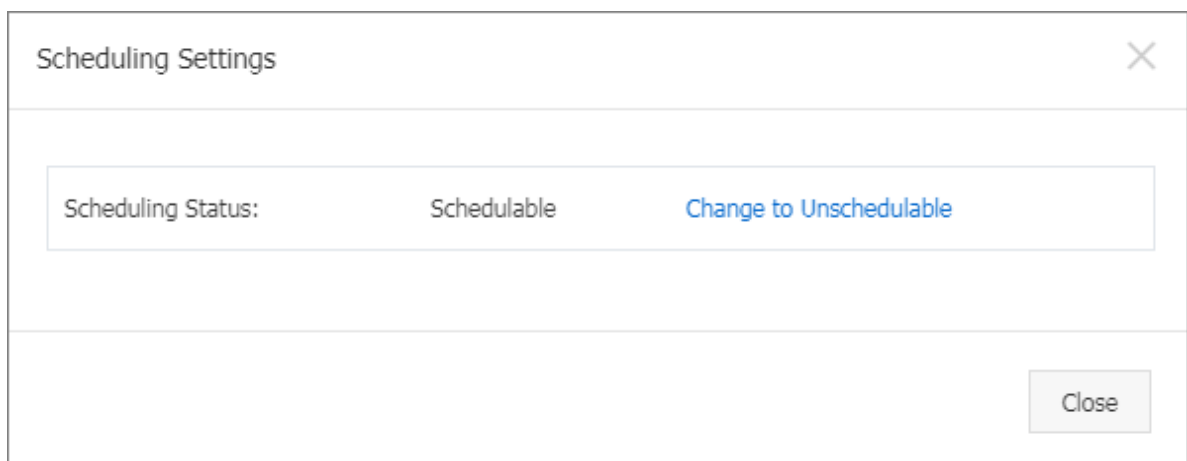


4. Set node scheduling in the displayed dialog box. In this example, click **Change to Unscheduleable** to set the node to unscheduleable.

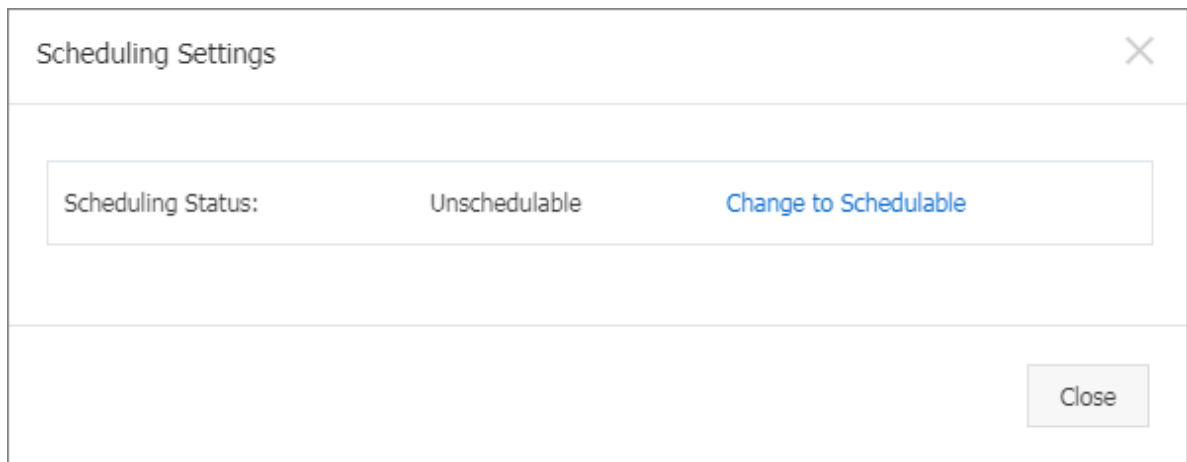


Note:

The scheduling status of the current node is displayed in the Scheduling Settings dialog box, which is schedulable by default. You can change the status.



After the status is set, the scheduling status of the node changes in the dialog box.



What's next

When you deploy your application later, you can find that pods are not scheduled to the node.

1.5 Namespaces

1.5.1 Create a namespace

Prerequisites

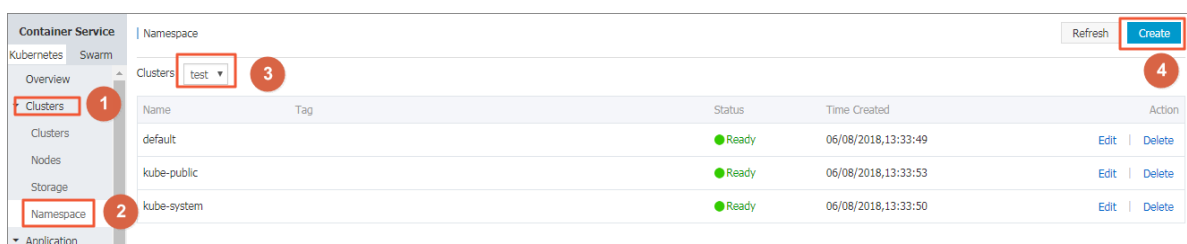
You have created a Kubernetes cluster. For more information, see [Create a Kubernetes cluster](#).

Context

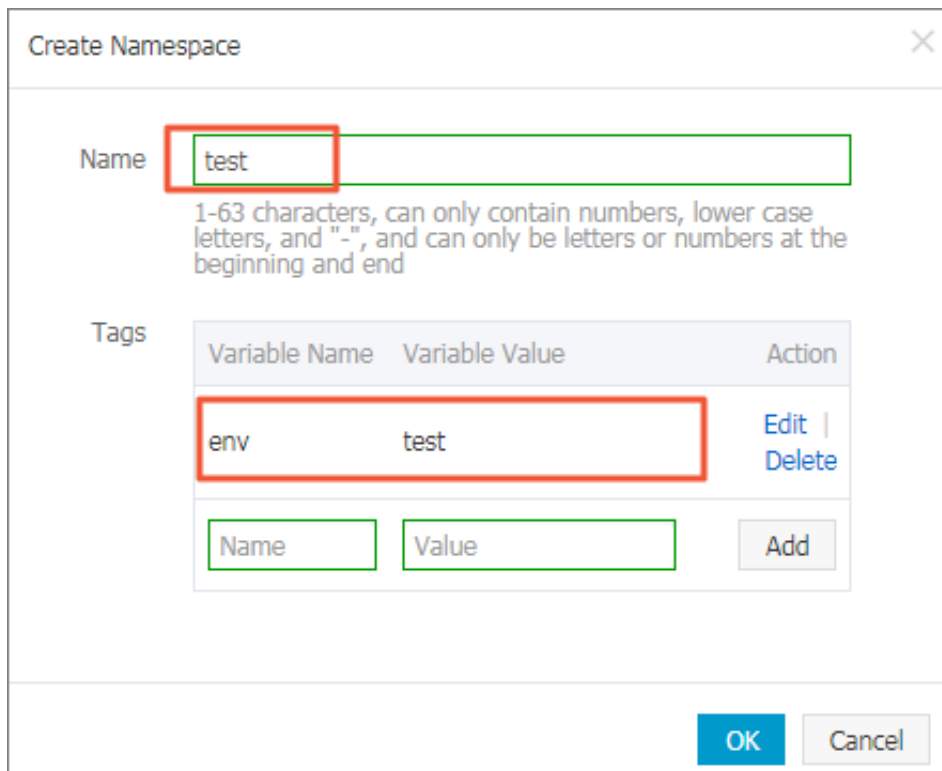
In Kubernetes clusters, you can use the Namespace function to create multiple virtual spaces. When the number of users in one cluster is large, multiple namespaces are used to divide the workspaces effectively and the cluster resources into different purposes. The namespace resources are assigned by using the [resource-quotas](#).

Procedure

1. Log on to the [Container Service console](#).
2. Under Kubernetes, click **Clusters** > **Namespace** in the left-side navigation pane.
3. Select the cluster from the Clusters drop-down list and then click **Create** in the upper-right corner.



4. Configure the namespace in the displayed dialog box.



The dialog box titled "Create Namespace" contains the following elements:

- Name:** A text input field containing "test". Below it, a note states: "1-63 characters, can only contain numbers, lower case letters, and '-', and can only be letters or numbers at the beginning and end".
- Tags:** A table with columns "Variable Name", "Variable Value", and "Action".

Variable Name	Variable Value	Action
env	test	Edit Delete

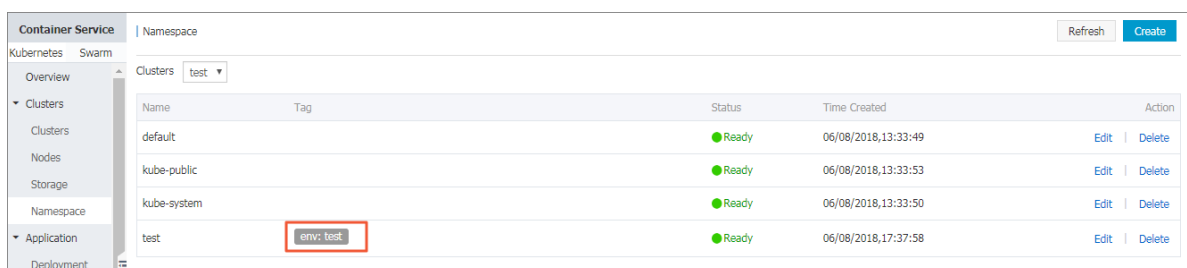
 Below the table are two input fields labeled "Name" and "Value", and an "Add" button.
- Buttons:** "OK" and "Cancel" buttons at the bottom right.

- **Name:** Enter the namespace name, which is 1–63 characters long, can only contain numbers, letters, and hyphens (-), and must start and end with a letter or number. In this example, enter test as the name.
- **Tags:** Add one or more tags for the namespace to identify the characteristics of the namespace, for example, to identify this namespace to be used for the test environment.

You can enter the variable name and variable value, and then click **Add** on the right to add a tag for the namespace.

5. Click **OK** after completing the configurations.

6. The namespace test is successfully created and displayed in the namespace list.



Name	Tag	Status	Time Created	Action
default		Ready	06/08/2018,13:33:49	Edit Delete
kube-public		Ready	06/08/2018,13:33:53	Edit Delete
kube-system		Ready	06/08/2018,13:33:50	Edit Delete
test	env: test	Ready	06/08/2018,17:37:58	Edit Delete

1.5.2 Configure resource quotas for namespaces

Prerequisites

- You have created a Kubernetes cluster. For more information, see [Create a Kubernetes cluster](#).
- You have created a namespace **test**. For more information, see [Create a namespace](#).
- Connect to the master node SSH IP address of the cluster. For more information, see [Access Kubernetes clusters by using SSH](#).

Context

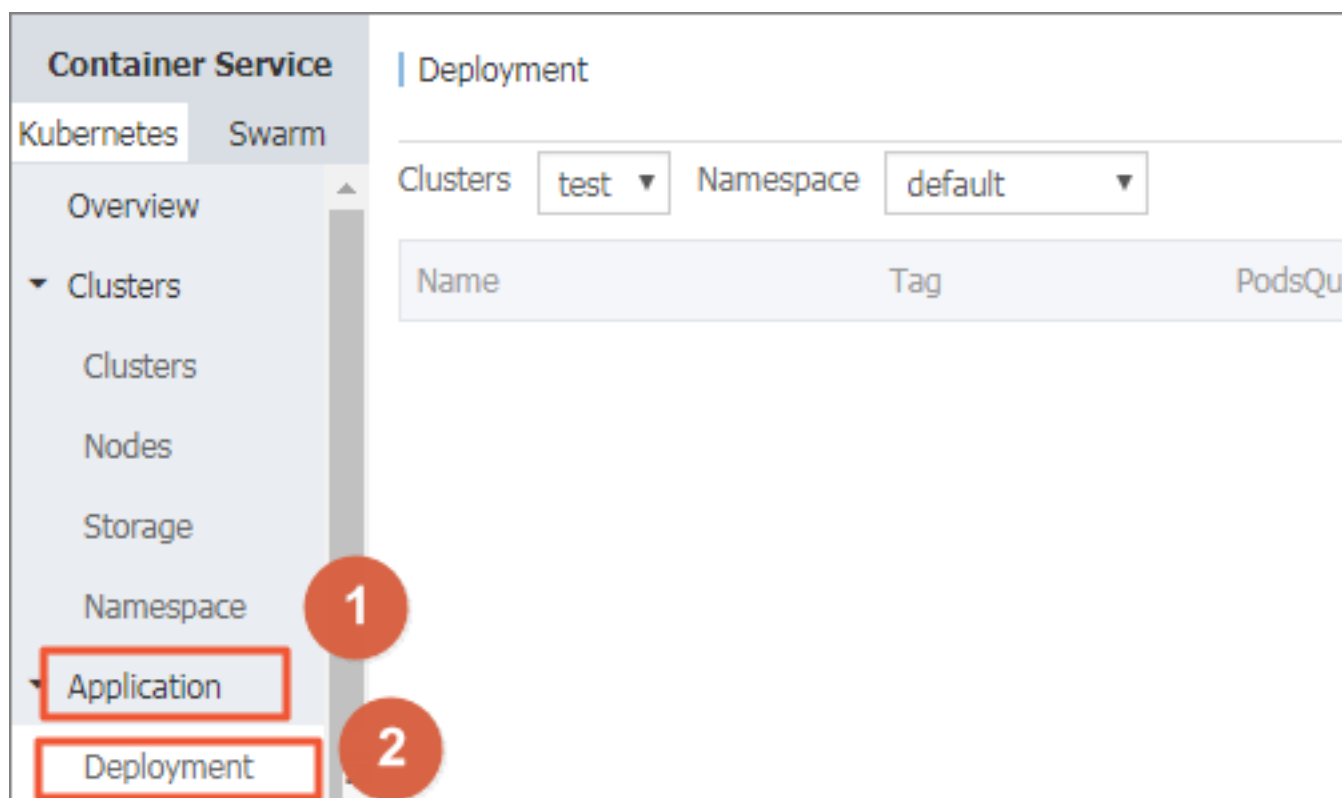
By default, a running pod can use the CPU and memory of nodes unlimitedly, which means any pod can use the computing resources of the cluster unlimitedly, and the pods of a namespace may use up the cluster resources.

One of the important functions of namespaces is to act as a virtual cluster for multiple purposes and meeting the requirements of multiple users. Therefore, configuring the resource quotas for a namespace is a kind of best practice.

You can configure the resource quotas for a namespace, including CPU, memory, and number of pods. For more information, see [Resource Quotas](#).

Procedure

1. Log on to the [Container Service console](#).
2. Under Kubernetes, click **Application > Deployment** in the left-side navigation pane. Click **Create by template** in the upper-right corner.



3. On the Deploy templates page, select the cluster and namespace (**test** in this example) from the Clusters and Namespace drop-down lists. Use a custom template or the example template Resource – ResourceQuota.

Deploy templates

Only Kubernetes versions 1.8.4 and above are supported. For clusters of version 1.8.1, you

Clusters

test

Namespace

test

Resource Type

Resource - ResourceQuota

Template

```

1  apiVersion: v1
2  kind: ResourceQuota # restrict resource
   roller, pods, service, secret, configma
3  metadata:
4    name: quota
5    # namespace: users-namespace # specif
6  spec:
7    hard:
8      cpu: "2" # adjust limits of cpu fo
9      memory: 4Gi # adjust memory upper
10     requests.storage: 1024G # adjust r
11     persistentvolumeclaims: "50" # adj
12     pods: "50" #adjust number of Pod i
13     replicationcontrollers: "10" # adj
14     services: "10" # adjust number of
15     secrets: "100" # adjust number of
16     configmaps: "100" # adjust number

```

You must configure the ResourceQuota template according to your cluster resources and the plan for the namespace resources. In this example, the template is as follows:

```

apiVersion: v1
kind: ResourceQuota # restrict resource quota for cpu, memory
, storage, pvc, replicationcontroller, pods, service, secret,
configmap

```



```

metadata:
  name: quota
# namespace: users-namespace # specify your namespace to apply
resource quota
spec:
  hard:
    cpu: "2" # adjust limits of cpu for your namespace
    memory: 4Gi # adjust memory upper limits for your namespace
    requests.storage: 1024G # adjust request of storage size for
your namespace
    persistentvolumeclaims: "50" # adjust number of pvc for your
namespace
    pods: "50" #adjust number of Pod in your namespace
    replicationcontrollers: "10" # adjust number of Replicatio
nController in your namespace
    services: "10" # adjust number of service for your namespace
    secrets: "100" # adjust number of secrets for your namespace
    configmaps: "100" # adjust number of configmap for your
namespace

```

4. You have configured the resource quotas for this namespace. Connect to the master node SSH IP address and run the following command to view the resource quotas and usage of this namespace.

```

# kubectl describe quota quota --namespace=test
Name: quota
Namespace: test
Resource Used Hard
-----
configmaps 0 100
cpu 0 2
memory 0 4Gi
persistentvolumeclaims 0 50
pods 0 50
replicationcontrollers 0 10
requests.storage 0 1024G
secrets 1 100
services 0 10

```

1.5.3 Update a namespace

Prerequisites

- You have created a Kubernetes cluster. For more information, see [Create a Kubernetes cluster](#).
- You have created a namespace **test**. For more information, see [Create a namespace](#).

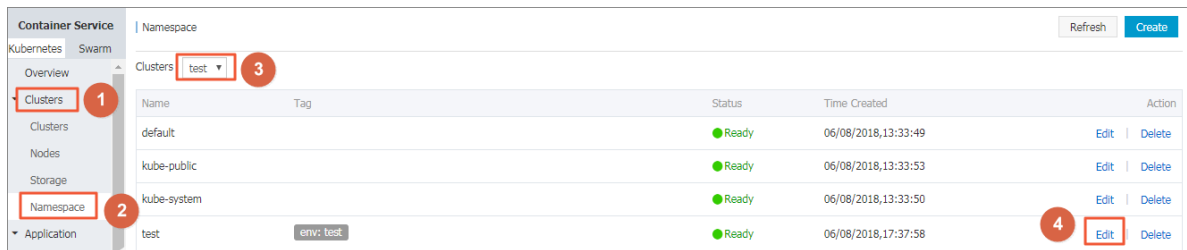
Context

You can update a namespace to add, modify, or delete the namespace tags.

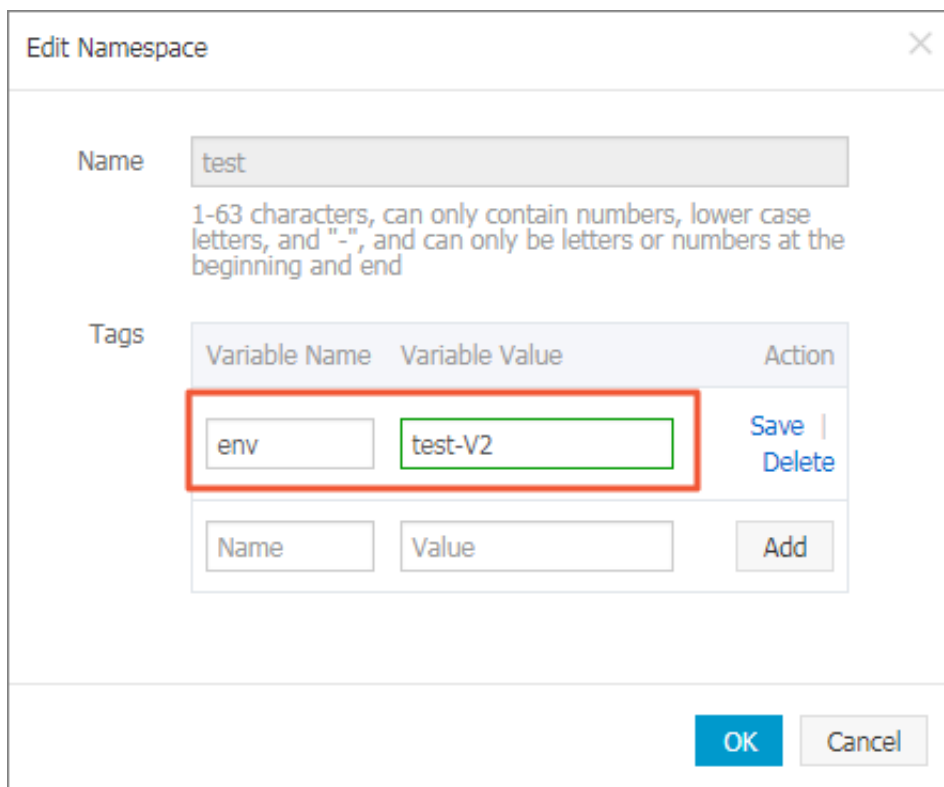
Procedure

1. Log on to the [Container Service console](#).

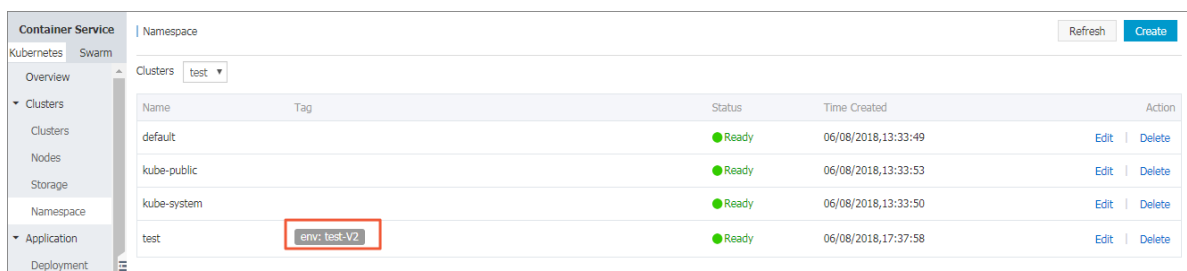
2. Under Kubernetes, click **Clusters** > **Namespace** in the left-side navigation pane.
3. Select the cluster from the Clusters drop-down list and click **Edit** at the right of the cluster.



4. Update the namespace tags in the displayed dialog box. For example, change the tag to `env: test-V2`.



5. Click Save on the right and then click **OK**. The updated namespace tag is displayed in the namespace list.



1.5.4 Delete a namespace

You can delete the namespaces that are no longer in use.

Prerequisites

- You have created a Kubernetes cluster. For more information, see [Create a Kubernetes cluster](#).
- You have created a namespace **test**. For more information, see [Create a namespace](#).

Context

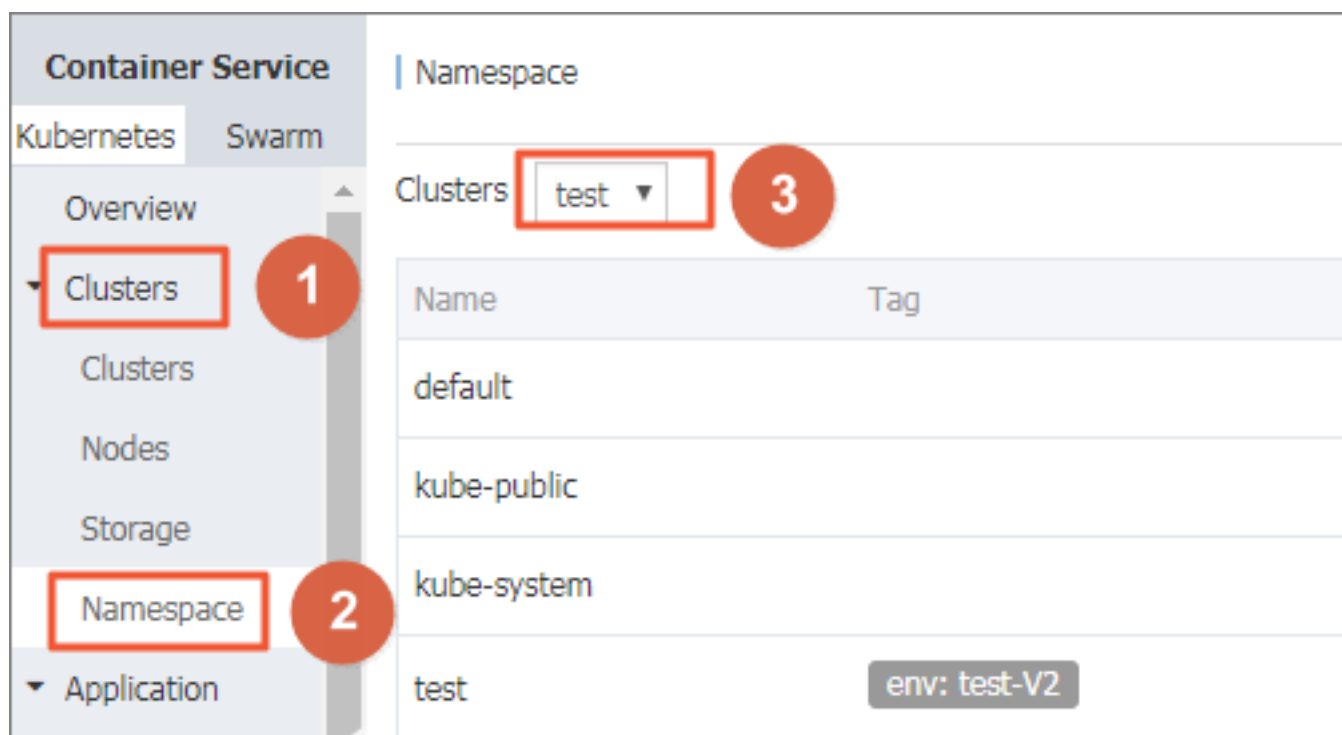


Note:

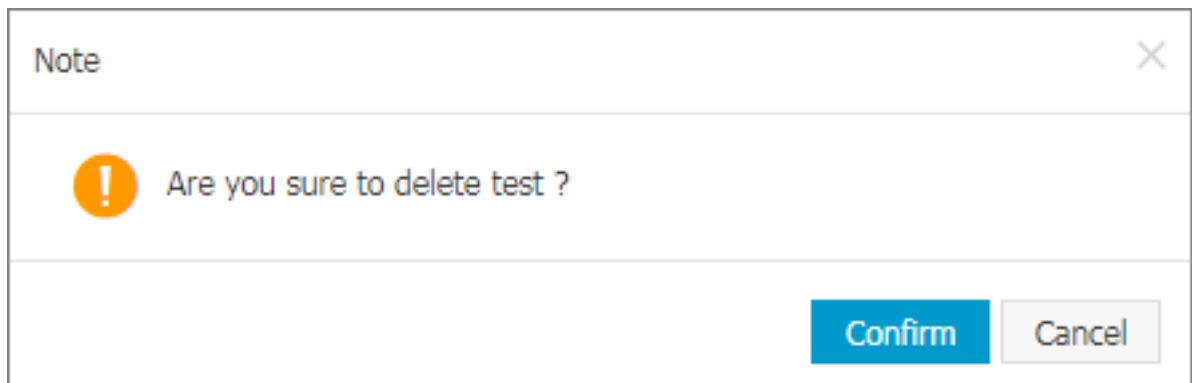
Deleting a namespace also deletes all of its resource objects, so proceed with caution.

Procedure

- Log on to the [Container Service console](#).
- Under Kubernetes, click **Clusters** > **Namespace** in the left-side navigation pane.
- Select the cluster from the Clusters drop-down list and click **Delete** at the right of the cluster.



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



5. The namespace is deleted from the namespace list and its resource objects are also deleted.

1.6 Applications

1.6.1 Create an application by using an image

Prerequisites

Create a Kubernetes cluster. For more information, see [Create a Kubernetes cluster](#).

Procedure

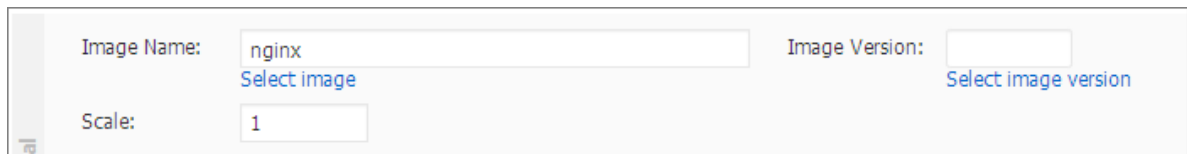
1. Log on to the [Container Service console](#).
2. Under Kubernetes, click **Application** > **Deployment** in the left-side navigation pane. Enter the Deployment List page and click **Create by image** in the upper-right corner.
3. Enter the application **Name**, then select the **Cluster** and **Namespace**. Click **Next** to go to the Configuration step.

By default, the system uses the default namespace if the **namespace** is not configured.

4. Configure the general settings for the application.
 - **Image name:** You can click **Select image** to select the image in the displayed dialog box and then click **OK**. In this example, the image name is nginx.

You can also enter the private registry in the format of `domainname/namespace/imagename:tag`.

- **Image version:** Click **Select image version** to select the version. If the image version is not specified, the system uses the latest version by default.
- **Scale:** Specify the number of containers. In this example, only one container is in the pod. If multiple containers are specified, the same number of pods will be started.



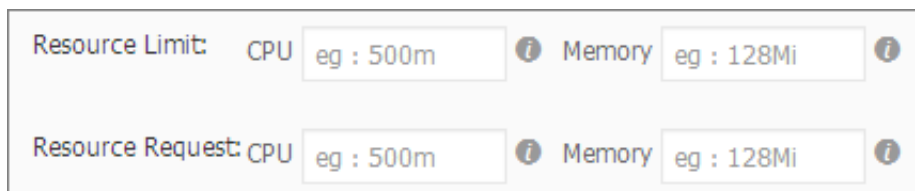
The screenshot shows a configuration panel with the following fields:

- Image Name:** A text input field containing "nginx" and a blue link "Select image" below it.
- Image Version:** A text input field and a blue link "Select image version" below it.
- Scale:** A text input field containing the number "1".

5. Configure the resource limit and resource reserve for the container.

- **Resource Limit:** Specify the upper limit for the resources (CPU and memory) that can be used by this application to avoid occupying excessive resources.
- **Resource Request:** Specify how many resources (CPU and memory) are reserved for the application, that is, these resources are exclusive to the container. Other services or processes will compete for resources when the resources are insufficient. By specifying the Resource Request, the application will not become unavailable because of insufficient resources.

CPU is measured in millicores (one thousandth of one core). Memory is measured in bytes, which can be Gi, Mi, or Ki.



The screenshot shows two rows of configuration fields:

- Resource Limit:** CPU (eg : 500m) and Memory (eg : 128Mi), each with an information icon (i).
- Resource Request:** CPU (eg : 500m) and Memory (eg : 128Mi), each with an information icon (i).

6. Configure the data volumes.

Local storage and cloud storage can be configured.

- **Local storage:** Supports hostPath, configmap, secret, and temporary directory. The local data volumes mount the corresponding mount source to the container path. For more information, see [volumes](#).
- **Cloud storage:** Supports three types of cloud storage: cloud disk, Network Attached Storage (NAS), and Object Storage Service (OSS).

In this example, a data volume of cloud disk is configured. Mounting the cloud disk to the `/tmp` container path stores data generated in this path to the cloud disk.

Storage type	Mount source	Container Path
Disk	pv-yunpan-test	/tmp

7. Configure the environment variable.

You can configure the environment variable for the pod in the format of key-value pairs to add the environment label or pass the configurations for the pod. For more information, see [Pod variable](#).

8. Configure the container.

You can configure the Command, Args, and Container Config for the container running in the pod.

- **Command and Args:** If not configured, the default settings of the image are used. If configured, the default settings of the image are overwritten. If only the Args is configured, the default command will run the new arguments when the container is started. Command and Args cannot be modified after the pod is created.
- **Container Config:** Select the stdin check box to enable standard input for the container. Select the tty check box to assign an virtual terminal to send signals to the container. These two options are usually used together, which indicates to bind the terminal (tty) to the container standard input (stdin). For example, an interactive program obtains standard input from you and then displays the obtained standard input in the terminal.

9. Configure health check

The health check function includes liveness probes and readiness probes. Liveness probes are used to detect when to restart the container. Readiness probes determine if the container is ready for receiving traffic. For more information about health checks, see <https://kubernetes.io/docs/tasks/configure-pod-container/configure-liveness-readiness-probes>.

Request method	Configuration description
HTTP request	<p>An HTTP GET request is sent to the container. The following are supported parameters:</p> <ul style="list-style-type: none">• Protocol: HTTP/HTTPS• Path: Path to access the HTTP server• Port: Number or name of the port exposed by the container. The port number must be in the range of 1 to 65535.• HTTP Header: Custom headers in the HTTP request. HTTP allows repeated headers. Supports the correct configuration of key values.• Initial Delay (in seconds): Namely, the initialDelaySeconds. Seconds for the first liveness or readiness probe has to wait after the container is started.• Period (in seconds): Namely, the periodseconds. Intervals at which the probe is performed. The default value is 10 seconds. The minimum value 1 second.• Timeout (in seconds): Namely, the timeoutSeconds. The time of probe timeout. The default value is 1 second and the minimum value is 1 second.• Success Threshold: The minimum number of consecutive successful probes that are considered as successful after a failed probe. The default value is 1. This parameter must be 1 for liveness. The minimum value is 1.• Failure Threshold: The minimum number of consecutive failed probes that are considered as failed after a successful probe. The default value is 3. The minimum value is 1.
TCP connection	<p>A TCP socket is send to the container. The kubelet attempts to open a socket to your container on the specified port. If a connection can be established, the container is considered healthy. If not, it is considered</p>

Request method	Configuration description
	<p>as a failure. The following are supported parameters:</p> <ul style="list-style-type: none"> • Port: Number or name of the port exposed by the container. The port number must be in the range of 1 to 65535. • Initial Delay (in seconds): Namely, the initialDelaySeconds. Seconds for the first liveness or readiness probe has to wait after the container is started. • Period (in seconds): Namely, the periodseconds. Intervals at which the probe is performed. The default value is 10 seconds. The minimum value 1 second. • Timeout (in seconds): Namely, the timeoutSeconds. The time of probe timeout. The default value is 1 second and the minimum value is 1 second. • Success Threshold: The minimum number of consecutive successful probes that are considered as successful after a failed probe. The default value is 1. This parameter must be 1 for liveness. The minimum value is 1. • Failure Threshold: The minimum number of consecutive failed probes that are considered as failed after a successful probe. The default value is 3. The minimum value is 1.
Command line	<p>Detect the health of the container by executing probe detection commands in the container. The following are supported parameters:</p> <ul style="list-style-type: none"> • Command: A probe command used to detect the health of the container. • Initial Delay (in seconds): Namely, the initialDelaySeconds. Seconds for the first liveness or readiness probe has to wait after the container is started. • Period (in seconds): Namely, the periodseconds. Intervals at which the

Request method	Configuration description
	<p>probe is performed. The default value is 10 seconds. The minimum value 1 second.</p> <ul style="list-style-type: none"> • Timeout (in seconds): Namely, the timeoutSeconds. The time of probe timeout. The default value is 1 second and the minimum value is 1 second. • Success Threshold: The minimum number of consecutive successful probes that are considered as successful after a failed probe. The default value is 1. This parameter must be 1 for liveness. The minimum value is 1. • Failure Threshold: The minimum number of consecutive failed probes that are considered as failed after a successful probe. The default value is 3. The minimum value is 1.

10. Select whether or not to enable the **Auto Scaling**.

You can choose whether to enable **Auto Scaling**. To meet the demands of applications under different loads, Container Service supports the container auto scaling, which automatically adjusts the number of containers according to the container CPU and memory usage.

Deploy

Auto Scaling: ☒ Enable
Metric: CPU Usage
Condition: Usage %
Max Number of Containers: Range : 2-100
Min Number of Containers: Range : 1-100



Note:

To enable auto scaling, you must configure required resources for the deployment. Otherwise, the container auto scaling cannot take effect.

- **Metric:** CPU and memory. Configure a resource type as needed.
- **Condition:** The percentage value of resource usage. The container begins to expand when the resource usage exceeds this value.

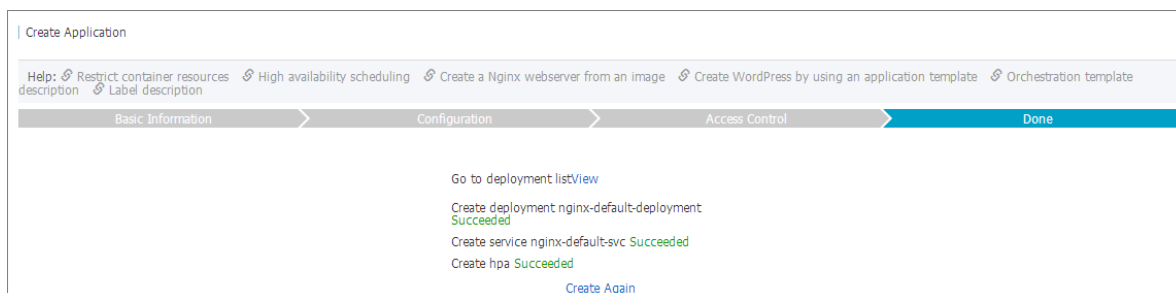
- **Maximum number of containers:** The maximum number of containers that the deployment can expand to.
- **Minimum number of containers:** The minimum number of containers that the deployment can shrink to.

11. Click **Next** after completing the configurations.

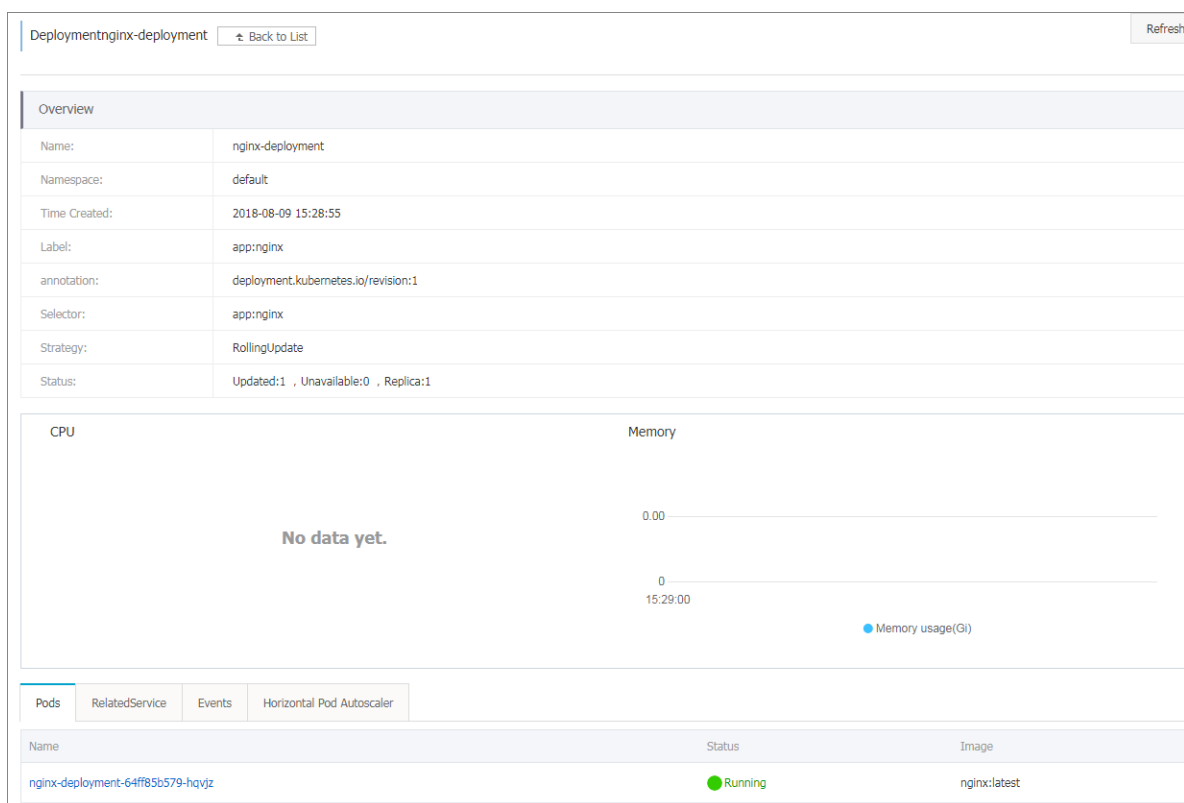
12. In the Access Control step, configure a service to bind with the backend pods. Click **Create** after the access control configurations.

- **Service:** Select None to not create a service, or select a service type as follows:
 - **ClusterIP:** Exposes the service by using the internal IP address of your cluster. With this type selected, the service is accessible only within the cluster.
 - **NodePort:** Exposes the service by using the IP address and static port (NodePort) on each node. A ClusterIP service, to which the NodePort service is routed, is automatically created. You can access the NodePort service from outside the cluster by requesting `<NodeIP>:<NodePort>`.
 - **Server Load Balancer:** Exposes the service by using Server Load Balancer, which is provided by Alibaba Cloud. Select public or inner to access the service by using the Internet or intranet. Server Load Balancer can route to the NodePort and ClusterIP services.
- **Name:** By default, a service name composed of the application name and the suffix svc is generated. In this example, the generated service name is nginx-default-svc. You can modify the service name as needed.
- **Port Mapping:** You must add the service port and the container port. If NodePort is selected as the service type, you must configure the node port to avoid the port conflict. Select TCP or UDP as the Protocol.

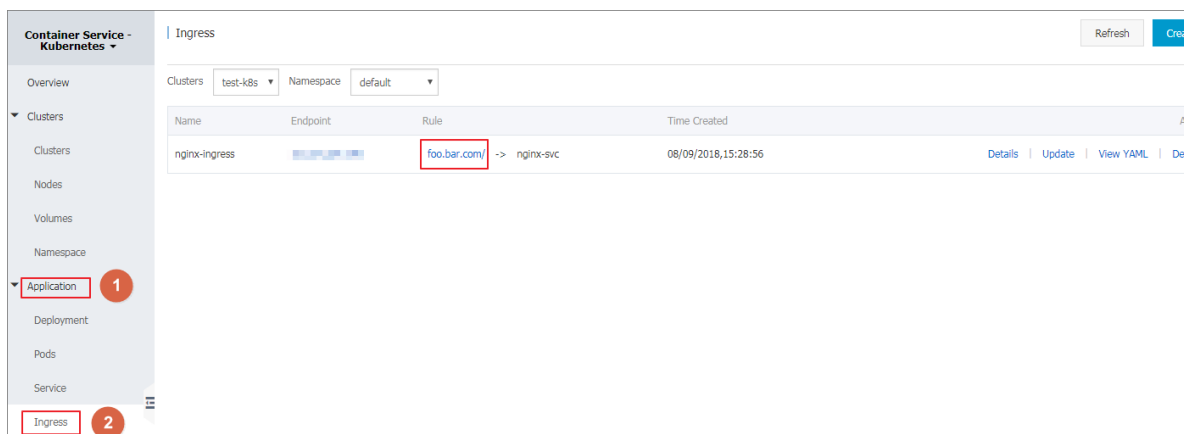
13. The Done step indicating the successful creation appears. The objects contained in the application are displayed. You can click **View** to view the deployment list.



14. The newly created deployment nginx-default-deployment is displayed on the Deployment page.



15. Click **Application > Service** in the left-side navigation pane. The newly created service nginx-default-svc is displayed on the Service List page.



16. Access the external endpoint in the browser to access the Nginx welcome page.

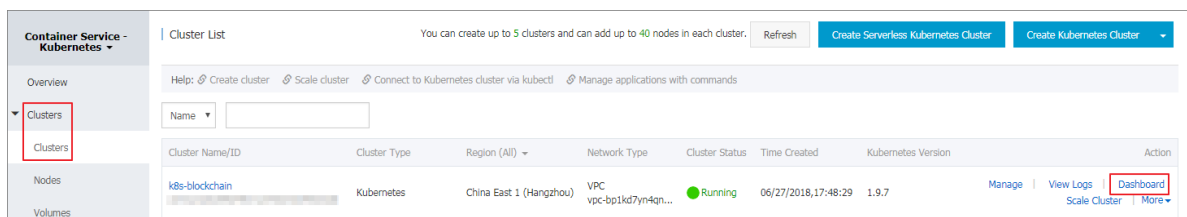


1.6.2 Create an application in Kubernetes dashboard

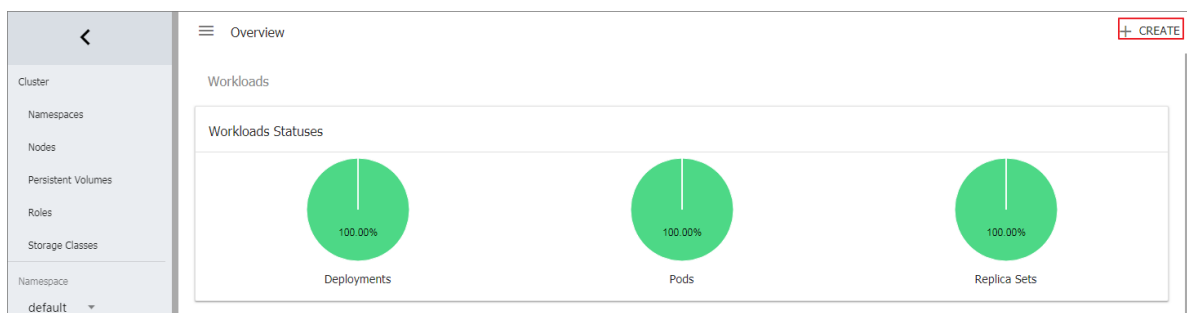
You can create an application in the Kubernetes dashboard.

Procedure

1. Log on to the [Container Service console](#).
2. Under Kubernetes, click **Clusters** in the left-side navigation pane.
3. Click **Dashboard** at the right of the cluster to enter the Kubernetes dashboard.



4. In the Kubernetes dashboard, click **CREATE** in the upper-right corner to create an application.



5. The Resource creation page appears. Configure the application information.

Create an application in any of the following three ways:

- **CREATE FROM TEXT INPUT:** Directly enter the orchestration codes in the YAML or JSON format to create an application. You must know the corresponding orchestration format.

Resource creation
+ CREATE

CREATE FROM TEXT INPUT
CREATE FROM FILE
CREATE AN APP

Enter YAML or JSON content specifying the resources to deploy to the currently selected namespace. [Learn more](#)

```

1 apiVersion: apps/v1beta2 # for versions before 1.8.0 use apps/v1beta1
2 kind: Deployment
3 metadata:
4   name: nginx-deployment-basic
5   labels:
6     app: nginx
7 spec:
8   replicas: 2
9   selector:
10    matchLabels:
11      app: nginx
12 template:
13   metadata:
14     labels:
15       app: nginx
16   spec:
17     # nodeSelector:
18     #   env: test-team
19     containers:
20     - name: nginx
21       image: nginx:1.7.9 # replace it with your exactly <image_name>:tags>

```

UPLOAD
CANCEL

- **CREATE AN APP:** Complete the following configurations to create an application.
 - **App name:** Enter the name of the application you are about to create. In this example, enter `nginx-test`.
 - **Container image:** Enter the URL of the image to be used. In this example, use Docker [Nginx](#).
 - **Number of pods:** Configure the number of pods for this application.
 - **Service:** Select **External** or **Internal**. **External** indicates to create a service that can be accessed from outside the cluster. **Internal** indicates to create a service that can be accessed from within the cluster.
 - **Advanced options:** To configure the information such as labels and environment variables, click **SHOW ADVANCED OPTIONS**. This configuration distributes the traffic load evenly to three pods.

CREATE FROM TEXT INPUT
CREATE FROM FILE
CREATE AN APP

App name *
nginx-test

Container image *
nginx

Number of pods *
3

Service *
None

SHOW ADVANCED OPTIONS

DEPLOY
CANCEL

An 'app' label with this value will be added to the Deployment and Service that get deployed. [Learn more](#)

Enter the URL of a public image on any registry, or a private image hosted on Docker Hub or Google Container Registry. [Learn more](#)

A Deployment will be created to maintain the desired number of pods across your cluster. [Learn more](#)

Optionally, an internal or external Service can be defined to map an incoming Port to a target Port seen by the container. The internal DNS name for this Service will be: nginx-test. [Learn more](#)



- **CREATE FROM FILE:** Upload an existing YAML or JSON configuration file to create an application.

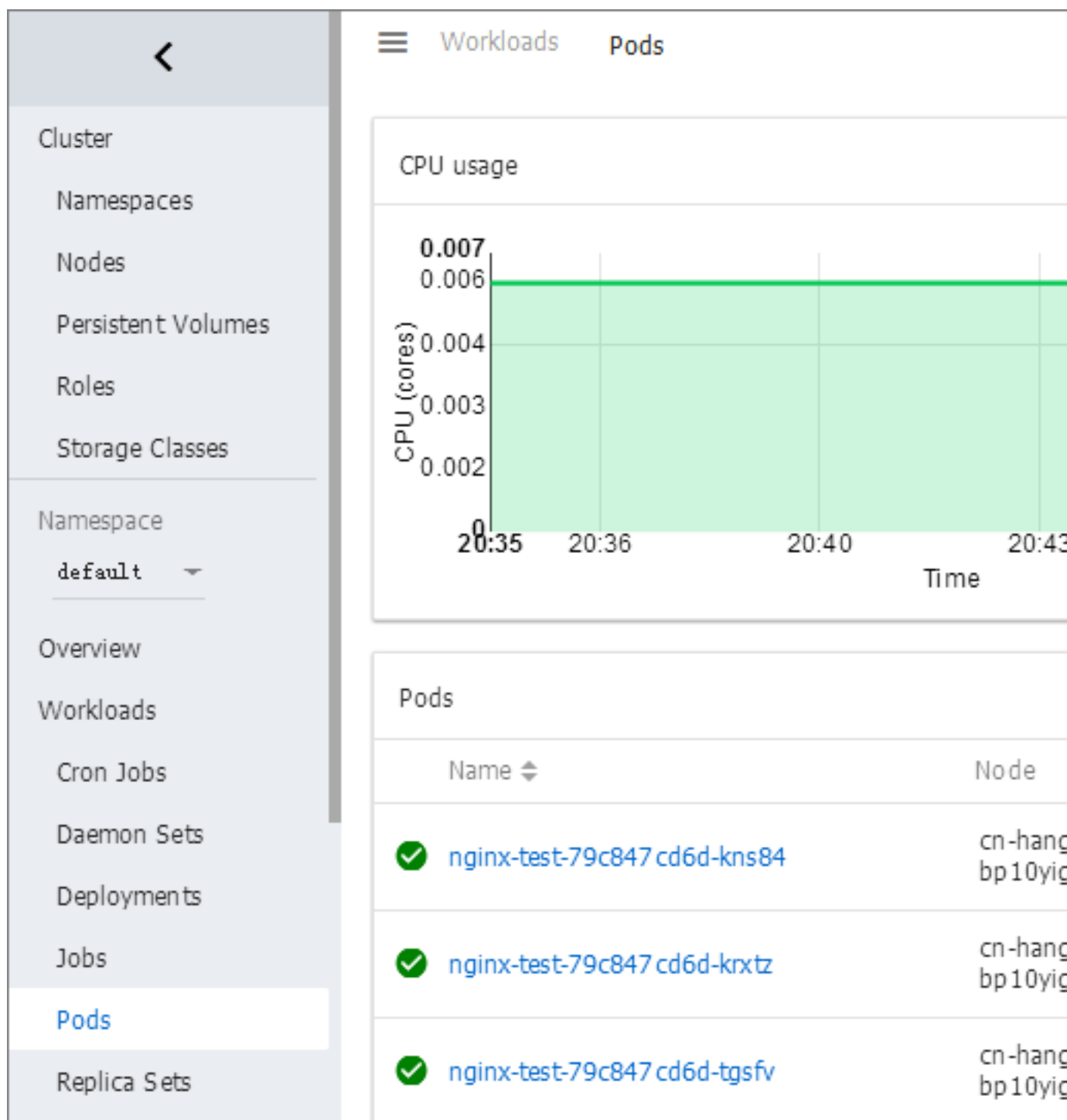
6. Click **UPLOAD** or **DEPLOY** to deploy the containers and services.

You can also click **SHOW ADVANCED OPTIONS** to configure more parameters.

What's next

After clicking **UPLOAD** or **DEPLOY**, you can view the services and containers of the application.

Click **Pods** in the left-side navigation pane. You can check the status of each Kubernetes object according to the icon on the left.  indicates the object is still being deployed.  indicates the object has completed the deployment.



1.6.3 Create an application by using an orchestration template

Prerequisites

Create a Kubernetes cluster. For more information, see [Create a Kubernetes cluster](#).

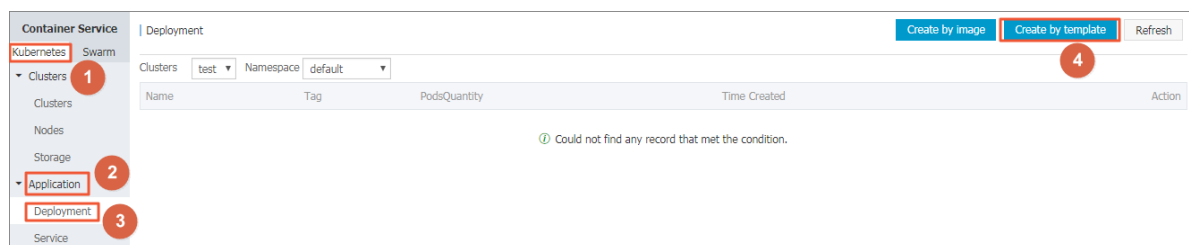
Context

In the Container Service Kubernetes orchestration template, you must define a resource object required for running an application, and combine the resource objects into a complete application by using label selector.

Create an Nginx application in this example. Firstly, create a backend pod resource object by creating the deployment. Then, deploy the service to bind it to the backend pod, forming a complete Nginx application.

Procedure

1. Log on to the [Container Service console](#).
2. Click **Kubernetes > Application > Deployment** in the left-side navigation pane.
3. Click **Create by template** in the upper-right corner.



4. Configure the template and then click **DEPLOY**.
 - **Clusters:** Select the cluster in which the resource object is to be deployed.
 - **Namespace:** Select the namespace to which the resource object belongs. default is selected by default. Except for the underlying computing resources such as nodes and persistent storage volumes, most of the resource objects must act on a namespace.
 - **Resource Type:** Alibaba Cloud Container Service provides Kubernetes YAML sample templates of many resource types for you to deploy resource objects quickly. You can write your own template based on the format requirements of Kubernetes YAML orchestration to describe the resource type you want to define.

Deploy templates

Only Kubernetes versions 1.8.4 and above are supported. For clusters of version 1.8.1, you can perform "upgrade cluster" operation in the cluster list

Clusters
test

Namespace
default

Resource Type
Resource - basic Deployment

1

Template

```

1 apiVersion: apps/v1beta2 # for versions before 1.8.0 use apps/v1beta1
2 kind: Deployment
3 metadata:
4   name: nginx-deployment-basic
5   labels:
6     app: nginx
7 spec:
8   replicas: 2
9   selector:
10    matchLabels:
11      app: nginx
12   template:
13     metadata:
14       labels:
15         app: nginx
16     spec:
17       # nodeSelector:
18       #   env: test-team
19     containers:
20     - name: nginx
21       image: nginx:1.7.9 # replace it with your exactly <image_name:tags>
22       ports:
23         - containerPort: 80

```

2

DEPLOY

The deployment sample orchestration of an Nginx application is as follows. By using this orchestration template, you can create a deployment that belongs to an Nginx application quickly.

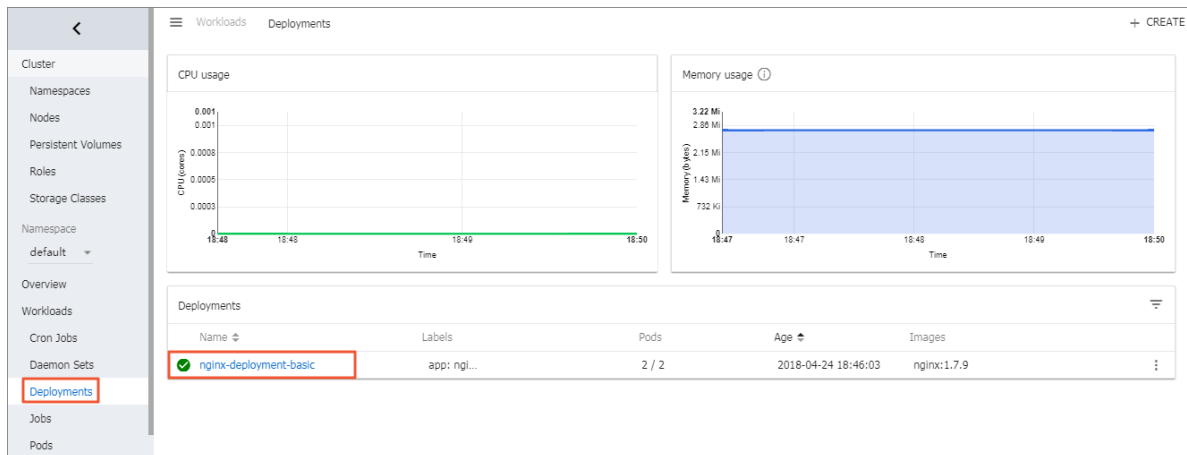
```

apiVersion: apps/v1beta2 # for versions before 1.8.0 use apps/
v1beta1
kind: Deployment
metadata:
  name: nginx-deployment
  labels:
    app: nginx
spec:
  replicas: 2
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx:1.7.9 # replace it with your exactly <
image_name:tags>
        ports:

```

```
- containerPort: 80
```

- After you click **DEPLOY**, a message indicating the deployment status is displayed. After the successful deployment, click **Kubernetes Dashboard** in the message to go to the dashboard and check the deployment progress.



- Go back to the **Deploy templates** page and deploy a service resource object.

Container Service provides a service sample template of an Nginx application. In this example, modify the template a little by changing the access type to LoadBalancer, and then you can create a service bound to the backend pod, which allows you to access the service in the browser.



Note:

In this example, the selector values in the pod orchestration and service orchestration are both nginx, so no modification is required. Make the corresponding modifications according to your actual situations.

```
apiVersion: v1 # for versions before 1.8.0 use apps/v1beta1
kind: Service
metadata:
  name: my-service1 #TODO: to specify your service name
  labels:
    app: nginx
spec:
  selector:
    app: nginx #TODO: change label selector to match your backend
  ports:
    - protocol: TCP
      name: http
      port: 30080 #TODO: choose an unique port on each node to avoid
      targetPort: 80
```

```
type: LoadBalancer ##In this example, change the type from
NodePort to LoadBalancer.
```

7. Enter the preceding orchestration contents in the Template field and then click **DEPLOY**. A message indicating the deployment status is displayed after you click DEPLOY. After the successful deployment, click **Kubernetes Dashboard** in the message to go to the dashboard and check the deployment progress of the service.

Clusters: test

Namespace: default

Resource Type: Resource - Service

Template:

```
1 apiVersion: v1
2 kind: Service
3 metadata:
4   name: my-service1 #TODO: to specify your service name
5   labels:
6     app: nginx-svc
7 spec:
8   selector:
9     app: nginx #TODO: change label selector to match your backend pod
10  ports:
11    - protocol: TCP
12      name: http
13      port: 30080 #TODO: choose an unique port on each node to avoid port conflict
14      targetPort: 80
15  type: LoadBalancer
```

Deployed successfully. Go to Dashboard to see the deployment progress: [Kubernetes Dashboard](#)

DEPLOY

8. In the Kubernetes dashboard, you can see the service my-service1 is successfully deployed and exposes the external endpoint. Click the access address under **External endpoints**.

Nodes

Persistent Volumes

Roles

Storage Classes

Namespace: default

Overview

Workloads

Cron Jobs

Daemon Sets

Deployments

Jobs

Pods

Replica Sets

Replication Controllers

Stateful Sets

Discovery and Load Balancing

Ingresses

Services

Discovery and load balancing Services

Name	Labels	Cluster IP	Internal endpoints	External endpoints	Age
my-service1	app: nginx-s...	172.17.0.70	my-service1:30080 TCP my-service1:31859 TCP	114.118.206.233:30080	2018-04-24 19:09:46
kubernetes	component: apiser... provider: kubernet...	172.17.0.1	kubernetes:443 TCP	-	2018-04-24 19:09:28

9. You can access the Nginx service welcome page in the browser.



1.6.4 Manage applications by using commands

You can create applications or view containers in applications by using commands.

Prerequisites

Before using commands to manage applications, [Connect to a Kubernetes cluster by using kubectl](#).

Create an application by using commands

Run the following statements to run a simple container (a Nginx Web server in this example).

```
root@master # kubectl run -it nginx --image=registry.aliyuncs.com/spacexnice/netdia:latest
```

This command creates a service portal for this container. Specify `--type=LoadBalancer` and an Alibaba Cloud Server Load Balancer route will be created to the Nginx container.

```
root@master # kubectl expose deployment nginx --port=80 --target-port=80 --type=LoadBalancer
```

View containers by using commands

Run the following command to list all the running containers in the default namespaces.

```
root@master # kubectl get pods
NAME READY STATUS RESTARTS AGE
nginx-2721357637-dvwq3 1/1 Running 1 9h
```

1.6.5 Simplify Kubernetes application deployment by using Helm

In Kubernetes, app management is the most challenging and in demand field. The Helm project provides a uniform software packaging method which supports version control and greatly simplifies Kubernetes app distribution and deployment complexity.

Alibaba Cloud Container Service integrates the app catalog management function with the Helm tool, extends the functions, and supports official repository, allowing you to deploy the application quickly. You can deploy the application in the Container Service console or by using command lines.

This document introduces the basic concepts and usage of Helm and demonstrates how to use Helm to deploy the sample applications WordPress and Spark on an Alibaba Cloud Kubernetes cluster.

Basic concepts of Helm

Helm is an open-source tool initiated by Deis and helps to simplify the deployment and management of Kubernetes applications.

You can understand Helm as a Kubernetes package management tool that facilitates discovery, sharing and use of apps built for Kubernetes. It involves several basic concepts.

- **Chart:** A Helm package containing the images, dependencies, and resource definitions required for running an application. It may also contain service definitions in a Kubernetes cluster, similar to the formula of Homebrew, the dpkg of APT, or the rpm file of Yum.
- **Release:** A chart running on a Kubernetes cluster. A chart can be installed multiple times on the same cluster. A new release will be created every time a chart is installed. For example, to run two databases on the server, you can install the MySQL chart twice. Each installation will generate its own release with its own release name.
- **Repository:** The repository for publishing and storing charts.

Helm components

Helm adopts a client/server architecture composed of the following components:

- Helm CLI is the Helm client and can be run locally or on the master nodes of the Kubernetes cluster.
- Tiller is the server component and runs on the Kubernetes cluster. It manages the lifecycles of Kubernetes applications.
- Repository is the chart repository. The Helm client accesses the chart index files and packages in the repository by means of the HTTP protocol.

Use Helm to deploy applications

Prerequisites

- Before using Helm to deploy an application, create a Kubernetes cluster in Alibaba Cloud Container Service. For more information, see [Create a Kubernetes cluster](#).

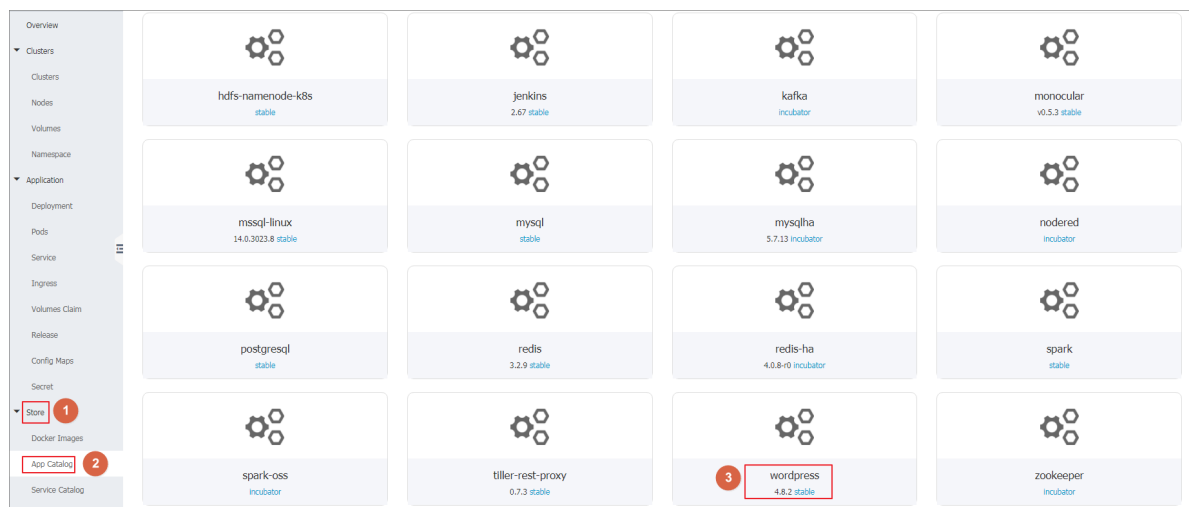
Tiller is automatically deployed to the cluster when the Kubernetes cluster is created. Helm CLI is automatically installed on all the master nodes and the configuration points to the Alibaba Cloud chart repository.

- Check the Kubernetes version of your cluster.

Only clusters whose Kubernetes version is 1.8.4 or later are supported. For clusters whose Kubernetes version is 1.8.1, **upgrade the cluster** on the Cluster List page.

Deploy applications in Container Service console

- Log on to the [Container Service console](#).
- Under Kubernetes, click **Store > App Catalog** in the left-side navigation pane.
- On the App Catalog page, click a chart (WordPress in this example) to enter the chart details page.



- Enter the basic information for the deployment on the right.
 - Clusters:** Select the cluster in which the application is to be deployed.
 - Namespace:** Select the namespace. default is selected by default.
 - Release Name:** Enter the release name for the application. Enter test in this example.

Readme
Values

WordPress

WordPress is one of the most versatile open source content management systems on the market. A publishing platform for building blogs and websites.

TL;DR;

```
$ helm install stable/wordpress
```

Introduction

This chart bootstraps a WordPress deployment on a Kubernetes cluster using the Helm package manager.

It also packages the Bitnami MariaDB chart which is required for bootstrapping a MariaDB deployment for the database requirements of the WordPress application.

Deploy

Only Kubernetes versions 1.8.4 and above are supported. For clusters of version 1.8.1, you can perform "upgrade cluster" operation in the cluster list

Clusters
k8s-cluster

Namespace
default

Release Name
wordpress-default

DEPLOY

5. Click the **Values** tab to modify the configurations.

In this example, bind dynamic data volumes of the cloud disk to a persistent storage volume claim (PVC). For more information, see .



Note:

You need to create a persistent storage volume (PV) of cloud disk in advance. The capacity of the PV cannot be less than the value defined by the PVC.

Readme
Values

```

72  ##
73  mariadbDatabase: bitnami_wordpress
74
75  ## Create a database user
76  ## ref: https://github.com/bitnami/bitnami-docker-mariadb/blob/master/README.md#creating-a
77  ## -database-user-on-first-run
78  mariadbUser: bn_wordpress
79
80  ## Password for mariadbUser
81  ## ref: https://github.com/bitnami/bitnami-docker-mariadb/blob/master/README.md#creating-a
82  ## -database-user-on-first-run
83  ## mariadbPassword:
84
85  ## Enable persistence using Persistent Volume Claims
86  ## ref: http://kubernetes.io/docs/user-guide/persistent-volumes/
87  ##
88  persistence:
89    enabled: true
90    ## mariadb data Persistent Volume Storage Class
91    ## If defined, storageClassName: <storageClass>
92    ## If set to "-", storageClassName: "", which disables dynamic provisioning
93    ## If undefined (the default) or set to null, no storageClassName spec is
94    ## set, choosing the default provisioner. (gp2 on AWS, standard on
95    ## GKE, AWS & OpenStack)
96    ##
97    storageClass: "alicloud-disk-efficiency"
98    accessMode: ReadWriteOnce
99    size: 20Gi
100
101  ## Kubernetes configuration
102  ## For minikube, set this to NodePort, elsewhere use LoadBalancer
103  ##
104  serviceType: LoadBalancer
105

```

Deploy

Only Kubernetes versions 1.8.4 and above are supported. For clusters of version 1.8.1, you can perform "upgrade cluster" operation in the cluster list

Clusters
k8s-cluster

Namespace
default

Release Name
wordpress-default

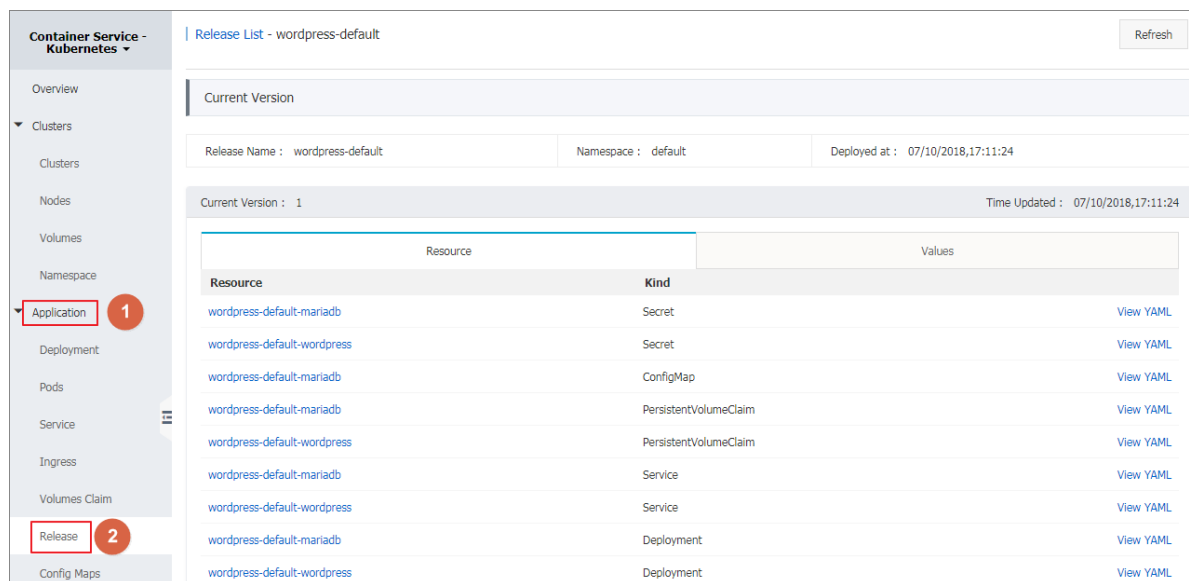
DEPLOY

Version

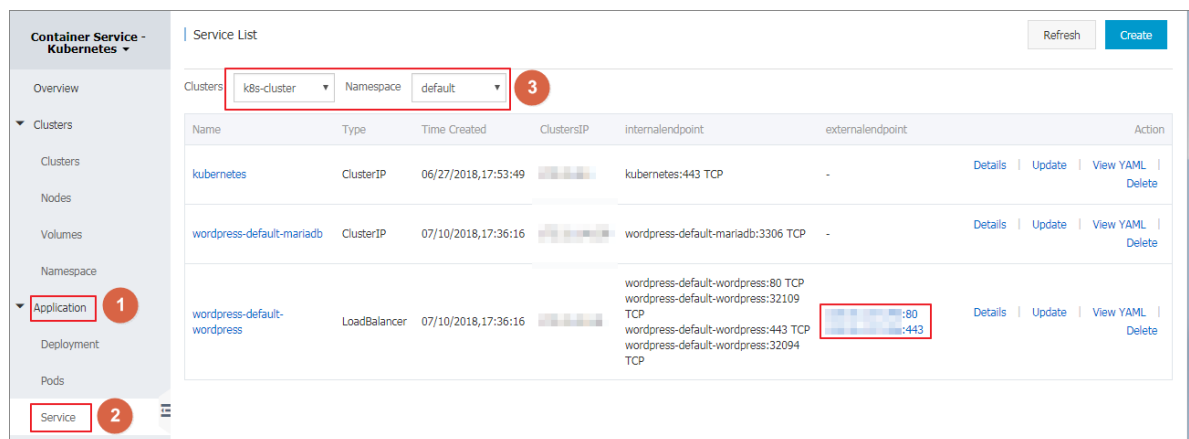
0.6.13

Project Homepage

6. Click **DEPLOY** after completing the configurations. After the successful deployment, you are redirected to the release page of this application.



7. Click **Application** > **Service** in the left-hand navigation pane. Select the target cluster and namespace and find the corresponding service. You can obtain the HTTP/HTTPS external endpoint address.



8. Click the preceding access address to enter the WordPress blog publishing page.

Deploy applications by using command lines

You can use SSH to log on to the master node of the Kubernetes cluster when deploying applications by using command lines (Helm CLI is automatically installed and has configured the repository). For more information, see [#unique_16](#). You can also install and configure the kubectl and Helm CLI locally.

In this example, install and configure the kubectl and Helm CLI locally and deploy the applications WordPress and Spark.

Install and configure kubectl and Helm CLI

1. Install and configure kubectl on a local computer.

For more information, see [Connect to a Kubernetes cluster by using kubectl](#).

To view information of the target Kubernetes cluster, enter the command `kubectl cluster-info`.

2. Install Helm on a local computer.

For the installation method, see [Install Helm](#).

3. Configure the Helm repository. Here the charts repository provided by Alibaba Cloud Container Service is used.

```
helm init --client-only --stable-repo-url https://aliacs-app-catalog
.oss-cn-hangzhou.aliyuncs.com/charts/
helm repo add incubator https://aliacs-app-catalog.oss-cn-hangzhou.
aliyuncs.com/charts-incubator/
helm repo update
```

Basic operations of Helm

- To view the list of charts installed on the cluster, enter the following command:

```
helm list
```

Or you can use the abbreviated version:

```
helm ls
```

- To view the repository configurations, enter the following command:

```
helm repo list
```

- To view or search for the Helm charts in the repository, enter one of the following commands:

```
helm search
helm search repository name #For example, stable or incubator.
helm search chart name #For example, wordpress or spark.
```

- To update the chart list to get the latest version, enter the following command:

```
helm repo update
```

For more information about how to use Helm, see [Helm document](#).

Deploy WordPress by using Helm

Use Helm to deploy a WordPress blog website.

Enter the following command.

```
helm install --name wordpress-test stable/wordpress
```

**Note:**

The Alibaba Cloud Kubernetes service provides the support for dynamic storage volumes of block storage (cloud disk). You need to create a storage volume of cloud disk in advance.

The result is as follows:

```
NAME: wordpress-test
LAST DEPLOYED: Mon Nov 20 19:01:55 2017
NAMESPACE: default
STATUS: DEPLOYED
...
```

Use the following command to view the release and service of WordPress.

```
helm list
kubectl get svc
```

Use the following command to view the WordPress related pods and wait until the status changes to Running.

```
kubectl get pod
```

Use the following command to obtain the WordPress access address:

```
echo http://$(kubectl get svc wordpress-test-wordpress -o jsonpath='{.status.loadBalancer.ingress[0].ip}')
```

Access the preceding URL in the browser, and you can see the familiar WordPress website.

You can also follow the chart instructions and use the following command to obtain the administrator account and password of the WordPress website:

```
echo Username: user
echo Password: $(kubectl get secret --namespace default wordpress-test-wordpress -o jsonpath="{.data.wordpress-password}" | base64 --decode)
```

To completely delete the WordPress application, enter the following command:

```
helm delete --purge wordpress-test
```

Deploy Spark by using Helm

Use Helm to deploy Spark for processing big data.

Enter the following command:

```
helm install --name myspark stable/spark
```

The result is as follows:

```
NAME: myspark
LAST DEPLOYED: Mon Nov 20 19:24:22 2017
NAMESPACE: default
STATUS: DEPLOYED
...
```

Use the following commands to view the release and service of Spark.

```
helm list
kubectl get svc
```

Use the following command to view the Spark related pods and wait until the status changes to Running. Pulling images takes some time because the Spark related images are large.

```
kubectl get pod
```

Use the following command to obtain the Spark Web UI access address:

```
echo http://$(kubectl get svc myspark-webui -o jsonpath='{.status.loadBalancer.ingress[0].ip}'):8080
```

Access the preceding URL in the browser, and you can see the Spark Web UI, on which indicating currently three worker instances exist.

Then, use the following command to use Helm to upgrade the Spark application and change the number of worker instances from three to four. The parameter name is case sensitive.

```
helm upgrade myspark --set "Worker.Replicas=4" stable/spark
```

The result is as follows:

```
Release "myspark" has been upgraded. Happy Helming!
LAST DEPLOYED: Mon Nov 20 19:27:29 2017
NAMESPACE: default
STATUS: DEPLOYED
...
```

Use the following command to view the newly added pods of Spark and wait until the status changes to Running.

```
kubectl get pod
```

Refresh the Spark Web UI in the browser. The number of worker instances changes to four.

To completely delete the Spark application, enter the following command:

```
helm delete --purge myspark
```

Use third-party chart repository

Besides the preset Alibaba Cloud chart repository, you can also use the third-party chart repository (make sure the network is accessible). Add the third-party chart repository in the following command format:

```
helm repo add repository name repository URL
helm repo update
```

For more information about the Helm related commands, see [Helm document](#).

References

Helm boosts the growth of communities. More and more software providers, such as Bitnami, have begun to provide high-quality charts. You can search for and discover existing charts at <https://kubernetes.io/>.

1.6.6 Create a service

A Kubernetes service, which is generally called a microservice, is an abstraction which defines a logical set of pods and a policy by which to access them. The set of pods accessed by a Kubernetes service is usually determined by a Label Selector.

Kubernetes pods are created and deleted in a short time even if they have their own IP addresses. Therefore, using pods directly to provide services externally is not a solution of high availability. The service abstraction decouples the relationship between the frontend and the backend. Therefore, the loose-coupling microservice allows the frontend to not care about the implementations of the backend.

For more information, see [Kubernetes service](#).

Prerequisite

You have created a Kubernetes cluster successfully. For how to create a Kubernetes cluster, see [#unique_58](#).

Step 1 Create a deployment

1. Log on to the [Container Service console](#).

2. Under Kubernetes, click **Application > > Deployment** in the left-side navigation pane. Click **Create by template** in the upper-right corner.
3. Select the cluster and namespace to create the deployment. In the Resource Type drop-down list, select Custom to customize the template or a sample template. Then, click **DEPLOY**.

In this example, the sample template is an Nginx deployment.

```
apiVersion: apps/v1beta2 # for versions before 1.8.0 use apps/
v1beta1
kind: Deployment
metadata:
  name: nginx-deployment-basic
  labels:
    app: nginx
spec:
  replicas: 2
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
        - name: nginx
          image: nginx:1.7.9 # replace it with your exactly <
image_name:tags>
          ports:
            - containerPort: 80 ##You must expose this port in the
service.
```

4. Go to the Kubernetes dashboard to view the running status of this deployment.

Step 2 Create a service

1. Log on to the [Container Service console](#).
2. Under Kubernetes, click **Application > > Service** in the left-side navigation pane.
3. Select the cluster and namespace from the Clusters and Namespace drop-down lists. Click **Create** in the upper-right corner.
4. Complete the configurations in the displayed Create Service dialog box.
 - **Name:** Enter the service name. In this example, enter nginx-svc.
 - **Type:** Select the service type, namely, the access method of the service.

- **ClusterIP:** Exposes the service by using the internal IP address of your cluster. With this type selected, the service is only accessible from within the cluster. This type is the default service type.
 - **NodePort:** Exposes the service by using the IP address and static port (NodePort) on each node. A ClusterIP service, to which the NodePort service is routed, is automatically created. You can access the NodePort service from outside the cluster by requesting `<NodeIP>:<NodePort>`.
 - **Server Load Balancer:** Exposes the service by using Server Load Balancer, which is provided by Alibaba Cloud. Select public or inner to access the service by using the Internet or intranet. Alibaba Cloud Server Load Balancer can route to the NodePort and ClusterIP services.
 - **Related deployment:** Select the backend object to bind with this service. In this example, select `nginx-deployment-basic`, the deployment created in the preceding step. The corresponding Endpoints object is not created if no deployment is selected here. You can manually map the service to your own endpoints. For more information, see [Services without selectors](#).
 - **Port Mapping:** Add the service port and container port. The container port must be the same as the one exposed in the backend pod.
5. Click **Create**. The `nginx-svc` service is displayed on the Service List page.
6. View the basic information of the service. Access the external endpoint of the `nginx-svc` service in the browser.

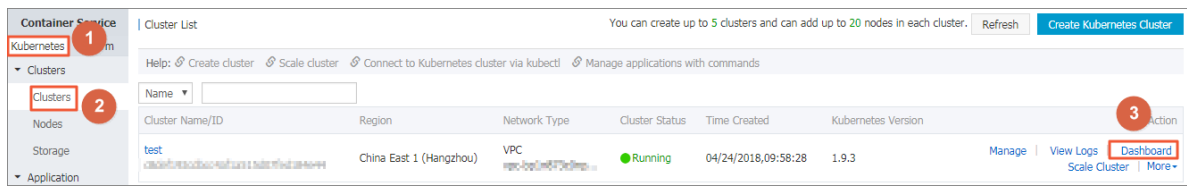
Then, you have created a service that is related to a backend deployment and accessed the Nginx welcome page successfully.

1.6.7 Service scaling

After an application is created, you can scale out or in the services as per your needs.

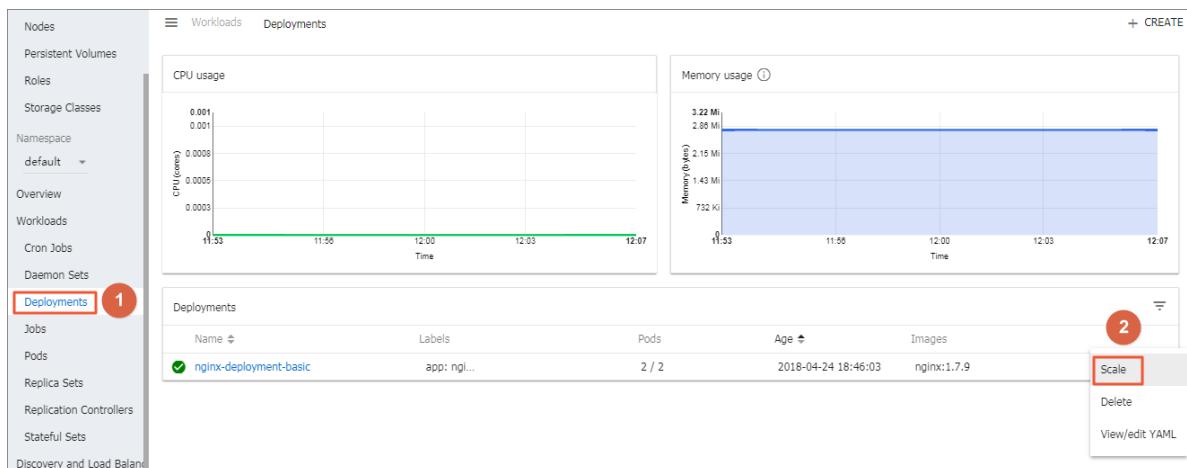
Procedure

1. Log on to the [Container Service console](#).
2. Click **Kubernetes > Clusters** in the left-side navigation pane.
3. Click **Dashboard** at the right of the cluster to enter the Kubernetes dashboard.



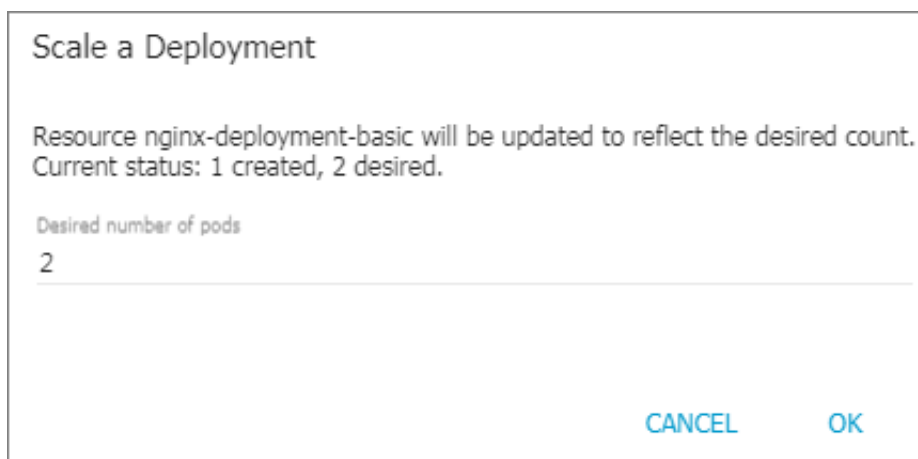
4. In the Kubernetes dashboard, click **Deployments** in the left-side navigation pane to view the created deployments.

5. Click the icon at the right of the deployment and then select **Scale**.



6. The Scale a Deployment dialog box appears. Modify the value of **Desired number of pods** to 2 and then click **OK**.

Then, a pod is added by expansion and the number of replicas rises to 2.



What's next

You can check the status of each Kubernetes object according to the icon on the left.



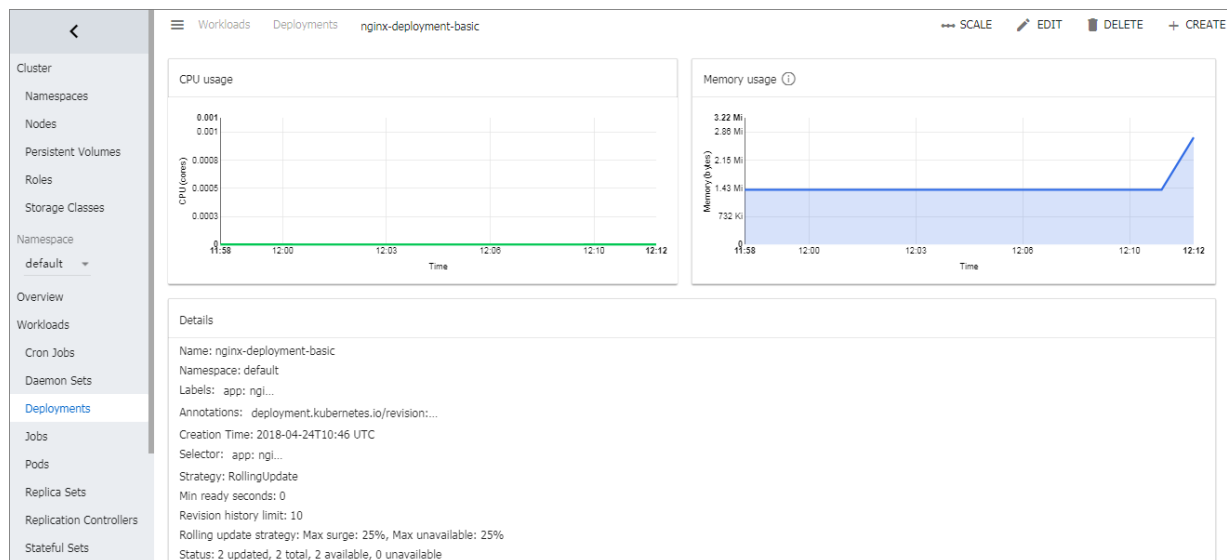
indicates the object is being deployed.  indicates the object has completed the deployment.

After the application completes the deployment, you can click a deployment name to view the details of the running Web service. You can view the replica sets in the deployment, and the CPU usage and memory usage of these replica sets. You can also click Pods in the left-side navigation pane, open a pod, and click **LOGS** in the upper-right corner to view the container logs.



Note:

Wait a few minutes if you cannot view any resources.



1.6.8 View services

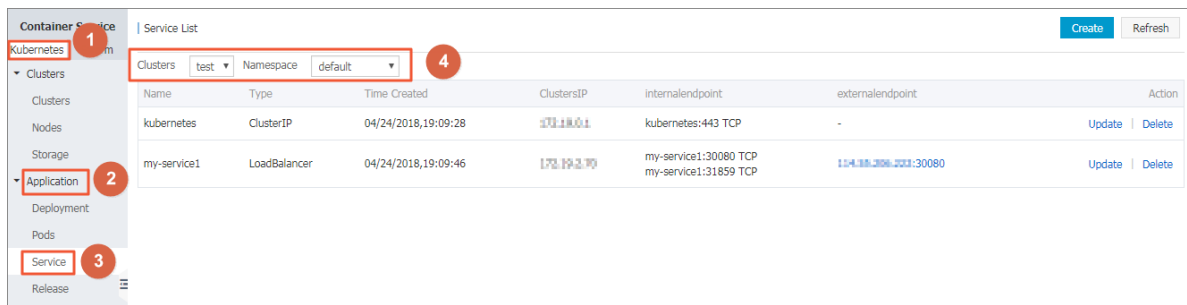
Context

If the external service is configured when you create the application, in addition to running containers, Kubernetes dashboard creates the external services for pre-assigning the Server Load Balancer to bring traffic to the containers in the cluster.

Procedure

1. Log on to the [Container Service console](#).
2. Click Kubernetes > **Application** > > **Service** in the left-side navigation pane.
3. Select the cluster and namespace from the Clusters and Namespace drop-down lists to view the deployed services.

You can view the name, type, created time, cluster IP address, internal endpoint, and external endpoint of a service. In this example, you can view the external endpoint (IP address) assigned to the service. Click the IP address to access the Nginx welcome page.



You can also enter the Kubernetes dashboard of the cluster and click **Services** in the left-side navigation pane to view the services.

1.6.9 Update a service

You can update a service in the Container Service console or Kubernetes dashboard.

Update a service in Container Service console

1. Log on to the [Container Service console](#).
2. Under Kubernetes, click **Application** > > **Service** in the left-side navigation pane.
3. Select the cluster and namespace from the Clusters and Namespace drop-down lists. Click **Update** at the right of the service (nginx-svc in this example).
4. The Update dialog box appears. Modify the template. In this example, change the nodePort to **31000**. Then, click **OK**.
5. On the Service List page, view the changes of the service. In this example, the nodePort is changed as follows.

Update a service in Kubernetes dashboard

1. Log on to the [Container Service console](#).
2. Under Kubernetes, click **Clusters** in the left-side navigation pane.
3. Click **Dashboard** at the right of the cluster to go to the Kubernetes dashboard.
4. In the Kubernetes dashboard, select the corresponding namespace and click **Services** in the left-side navigation pane.
5. Click the icon at the right of the service and then select **View/edit YAML** from the drop-down list.

- The Edit a Service dialog box appears. Modify the configurations. In this example, change the nodePort to **31000**. Then, click **UPDATE**.

1.6.10 Delete a service

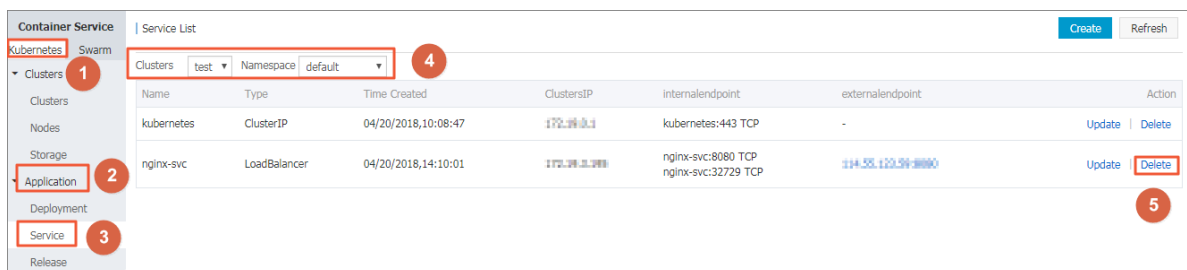
You can delete a Kubernetes service in the Container Service console.

Prerequisites

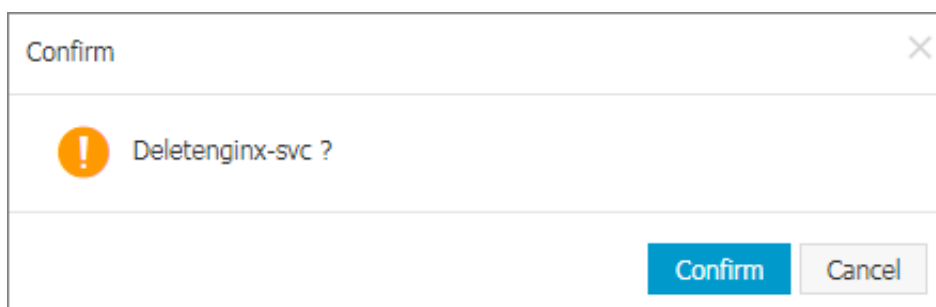
- You have created a Kubernetes cluster successfully. For more information, see [Create a Kubernetes cluster](#).
- You have created a service successfully. For more information, see [Create a service](#).

Procedure

- Log on to the [Container Service console](#).
- Click Kubernetes > **Application** > > **Service** in the left-side navigation pane.
- Select the cluster and namespace from the Clusters and Namespace drop-down lists. Click **Delete** at the right of the service (nginx-svc in this example).



- Click **Confirm** in the displayed dialog box. Then, the service is removed from the Service List page.



1.6.11 Use an application trigger

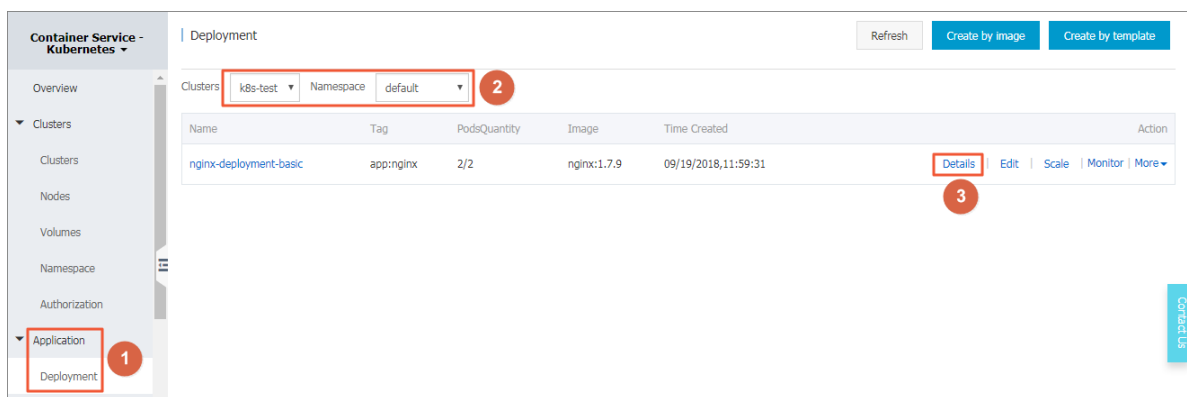
Alibaba Cloud Container Service Kubernetes supports the application trigger function. You can use an application trigger in many ways.

Prerequisites

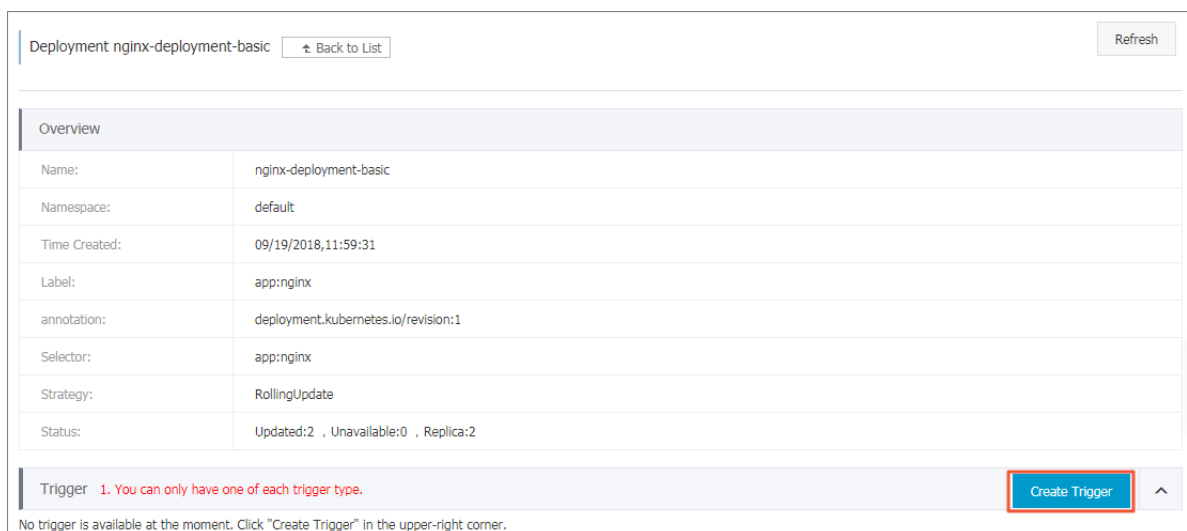
- You have created a Kubernetes cluster. For more information, see [Create a Kubernetes cluster](#).
- You have created an application that is used to create an application trigger and test the trigger. In this example, create an nginx application.

Procedure

- Log on to the [Container Service console](#).
- Click **Application > Deployment** and select a cluster and namespace. Click **Details** at the right of the target nginx application.



- On the nginx application details page, click **Create Trigger** on the right side of the trigger bar.



- In the pop-up dialog box, click **Redeploy** and click **Confirm**.

**Note:**

Currently, only the redeploy action is supported.

Create Trigger

* Action :

Redeploy

Confirm

Cancel

After the trigger is created, a trigger link is displayed in the trigger bar on the nginx application detail page.

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

Create Trigger

Trigger Link (move mouse over to copy)

https://cs.console.aliyun.com/hook/trigger?token=eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCJ9.eyJjbHVzdGVySWQ9IjNjN2NiZTYyODQ0NDMyMWFjYTNiZjkyYjQ3OGQx

Type	Action
Redeploy	Delete Trigger

- Copy the trigger link and visit it in the browser. A message is returned on the web page, containing information such as the request ID.

<https://cs.console.aliyun.com/hook/trigger>
 ("code":"200","message":"","requestId":"6e75bec1-69ce-4228-956b-56461da134db")

- Back to the nginx application detail page, you can see that a new pod appears.

Pods			
Access			
Events			
Horizontal Pod Autoscaler			
Name	Status	Image	
nginx-deployment-basic-6898cc69fb-9726v	Running	nginx:1.7.9	
nginx-deployment-basic-6898cc69fb-9nlns	Running	nginx:1.7.9	

After a period of time, the nginx application removes the old pod and keeps only the new pod.

What's next

You can call a trigger by using GET or POST in a third-party system. For example, you can run the `curl` command to call a trigger.

Call the redeploy trigger as follows:

```
curl https://cs.console.aliyun.com/hook/trigger?token=xxxxxxx
```

1.6.12 View pods

You can view the pods of a Kubernetes cluster in the Container Service console or in the Kubernetes dashboard.

View pods in Container Service console

1. Log on to the [Container Service console](#).
2. Under Kubernetes, click **Application** > **Pods** in the left-side navigation pane to go to the Pods page.
3. Select the target cluster and namespace, the target pod, and click **Details** on the right.



Note:

You can update or delete a pod. For pods created by using deployments, we recommend that you manage these pods by using deployments.

4. View the pod details.

View pods in Kubernetes dashboard

1. Log on to the [Container Service console](#).
2. Under Kubernetes, click **Clusters** in the left-side navigation pane.
3. Click **Dashboard** at the right of the cluster to enter the Kubernetes dashboard.
4. In the Kubernetes dashboard, click **Pods** in the left-side navigation pane to view the pods in the cluster.

You can also click **Services** in the left-side navigation pane and then click the service name to view the pods in this service.

5. You can check the status of each Kubernetes object according to the icon on the left. indicates the object is still being deployed. indicates the object has completed the deployment.
6. Click the pod name to view the details, CPU usage, and memory usage of the pod.

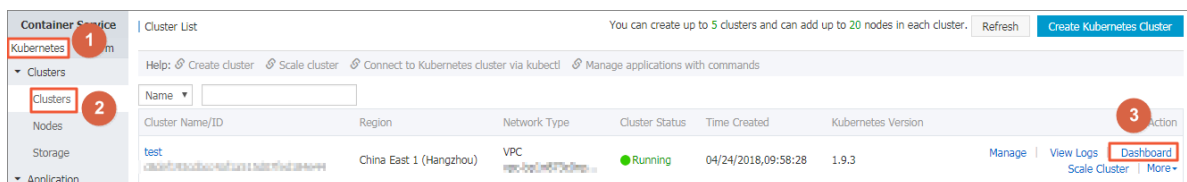
- Click **LOGS** in the upper-right corner to view the pod logs.
- You can also click the icon at the right of the pod and then select **Delete** to delete the pod.

1.6.13 Change container configurations

You can change the container configurations in the Container Service console.

Procedure

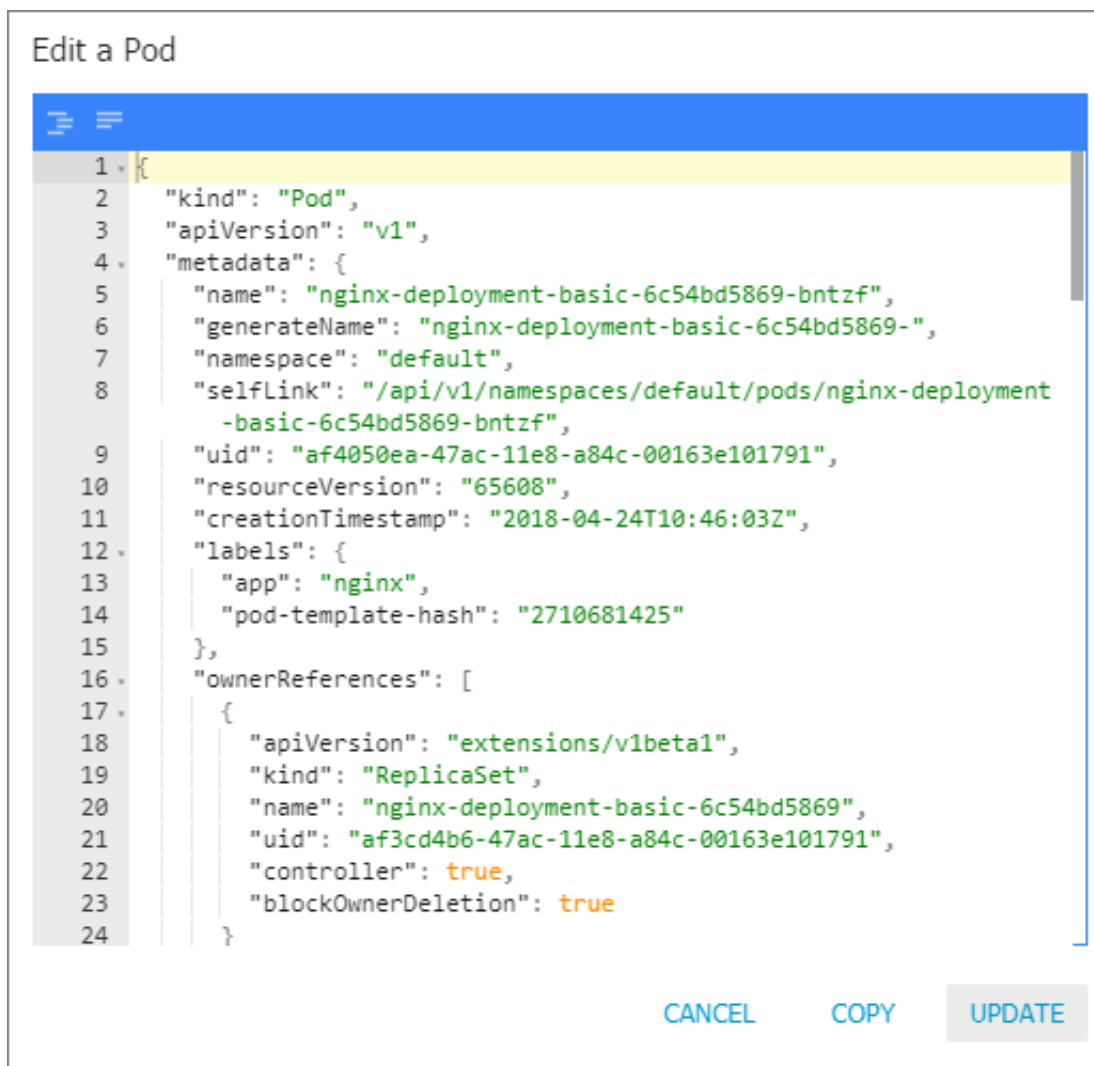
- Log on to the [Container Service console](#).
- Click **Kubernetes > Clusters** in the left-side navigation pane.
- Click **Dashboard** at the right of the cluster to enter the Kubernetes dashboard.



- In the Kubernetes dashboard, click **Pods** in the left-side navigation pane.
- Click the icon at the right of the pod and then select **View/edit YAML**.



- The Edit a Pod dialog box appears. Change the container configurations and then click **UPDATE**.



1.6.14 Schedule a pod to a specified node

You can add a node label and then configure the `nodeSelector` to schedule a pod to a specified node. For more information about the implementation principle of `nodeSelector`, see [nodeSelector](#).

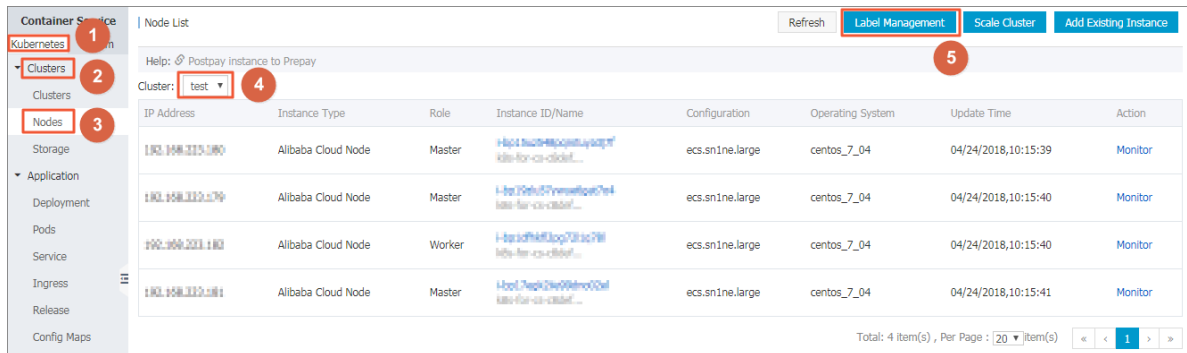
For business scenario needs, to deploy a service used for management and control to a master node, or deploy services to a machine with an SSD disk, you can use this method to schedule pods to specified nodes.

Prerequisites

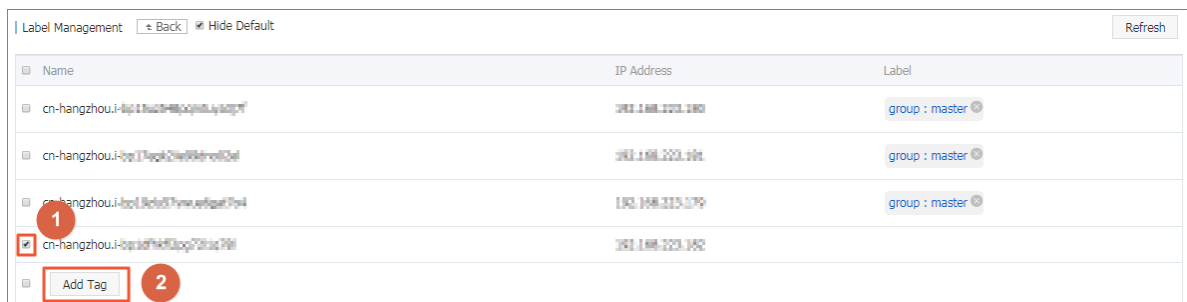
You have successfully created a Kubernetes cluster. For more information, see [Create a Kubernetes cluster](#).

Step 1 Add a node label

1. Log on to the [Container Service console](#).
2. Under the Kubernetes menu, click **Clusters** > **Nodes** in the left-side navigation pane.
3. Select the cluster from the Cluster drop-down list and then click **Label Management** in the upper-right corner.



4. Select one or more nodes by selecting the corresponding check boxes and then click **Add Tag**. In this example, select a worker node.



5. Enter the name and value of the label in the displayed dialog box and then click **OK**.



The node label `group:worker` is displayed on the Label Management page.

Label Management ← Back Hide Default			Refresh
Name	IP Address	Label	
cn-hangzhou.i- [redacted]	193.168.229.100	group : master	
cn-hangzhou.i- [redacted]	193.168.229.101	group : master	
cn-hangzhou.i- [redacted]	193.168.229.102	group : master	
cn-hangzhou.i- [redacted]	193.168.229.103	group : worker	
Add Tag			

You can also add a node label by running the command `kubectl label nodes <node-name> <label-key>=<label-value>`.

Step 2 Deploy a pod to a specified node

1. Log on to the [Container Service console](#).
2. Under the Kubernetes menu, click **Applications** > **Deployment** in the left-side navigation pane.
3. Click **Create by template** in the upper-right corner.

Container Service

Kubernetes

Clusters

Nodes

Storage

Application

Deployment

Pods

Service

Ingress

Deployment

test

Namespace

default

Create by image

Create by template

Refresh

Name	Tag	PodsQuantity	Time Created	Action
nginx-deployment-basic	app:nginx	2	04/24/2018,18:46:03	Details Update Delete
test-mariadb	app:test-mariadb chart:mariadb-0.7.0 release:test heritage:Tiller	1	04/25/2018,20:50:20	Details Update Delete
test-wordpress	app:test-wordpress chart:wordpress-0.6.13 release:test heritage:Tiller	1	04/25/2018,20:50:20	Details Update Delete

4. Configure the template to deploy a pod. After completing the configurations, click **DEPLOY**.
 - **Clusters:** Select a cluster.
 - **Namespace:** Select the namespace to which the resource object belongs. In this example, use default as the namespace.
 - **Resource Type:** Select Custom in this example.

Deploy templates

Only Kubernetes versions 1.8.4 and above are supported. For clusters of version 1.8.1, you can perform "upgrade cluster" operation in the cluster list

Clusters

test

Namespace

default

Resource Type

Custom

Template

```

1 apiVersion: v1
2 kind: Pod
3 metadata:
4   labels:
5     name: hello-pod
6     name: hello-pod
7 spec:
8   containers:
9     - image: nginx
10       imagePullPolicy: IfNotPresent
11       name: hello-pod
12       ports:
13         - containerPort: 8080
14           protocol: TCP
15       resources: {}
16       securityContext:
17         capabilities: {}
18         privileged: false
19         terminationMessagePath: /dev/termination-log
20       dnsPolicy: ClusterFirst
21       restartPolicy: Always
22       nodeSelector:
23         group: worker
24 status: {}

```

DEPLOY

The orchestration template in this example is as follows:

```

apiVersion: v1
kind: Pod
metadata:
  labels:
    name: hello-pod
    name: hello-pod
spec:
  containers:
    - image: nginx
      imagePullPolicy: IfNotPresent
      name: hello-pod
      ports:
        - containerPort: 8080
          protocol: TCP
      resources: {}
      securityContext:
        capabilities: {}
        privileged: false
        terminationMessagePath: /dev/termination-log
      dnsPolicy: ClusterFirst
      restartPolicy: Always
      nodeSelector:
        group: worker  ##The same as the node label configured in the
                        preceding step.
      status :{}

```

5. A message indicating the deployment status is displayed after you click **DEPLOY** . After the successful deployment, click **Kubernetes Dashboard** in the message to go to the dashboard and check the deployment status.

<

Cluster

Namespaces

Nodes

Persistent Volumes

Roles

Storage Classes

Namespace

default

Overview

Workloads

Cron Jobs

Daemon Sets

Deployments

Overview

nginx-deployment-basic

app: ngi...

2 / 2

2018-04-24 18:46:03

nginx:1.7.9

Pods

Name	Node	Status	Restarts	Age	CPU (cores)	Memory (bytes)
hello-pod	cn-hangzhou-i-b0t4t4k1jg73l1a27m	Running	0	2018-04-27 17:04:18	0	1.445 Mi
test-mariadb-9bb8f87dd-fjm2m	cn-hangzhou-i-b0t4t4k1jg73l1a27m	Running	0	2018-04-25 20:50:20	0.002	217.309 Mi
test-wordpress-5b74dcf48c-r8j9h	cn-hangzhou-i-b0t4t4k1jg73l1a27m	Running	0	2018-04-25 20:50:20	0.004	192.145 Mi
nginx-deployment-basic-6c54bd5869-wg2t5	cn-hangzhou-i-b0t4t4k1jg73l1a27m	Running	0	2018-04-25 12:11:48	0	1.344 Mi
nginx-deployment-basic-6c54bd5869-krpf7	cn-hangzhou-i-b0t4t4k1jg73l1a27m	Running	0	2018-04-24 18:46:03	0	1.395 Mi

6. Click the pod name to view the pod details.

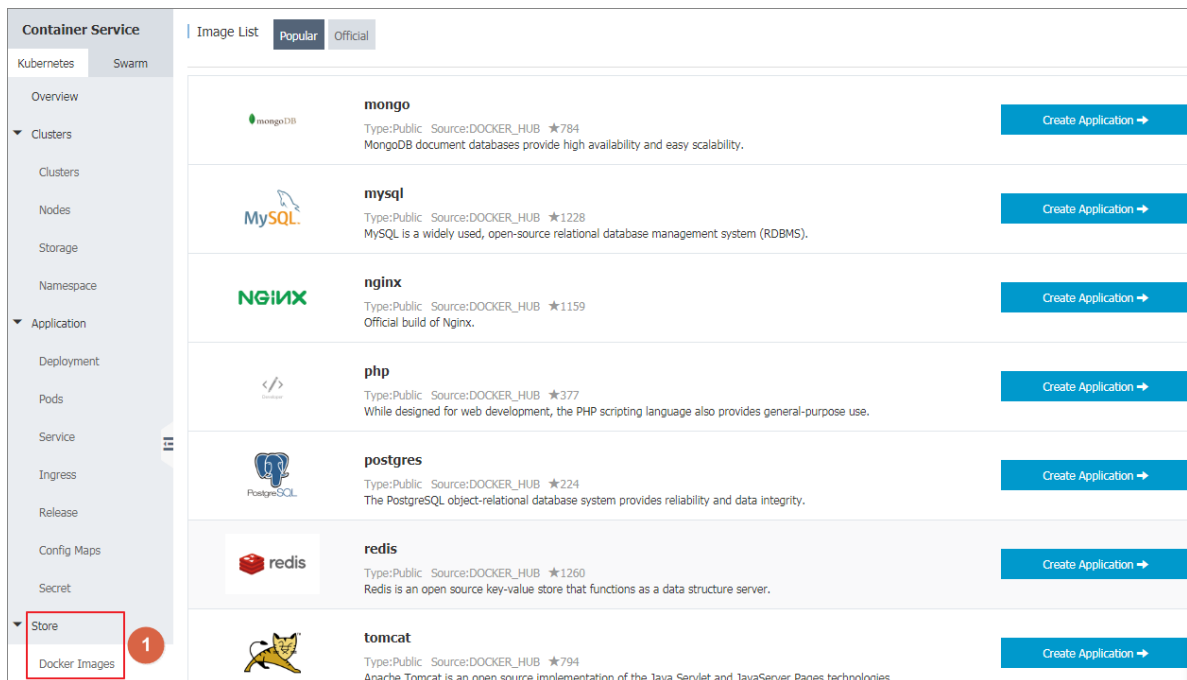
You can view the information such as the pod label and node ID, which indicates the pod is successfully deployed to a node with the label `group:worker`.



1.6.15 View image list

Procedure

1. Log on to the [Container Service console](#).
2. Under Kubernetes, click **Store** > **Docker Images** in the left-side navigation pane.



You can view the image category.

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

1.7 Server Load Balancer and Ingress

1.7.1 Overview

Kubernetes clusters provide a diversity of approaches to access container applications, and support accessing internal services and realizing load balancing by means of Alibaba Cloud Server Load Balancer or Ingress.

1.7.2 Access services by using Server Load Balancer

You can access services by using Alibaba Cloud Server Load Balancer.

Operate by using command lines

1. Create an Nginx application by using command lines.

```
root@master # kubectl run nginx --image=registry.aliyuncs.com/acs/netdia:latest
root@master # kubectl get po
NAME READY STATUS RESTARTS AGE
```

```
nginx-2721357637-dvwq3 1/1 Running 1 6s
```

2. Create Alibaba Cloud Server Load Balancer service for the Nginx application and specify `type=LoadBalancer` to expose the Nginx service to the Internet.

```
root@master # kubectl expose deployment nginx --port=80 --target-port=80 --type=LoadBalancer
root@master # kubectl get svc
NAME CLUSTER-IP EXTERNAL-IP PORT(S) AGE
nginx 172.19.10.209 101.37.192.20 80:31891/TCP 4s
```

3. Visit `http://101.37.192.20` in the browser to access your Nginx service.

Operate by using Kubernetes dashboard

1. Save the following yaml codes to the `nginx-svc.yml` file.

```
apiVersion: v1
kind: Service
metadata:
  labels:
    run: nginx
  name: http-svc
  namespace: default
spec:
  ports:
    - port: 80
      protocol: TCP
      targetPort: 80
  selector:
    run: nginx
  type: LoadBalancer
```

2. Log on to the [Container Service console](#). Under Kubernetes, click Clusters in the left-side navigation pane. Click **Dashboard** at the right of the cluster to enter the Kubernetes dashboard.
3. Click **CREATE** in the upper-right corner to create an application.



4. The Resource creation page appears. Click the **CREATE FROM FILE** tab and then upload the `nginx-svc.yml` file you saved.
5. Click **UPLOAD**.

An Alibaba Cloud Server Load Balancer instance that points to the created Nginx application is created. The service name is `http-svc`.

6. Select `default` under `Namespace` in the left-side navigation pane. Click **services** in the left-side navigation pane.

You can view the created Nginx service `http-svc` and the Server Load Balancer address `http://120.55.179.145:80`.

The screenshot shows the Kubernetes dashboard interface. On the left sidebar, the 'Services' menu item is highlighted with a red box and a red circle containing the number 1. The main panel displays the 'Services' page, which includes a table of services. The first row of the table, 'http-svc', is highlighted with a red box and a red circle containing the number 2. The table has columns for 'Name' and 'Labels'.

Name	Labels
✓ http-svc	run: ngi...
✓ nginx	app: ngi...
✓ kubernetes	component: apiser provider: kubern...

7. Copy the address to the browser to access the service.

More information

Alibaba Cloud Server Load Balancer also supports parameter configurations such as health check, billing method, and load balancing type. For more information, see [Server Load Balancer configuration parameters](#).

Annotations

Alibaba Cloud supports a plenty of Server Load Balancer features by using annotations.

Use existing intranet Server Load Balancer instance

You must specify two annotations. Replace with your own Server Load Balancer instance ID.

```
apiVersion: v1
kind: Service
metadata:
  annotations:
    service.beta.kubernetes.io/alibaba-loadbalancer-address-type:
intranet
    service.beta.kubernetes.io/alibaba-loadbalancer-id: your-
loadbalancer-id
  labels:
    run: nginx
    name: nginx
    namespace: default
spec:
  ports:
    - name: web
      port: 80
      protocol: TCP
      targetPort: 80
  selector:
    run: nginx
  sessionAffinity: None
  type: LoadBalancer
```

Save the preceding contents as `slb.svc` and then run the command `kubectl apply -f slb.svc`.

Create an HTTPS type Server Load Balancer instance

Create a certificate in the Alibaba Cloud console and record the `cert-id`. Then, use the following annotation to create an HTTPS type Server Load Balancer instance.

```
apiVersion: v1
kind: Service
metadata:
  annotations:
    service.beta.kubernetes.io/alibaba-loadbalancer-cert-id: your-
cert-id
    service.beta.kubernetes.io/alibaba-loadbalancer-protocol-port: "
https:443"
  labels:
    run: nginx
```



```

name: nginx
namespace: default
spec:
  ports:
  - name: web
    port: 443
    protocol: TCP
    targetPort: 443
  selector:
    run: nginx
  sessionAffinity: None
  type: LoadBalancer

```

**Note:**

Annotations are case sensitive.

Annotation	Description	Default value
service.beta.kubernetes.io/ alicloud-loadbalancer-protocol -port	Use commas (,) to separate multiple values. For example, https:443,http:80.	None
service.beta.kubernetes.io/ alicloud-loadbalancer-address -type	The value is Internet or intranet .	Internet
service.beta.kubernetes.io/ alicloud-loadbalancer-slb- network-type	Server Load Balancer network type. The value is classic or VPC.	classic
service.beta.kubernetes.io/ alicloud-loadbalancer-charge- type	The value is paybytraffic or paybybandwidth.	paybybandwidth
service.beta.kubernetes.io/ alicloud-loadbalancer-id	The Server Load Balancer instance ID. Use loadbalancer-id to specify an existing Server Load Balancer instance . The existing listener will be overwritten. The Server Load Balancer instance will not be deleted when the service is deleted.	None
service.beta.kubernetes.io/ alicloud-loadbalancer-backend -label	Use label to specify what nodes are mounted to the Server Load Balancer backend .	None

Annotation	Description	Default value
service.beta.kubernetes.io/alibaba-loadbalancer-region	The region in which Server Load Balancer resides.	None
service.beta.kubernetes.io/alibaba-loadbalancer-bandwidth	Server Load Balancer bandwidth.	50
service.beta.kubernetes.io/alibaba-loadbalancer-cert-id	Authentication ID on Alibaba Cloud. Upload the certificate first.	""
service.beta.kubernetes.io/alibaba-loadbalancer-health-check-flag	The value is on or off.	The default value is off. No need to modify the TCP parameters because TCP enables health check by default and you cannot configure it.
service.beta.kubernetes.io/alibaba-loadbalancer-health-check-type	See HealthCheck .	
service.beta.kubernetes.io/alibaba-loadbalancer-health-check-uri	See HealthCheck .	
service.beta.kubernetes.io/alibaba-loadbalancer-health-check-connect-port	See HealthCheck .	
service.beta.kubernetes.io/alibaba-loadbalancer-healthy-threshold	See HealthCheck .	
service.beta.kubernetes.io/alibaba-loadbalancer-unhealthy-threshold	See HealthCheck .	
service.beta.kubernetes.io/alibaba-loadbalancer-health-check-interval	See HealthCheck .	
service.beta.kubernetes.io/alibaba-loadbalancer-health-check-connect-timeout	See HealthCheck .	

Annotation	Description	Default value
service.beta.kubernetes.io/alibabacloud-loadbalancer-health-check-timeout	See HealthCheck .	

1.7.3 Support for Ingress

In Kubernetes clusters, Ingress is a collection of rules that authorize inbound connection to the cluster services and provides you with Layer-7 Server Load Balancer capabilities. You can provide the Ingress configuration with externally accessible URL, Server Load Balancer, SSL, and name-based virtual host.

Prerequisites

To test the complex routing service, create an Nginx application in this example. You must create the Nginx deployment and multiple services in advance to observe the routing effect. Replace with your own service in the actual test. In the actual test enter your own service.

```
root@master # kubectl run nginx --image=registry.cn-hangzhou.aliyuncs.com/acs/netdia:latest

root@master # kubectl expose deploy nginx --name=http-svc --port=80 --target-port=80
root@master # kubectl expose deploy nginx --name=http-svc1 --port=80 --target-port=80
root@master # kubectl expose deploy nginx --name=http-svc2 --port=80 --target-port=80
root@master # kubectl expose deploy nginx --name=http-svc3 --port=80 --target-port=80
```

Simple routing service

Create a simple Ingress service by using the following commands. All the accesses to the `/svc` path are routed to the Nginx service. `nginx.ingress.kubernetes.io/rewrite-target: /` redirects the path `/svc` to the path `/` that can be recognized by backend services.

```
root@master # cat <<EOF | kubectl create -f -
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
  name: simple
  annotations:
    nginx.ingress.kubernetes.io/rewrite-target: /
spec:
  rules:
  - http:
      paths:
      - path: /svc
        backend:
          serviceName: http-svc
```

```

        servicePort: 80
EOF
root@master # kubectl get ing
NAME HOSTS ADDRESS PORTS AGE
simple * 101.37.192.211 80 11s

```

Now visit `http://101.37.192.211/svc` to access the Nginx service.

Simple fanout routing based on domain names

If you have multiple domain names providing different external services, you can generate the following configuration to implement a simple fanout effect based on domain names:

```

root@master #cat <<EOF | kubectl create -f -
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
  name: simple-fanout
spec:
  rules:
  - host: foo.bar.com
    http:
      paths:
      - path: /foo
        backend:
          serviceName: http-svc1
          servicePort: 80
      - path: /bar
        backend:
          serviceName: http-svc2
          servicePort: 80
  - host: foo.example.com
    http:
      paths:
      - path: /film
        backend:
          serviceName: http-svc3
          servicePort: 80
EOF
root@master # kubectl get ing
NAME HOSTS ADDRESS PORTS AGE
simple-fanout * 101.37.192.211 80 11s

```

Then, you can access the `http-svc1` service by using `http://foo.bar.com/foo`, access the `http-svc2` service by using `http://foo.bar.com/bar`, and access the `http-svc3` service by using `http://foo.example.com/film`.



Note:

- In a production environment, point the domain name to the preceding returned address `101.37.192.211`.
- - In a testing environment, you can modify the `hosts` file to add a domain name mapping rule.

```
101.37.192.211 foo.bar.com
```

```
101.37.192.211 foo.example.com
```

Default domain name of simple routing

It does not matter if you do not have the domain name address. Container Service binds a default domain name for Ingress service. You can use this default domain name to access the services. The domain name is in the format of `*.[cluster-id].[region-id].alicontainer.com`. You can obtain the address on the cluster Basic Information page in the console.

Use the following configuration to expose two services with the default domain name.

```
root@master # cat <<EOF | kubectl create -f -
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
  name: shared-dns
spec:
  rules:
    - host: foo.[cluster-id].[region-id].alicontainer.com ##Replace with
      the default service access domain name of your cluster.
      http:
        paths:
          - path: /
            backend:
              serviceName: http-svc1
              servicePort: 80
    - host: bar.[cluster-id].[region-id].alicontainer.com ##Replace with
      the default service access domain name of your cluster.
      http:
        paths:
          - path: /
            backend:
              serviceName: http-svc2
              servicePort: 80
EOF
root@master # kubectl get ing
NAME HOSTS ADDRESS PORTS AGE
shared-dns foo.[cluster-id].[region-id].alicontainer.com,bar.[cluster-
id].[region-id].alicontainer.com 47.95.160.171 80 40m
```

Then, you can access the `http-svc1` service by using `http://foo.[cluster-id].[region-id].alicontainer.com/` and access the `http-svc2` service by using `http://bar.[cluster-id].[region-id].alicontainer.com`.

Configure a safe routing service

Management of multiple certificates is supported to provide security protection for your services.

1. Prepare your service certificate.

If no certificate is available, generate a test certificate in the following method:



Note:

The domain name must be consistent with your Ingress configuration.

```
root@master # openssl req -x509 -nodes -days 365 -newkey rsa:2048 -
keyout tls.key -out tls.crt -subj "/CN=foo.bar.com/O=foo.bar.com"
```

The above command generates a certificate file `tls.crt` and a private key file `tls.key`.

Create a Kubernetes secret named `foo.bar` using the certificate and private key. The secret must be referenced when you create the Ingress.

```
root@master # kubectl create secret tls foo.bar --key tls.key --cert
tls.crt
```

2. 1. Create a safe Ingress service.

```
root@master # cat <<EOF | kubectl create -f -
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
  name: tls-fanout
spec:
  tls:
  - hosts:
    - foo.bar.com
    secretName: foo.bar
  rules:
  - host: foo.bar.com
    http:
      paths:
      - path: /foo
        backend:
          serviceName: http-svc1
          servicePort: 80
      - path: /bar
        backend:
          serviceName: http-svc2
          servicePort: 80
EOF
root@master # kubectl get ing
NAME HOSTS ADDRESS PORTS AGE
tls-fanout * 101.37.192.211 80 11s
```

3. Follow the notes in **Simple fanout routing based on domain names** to configure the `hosts` file or set the domain name to access the TLS service.

You can access the `http-svc1` service by using `http://foo.bar.com/foo` and access the `http-svc2` service by using `http://foo.bar.com/bar`.

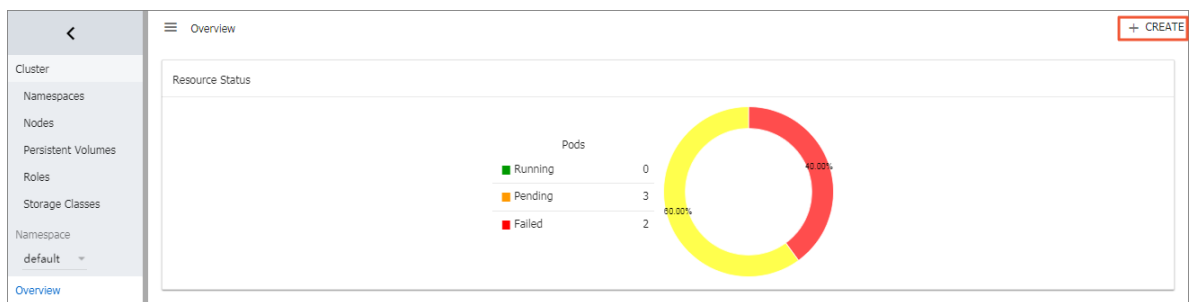
You can also access the HTTPS service by using HTTP. By default, Ingress redirects HTTP access configured with HTTPS to the HTTPS address. Therefore, access to `http://foo.bar.com/foo` will be automatically redirected to `https://foo.bar.com/foo`.

Deploy Ingress in Kubernetes dashboard

1. Save the following yml code to the `nginx-ingress.yml` file.

```
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
  name: simple
spec:
  rules:
  - http:
      paths:
      - path: /svc
        backend:
          serviceName: http-svc
          servicePort: 80
```

2. Log on to the [console](#). Under Kubernetes, click Clusters in the left-side navigation pane. Click **Dashboard** at the right of the cluster to enter the Kubernetes dashboard.
3. Click **CREATE** in the upper-right corner to create an application.

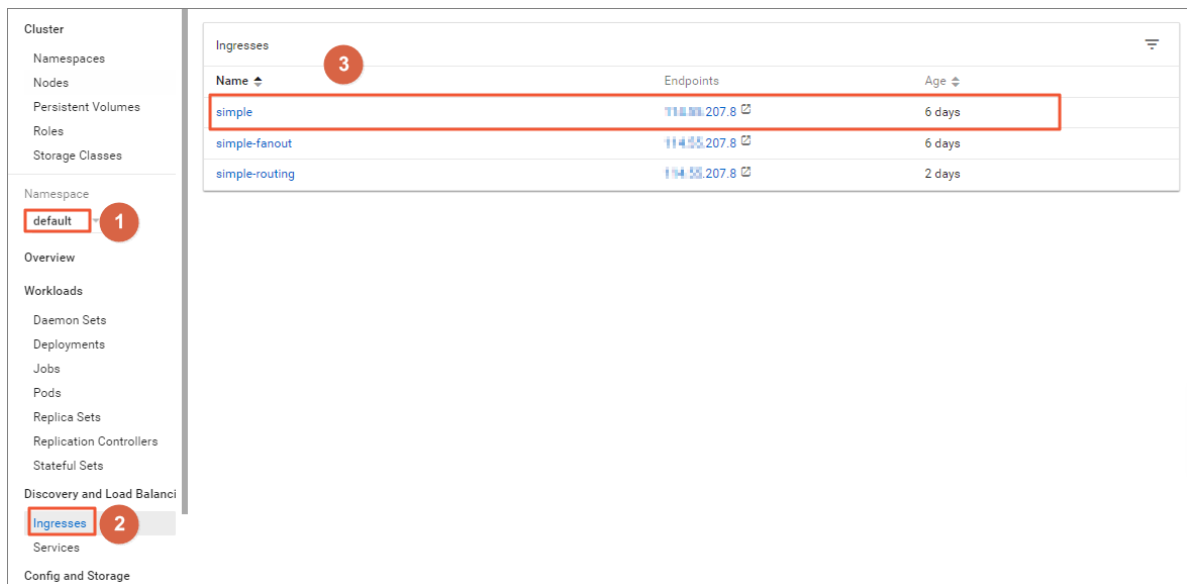


4. Click the **CREATE FROM FILE** tab. Select the `nginx-ingress.yml` file you saved.
5. Click **UPLOAD**.

Then an Ingress Layer-7 proxy route will be created to the `http-svc` service.

6. Click default under Namespace in the left-side navigation pane. Click **Ingresses** in the left-side navigation pane.

You can view the created Ingress resource and its access address `http://118.178.174.161/svc`.



7. Enter the address in the browser to access the created `http-svc` service.

1.7.4 Configure Ingress monitoring

You can view the Ingress monitoring data by enabling the default VTS module of Ingress.

Enable VTS module by running commands

1. Modify the Ingress ConfigMap configuration to add the configuration item `enable-vts-status`: `"true"`.

```
root@master # kubectl edit configmap nginx-configuration -n kube-system
configmap "nginx-configuration" edited
```

After the modification, the contents of the Ingress ConfigMap are as follows:

```
apiVersion: v1
data:
  enable-vts-status: "true" # Enable VTS module
  proxy-body-size: 20m
kind: ConfigMap
metadata:
  annotations:
    kubectl.kubernetes.io/last-applied-configuration: |

creationTimestamp: 2018-03-20T07:10:18Z
labels:
  app: ingress-nginx
name: nginx-configuration
namespace: kube-system
```



```
selfLink: /api/v1/namespaces/kube-system/configmaps/nginx-configuration
```

2. Verify if Ingress Nginx has enabled the VTS module normally.

```
root@master # kubectl get pods --selector=app=ingress-nginx -n kube-system
NAME READY STATUS RESTARTS AGE
nginx-ingress-controller-79877595c8-78gq8 1/1 Running 0 1h
root@master # kubectl exec -it nginx-ingress-controller-79877595c8-78gq8 -n kube-system -- cat /etc/nginx/nginx.conf | grep vhost_traffic_status_display
vhost_traffic_status_display;
vhost_traffic_status_display_format html;
```

3. Locally access the Ingress Nginx monitoring console.



Note:

By default, the VTS port is not opened for security considerations. Here use the port-forward method to access the console.

```
root@master # kubectl port-forward nginx-ingress-controller-79877595c8-78gq8 -n kube-system 18080
Forwarding from 127.0.0.1:18080 -> 18080
Handling connection for 18080
```

4. Use `http://localhost:18080/nginx_status` to access the VTS monitoring console.

Nginx Vhost Traffic Status

Server main

Host	Version	Uptime	Connections				Requests				Shared memory				
			active	reading	writing	waiting	accepted	handled	Total	Req/s	name	maxSize	usedSize	usedNode	
nginx-ingress-controller-79877595c8-78gq8	1.13.7	32m 41s	7	0	1	6	93566	93566	1428	1	vhost_traffic_status	10.0 MiB	2.4 KiB	1	

Server zones

Zone	Requests			Responses					Traffic						Cache							
	Total	Req/s	Time	1xx	2xx	3xx	4xx	5xx	Total	Sent	Rcvd	Sent/s	Rcvd/s	Miss	Bypass	Expired	Stale	Updating	Revalidated	Hit	Scarc	Total
-	660	1	0ms	0	660	0	0	0	660	1.7 MiB	145.4 KiB	1.1 KiB	503 B	0	0	0	0	0	0	0	0	0
*	660	1	0ms	0	660	0	0	0	660	1.7 MiB	145.4 KiB	1.1 KiB	503 B	0	0	0	0	0	0	0	0	0

Upstreams

upstream-default-backend

Server	State	Response Time	Weight	MaxFails	FailTimeout	Requests			Responses						Traffic				
						Total	Req/s	Time	1xx	2xx	3xx	4xx	5xx	Total	Sent	Rcvd	Sent/s	Rcvd/s	
172.16.3.6:8080	up	0ms	1	0	0	0	0	0ms	0	0	0	0	0	0	0	0 B	0 B	0 B	0 B

update interval: 1 sec

[JSON](#) | [GITHUB](#)

Enable VTS module by using the Kubernetes dashboard

1. Log on to the [Container Service console](#).
2. On the Cluster List page of Kubernetes clusters, click **Dashboard** at the right of a cluster to enter the Kubernetes dashboard page.

3. Select kube-system under Namespace in the left-side navigation pane. Click Config Maps in the left-side navigation pane. Click the icon at the right of nginx-configuration and then select View/edit YAML. Edit the config map to add the configuration item `enable-vts-status: "true"`.

The contents of the saved Ingress ConfigMap are as follows:

```
"kind": "ConfigMap",
"apiVersion": "v1",
"metadata": {
  "name": "nginx-configuration",
  "namespace": "kube-system",
  "selfLink": "/api/v1/namespaces/kube-system/configmaps/nginx-configuration",
  "creationTimestamp": "2018-03-20T07:10:18Z",
  "labels": {
    "app": "ingress-nginx"

    "annotations": {
      "kubectl.kubernetes.io/last-applied-configuration": "{\n  \"apiVersion\": \"v1\", \"data\": {\n    \"proxy-body-size\": \"20m\" \n  }, \"kind\": \"ConfigMap\", \"metadata\": {\n    \"annotations\": {}, \"labels\": {\n      \"app\": \"ingress-nginx\" \n    }, \"name\": \"nginx-configuration\", \"namespace\": \"kube-system\" \n  } }\n"

"data": {
  "proxy-body-size": "20m",
  "enable-vts-status": "true"
```

4. Locally access the Ingress Nginx monitoring console.



Note:

By default, the VTS port is not opened for security considerations. Here use the port-forward method to access the console.

```
root@master # kubectl port-forward nginx-ingress-controller-79877595c8-78gq8 -n kube-system 18080
Forwarding from 127.0.0.1:18080 -> 18080
Handling connection for 18080
```

5. Use `http://localhost:18080/nginx_status` to access the VTS monitoring console.

Nginx Vhost Traffic Status

Server main

Host	Version	Uptime	Connections				Requests				Shared memory			
			active	reading	writing	waiting	accepted	handled	Total	Req/s	name	maxSize	usedSize	usedNode

nginx-ingress-controller-79877595c8-78gq8	1.13.7	32m 41s	7	0	1	6	93566	93566	1428	1	vhost_traffic_status	10.0 MiB	2.4 KiB	1
---	--------	---------	---	---	---	---	-------	-------	------	---	----------------------	----------	---------	---

Server zones

Zone	Requests			Responses						Traffic				Cache								
	Total	Req/s	Time	1xx	2xx	3xx	4xx	5xx	Total	Sent	Rcvd	Sent/s	Rcvd/s	Miss	Bypass	Expired	Stale	Updating	Revalidated	Hit	Scare	Total
•	660	1	0ms	0	660	0	0	0	660	1.7 MIB	145.4 KIB	1.1 KIB	503 B	0	0	0	0	0	0	0	0	0
	660	1	0ms	0	660	0	0	0	660	1.7 MIB	145.4 KIB	1.1 KIB	503 B	0	0	0	0	0	0	0	0	0

-	660	1	0ms	0	660	0	0	0	660	1.7 MiB	145.4 KiB	1.1 KiB	503 B	0	0	0	0	0	0	0	0	0
---	-----	---	-----	---	-----	---	---	---	-----	---------	-----------	---------	-------	---	---	---	---	---	---	---	---	---

*	660	1	0ms	0	660	0	0	0	660	1.7 MiB	145.4 KiB	1.1 KiB	503 B	0	0	0	0	0	0	0	0	0
---	-----	---	-----	---	-----	---	---	---	-----	---------	-----------	---------	-------	---	---	---	---	---	---	---	---	---

Upstreams

upstream-default-backend

Server	State	Response Time	Weight	MaxFails	FailTimeout	Requests			Responses						Traffic					
						Total	Req/s	Time	1xx	2xx	3xx	4xx	5xx	Total	Sent	Rcvd	Sent/s	Rcvd/s		
172.16.3.6:8080	up	0ms	1	0	0	0	0	0ms	0	0	0	0	0	0	0	0	0	0	0	0

172.16.3.6:8080	up	0ms	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
-----------------	----	-----	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

update interval: sec

[JSON](#) | [GITHUB](#)

1.7.5 Ingress configurations

Alibaba Cloud Container Service provides the highly reliable Ingress controller components and integrates with Alibaba Cloud Server Load Balancer to provide the flexible and reliable Ingress service for your Kubernetes clusters.

See the following Ingress orchestration example. You must configure the annotation when creating an Ingress in the Container Service console. Some configurations must create dependencies. For more information, see [Create an Ingress in Container Service console](#), [Support for Ingress](#), and [Kubernetes Ingress](#). Ingress also supports the configuration of configmap. For more information, see <https://kubernetes.github.io/ingress-nginx/user-guide/nginx-configuration/configmap/>.

```
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
  Annotations:
    nginx.ingress.kubernetes.io/service-match: 'new-nginx: header("foo", /^bar$/)' #Grayscale publish rule, this example is request header
    Nginx.ingress.kubernetes.io/service-weight: 'New-nginx: 50, old-nginx: 50' # FTraffic weight annotation
    creationTimestamp: null
    generation: 1
    name: nginx-ingress
    selfLink: /apis/extensions/v1beta1/namespaces/default/ingresses/nginx-ingress
spec:
  rules: ##The Ingress rule
  - host: foo.bar.com
    http:
      paths:
      - backend:
          serviceName: new-nginx
          servicePort: 80
```

```
    path: /
  - backend:
      serviceName: old-nginx
      servicePort: 80
    path: /
tls: ## Enable TLS to configure the secure Ingress service
  - hosts:
      - *.xxxxxx.cn-hangzhou.alicontainer.com
      - foo.bar.com
    secretName: nginx-ingress-secret ##The name of the used secret.
status:
  loadBalancer: {}
```

Annotation

You can configure an ingress annotation, specifying the ingress controller to use, rules for routing, such as routing weight rules, grayscale publish, and rewrite rules. For more information, see <https://kubernetes.github.io/ingress-nginx/user-guide/nginx-configuration/annotations/>.

For example, a typical rewrite annotation `nginx.ingress.kubernetes.io/rewrite-target` : `/` redirects the path `/path` to the path `/` that can be recognized by the backend services.

Rules

The rules indicate those that authorize the inbound access to the cluster and are generally the HTTP rules, including the domain name (virtual hostname), URL access path, service name, and port.

You must complete the following configurations for each HTTP rule:

- Host: Enter the testing domain name of an Alibaba Cloud Kubernetes cluster or a virtual hostname, such as `foo.bar.com`.
- Path: Specify the URL path of the service access. Each path is associated with a backend service. Before Alibaba Cloud Server Load Balancer forwards the traffic to the backend, all inbound requests must match with the domain name and path.
- Backend configuration: Service configuration that is a combination of `service:port` and traffic weight. The Ingress traffic is forwarded to the matched backend services based on the traffic weight.
 - Name: The name of the backend service forwarded by Ingress.
 - Port: The port exposed by the service.
 - Weight: The weight rate of each service in a service group.



Note:

1. The service weight is calculated in relative values. For example, if both service weights are set to 50, the weight ratio of both services is 50%.
2. A service group (a service with the same Host and Path in the same ingress yaml) has a default weight value of 100 and the weight is not explicitly set.

Grayscale publish

Container Service supports different traffic segmentation methods for grayscale publish and AB test scenarios.



Note:

Currently, the Alibaba Cloud Container Service Kubernetes Ingress Controller requires 0.12.0-5 and above to support the traffic segmentation feature.

1. Traffic segmentation based on the request header.
2. Traffic segmentation based on cookie.
3. Traffic segmentation based on query (request) parameters.

After the grayscale rule is configured, the request that matches the grayscale publish rule can be routed to the set service. If the service sets a weight rate of less than 100%, requests that match the grayscale publish rule continue to be routed to the corresponding service based on the weight rate.

TLS

You can encrypt the Ingress by specifying a secret that contains the TLS private key and certificate to implement the secure Ingress access. The TLS secret must contain the certificate named `tls.crt` and private key named `tls.key`. For more information about the TLS principles, see [TLS](#). For how to create a secret, see [Configure a safe routing service](#).

Label

You can add tags for Ingress to indicate the characteristics of the Ingress.

1.7.6 Create an Ingress in Container Service console

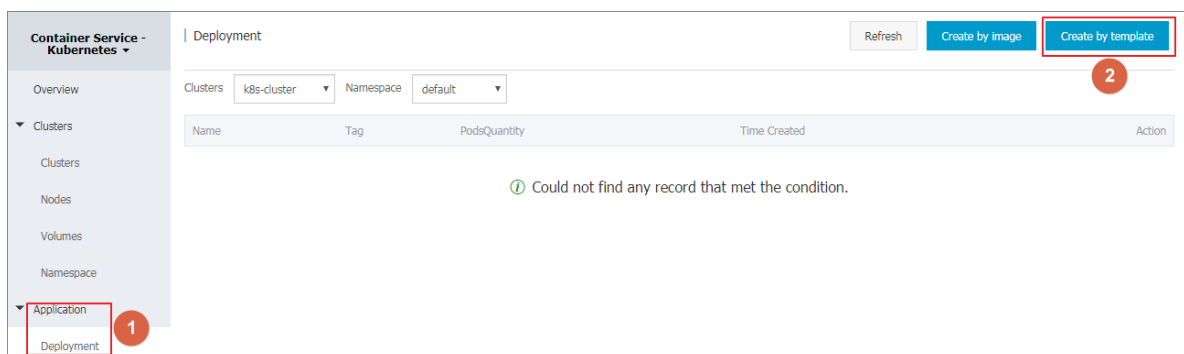
Alibaba Cloud Container Service console integrates with the Ingress service, which allows you to quickly create an Ingress service in the Container Service console to build the flexible and reliable traffic access layer.

Prerequisites

- You have successfully created a Kubernetes cluster and Ingress controller is running normally in the cluster. For how to create a Kubernetes cluster, see [Create a Kubernetes cluster](#).
- Log on to the master node by using SSH. For more information, see [#unique_16](#).

Step 1. Create a deployment and a service

- Log on to the [Container Service console](#).
- Under Kubernetes, click **Application** > **Deployment** in the left-side navigation pane to enter the Deployment List page.
- Click **Create by template** in the upper-right corner.



- Select the cluster and namespace from the Clusters and Namespace drop-down lists. Select a sample template or Custom from the Resource Type drop-down list. Click **DEPLOY**.

In this example, three nginx applications are created. One for the old application (old-nginx), one for the new (new-nginx), and an application for testing the cluster access domain name (domain-nginx).

Clusters: k8s-cluster

Namespace: default

Resource Type: Resource - basic Deployment

Template

```

1  apiVersion: extensions/v1beta1
2  kind: Deployment
3  metadata:
4    name: new-nginx
5  spec:
6    replicas: 1
7    selector:
8      matchLabels:
9        run: new-nginx
10   template:
11     metadata:
12       labels:
13         run: new-nginx
14     spec:
15       containers:
16       - image: registry.cn-hangzhou.aliyuncs.com/xianlu/new-nginx
17         imagePullPolicy: Always
18         name: new-nginx
19         ports:
20         - containerPort: 80
21           protocol: TCP
22         restartPolicy: Always
23   ---
24   apiVersion: extensions/v1beta1
25   kind: Deployment
26   metadata:
27     name: new-nginx
28   spec:
29     replicas: 1
30     selector:
31       matchLabels:
32         run: new-nginx
33     template:
34       metadata:
35         labels:
36

```

Add Deployment

DEPLOY

The orchestration template for old-nginx is as follows:

```

apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: old-nginx
spec:
  replicas: 2
  selector:
    matchLabels:
      run: old-nginx
  template:
    metadata:
      labels:
        run: old-nginx
    spec:
      containers:
      - image: registry.cn-hangzhou.aliyuncs.com/xianlu/old-nginx
        imagePullPolicy: Always
        name: old-nginx
        ports:
        - containerPort: 80
          protocol: TCP
        restartPolicy: Always
    ---
apiVersion: v1
kind: Service
metadata:
  name: old-nginx
spec:
  ports:
  - port: 80
    protocol: TCP
    targetPort: 80
  selector:
    run: old-nginx
  sessionAffinity: None

```

```
type: NodePort
```

The orchestration template for new-nginx is as follows:

```
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: new-nginx
spec:
  replicas: 1
  selector:
    matchLabels:
      run: new-nginx
  template:
    metadata:
      labels:
        run: new-nginx
    spec:
      containers:
        - image: registry.cn-hangzhou.aliyuncs.com/xianlu/new-nginx
          imagePullPolicy: Always
          name: new-nginx
          ports:
            - containerPort: 80
              protocol: TCP
          restartPolicy: Always
---
apiVersion: v1
kind: Service
metadata:
  name: new-nginx
spec:
  ports:
    - port: 80
      protocol: TCP
      targetPort: 80
  selector:
    run: new-nginx
  sessionAffinity: None
  type: NodePort
```

The orchestration template for domain-nginx is as follows:

```
apiVersion: apps/v1beta2 # For versions before 1.8.0 use apps/
v1beta1
kind: Deployment
metadata:
  name: domain-nginx
  labels:
    app: nginx
spec:
  replicas: 2
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
```



```

    containers:
      - name: nginx
        image: nginx:1.7.9 # replace it with your exactly <
image_name:tags>
        ports:
          - containerPort: 80

---
apiVersion: v1
kind: Service
metadata:
  name: domain-nginx
spec:
  ports:
    - port: 80
      protocol: TCP
      targetPort: 80
  selector:
    app: nginx
  sessionAffinity: None
  type: NodePort

```

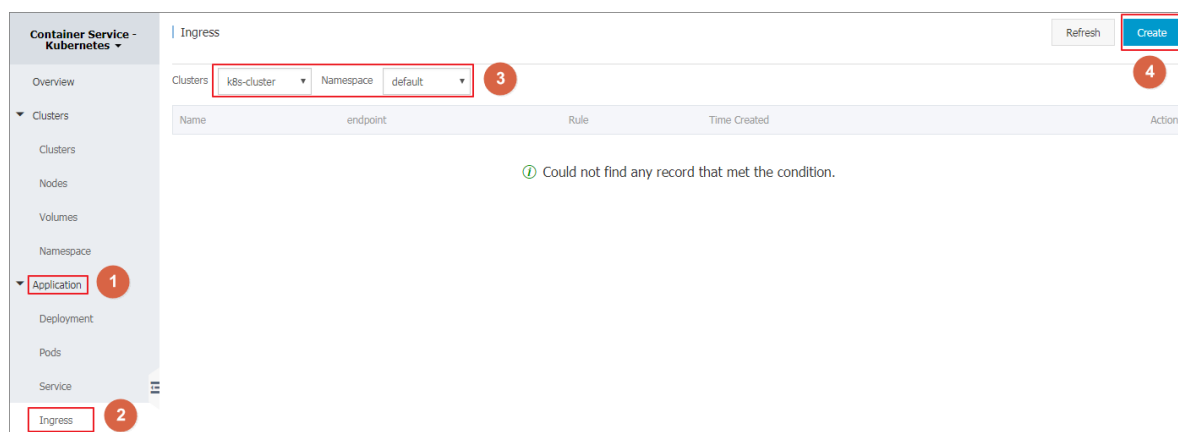
5. Click **Application** > **Service** in the left-side navigation pane to enter the Services List page.

After the service is created, you can see it on the Service List page.

Service List							Refresh	Create
Clusters	k8s-cluster	Namespace	default					
Name	Type	Time Created	ClustersIP	InternalEndpoint	ExternalEndpoint	Action		
domain-nginx	NodePort	07/11/2018,17:43:32		domain-nginx:80 TCP domain-nginx:32347 TCP	-	Details	Update	View YAML Delete
kubernetes	ClusterIP	07/11/2018,17:35:35		kubernetes:443 TCP	-	Details	Update	View YAML Delete
new-nginx	NodePort	07/11/2018,17:37:01		new-nginx:80 TCP new-nginx:32637 TCP	-	Details	Update	View YAML Delete
old-nginx	NodePort	07/11/2018,17:37:01		old-nginx:80 TCP old-nginx:32039 TCP	-	Details	Update	View YAML Delete

Step 2. Create an Ingress

1. Log on to the [Container Service console](#).
2. Under Kubernetes, click **Application** > **Ingress** in the left-side navigation pane.
3. Select the cluster and namespace from the Clusters and Namespace drop-down lists. Then click **Create** in the upper-right corner.



4. In the displayed dialog box, enter the Ingress name. In this example, enter nginx-ingress.

Name:

5. Configure the rules.

The Ingress rules are the rules that authorize the inbound access to the cluster and are generally the HTTP rules. Configure the domain name (virtual hostname), URL path, service name, and port. For more information, see [Ingress configurations](#).

In this example, add a complicated Ingress rule. Configure the default test domain name and virtual hostname of the cluster to display the Ingress service based on the domain names.

Rule: + Add

Domain ✖

test.c[REDACTED].cn-hangzhou.alicontainer.com

Select *, c[REDACTED].cn-hangzhou.alicontainer.com or Custom

path

/

Service + Add

Name	Port	Weight	Percent of Weight	
domain-nginx	80	100	100.0%	-

Domain ✖

foo.bar.com

Select *, c[REDACTED].cn-hangzhou.alicontainer.com or Custom

path

/

Service + Add

Name	Port	Weight	Percent of Weight	
new-nginx	80	50	50.0%	-
old-nginx	80	50	50.0%	-

- The simple Ingress based on the default domain name, that is, provide the access service externally by using the default domain name of the cluster.

— **Domain:** Enter the default domain name of the cluster. In this example, use `test.[cluster-id].[region-id].alicontainer.com`.

The default domain name of this cluster is displayed in the Create dialog box, in the `*.[cluster-id].[region-id].alicontainer.com` format. You can also obtain the default domain name on the Basic Information page of the cluster.

— **Service:** Configure the access path, name, and port of the service.

- **Path:** Specify the URL path of the service access. The default is the root path `/`, which is not configured in this example. Each path is associated with a backend service. Before Alibaba Cloud Server Load Balancer forwards the traffic to the backend, all inbound requests must match with the domain name and path.

- **Service configuration:** The backend configuration, which is a combination of service name, port, and service weight. The configuration of multiple services in the same access path is supported, and Ingress traffic is split and is forwarded to the matched backend services.
- The simple fanout Ingress based on the domain name. In this example, use a virtual hostname as the testing domain name to provide the access service externally. You can use the recorded domain name in the production environment to provide the access service. You can use the recorded domain name in the production environment to provide the access service.

— **Domain:** In this example, use the testing domain name `foo.bar.com`.

You must modify the hosts file to add a domain name mapping rule.

```
118.178.108.143 foo.bar.com # Ingress IP address
```

— **Service:** Configure the access path, name, and port of the service.

- **Path:** Specify the URL path of the service access. Path is not configured in this example, and the root path is `/`.
- **Name:** In this example, set up both new and old services, `nginx-new` and `nginx-old`.
- **Port:** Expose 80 port.
- **Weight settings:** Set the weight of multiple services under this path. The service weight is calculated by relative value. The default value is 100. As shown in this example, the service weight values of both the old and new versions are 50, which means that the weight rate of both services is 50%.

6. Grayscale publish configuration.



Note:

Currently, the Alibaba Cloud Container Service Kubernetes Ingress Controller requires 0.12.0-5 and above to support the traffic segmentation feature.


Container Service supports different traffic segmentation methods for grayscale publish and AB test scenarios.

1. Traffic segmentation based on the request header.
2. Traffic segmentation based on cookie.
3. Traffic segmentation based on query (request) parameters.

After the grayscale rule is configured, the request that matches the grayscale publish rule can be routed to the new service version new-nginx. If the service sets a weight rate of less than 100%, requests that match the grayscale publish rule continue to be routed to the corresponding service based on the weight rate.

In this case, set the request header to meet a grayscale publish rule of `foo=^bar$`, only requests with the request header can access the new-nginx service.

Grayscale release:

 **Add** After the gray rule is set, the request meeting the rule will set a weight other than 100, the request to satisfy the gamma rule and old version services according to the weights.

Service	Type	Name
nginx-new	Header	foo:bar

- **Service:** Routing rule configuration service.
- **Type:** matching request header, cookie, and query (request) parameters are supported.
- **Name and match value:** User-defined request field, name and match value are key-value pairs.
- **Match rules:** Regular and exact matches are supported.

7. Configure the annotations.

Click **rewrite annotation**, a typical redirection annotation can be added to the route. `nginx.ingress.kubernetes.io/rewrite-target: /` indicates that the `/path` is redirected to the root path `/` that the backend service can recognize.



Note:

In this example, the access path is not configured, so no need to configure rewrite annotations. The purpose of the rewrite annotation is to enable Ingress to forward to the backend as the root path, avoiding 404 errors caused by incorrect access path configuration.

You can also click **Add** to enter the annotation name and value, which is the annotation key-value pair for Ingress. For more information, see <https://kubernetes.github.io/ingress-nginx/user-guide/nginx-configuration/annotations/>.

annotation: [+ Add rewrite annotation](#)

Name	Value
nginx.ingress.kubernetes.io/rewri	/

8. Configure TLS. Select **Enable** and configure the secure Ingress service. For more information, see [Configure a safe routing service](#).

- You can select to use an existing secret.

TLS: ☒ Enable ☐ Exist secret ☐ Create secret

foo.bar

1. Log on to the master node and create `tls.key` and `tls.crt`.

```
openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout tls.key -out tls.crt -subj "/CN=foo.bar.com/O=foo.bar.com"
```

2. Create a secret.

```
kubectl create secret tls foo.bar --key tls.key --cert tls.crt
```

3. Run the `kubectl get secret` command to see that secret has been successfully created. You can use the secret that you have created in the Web interface, `foo.bar`.

- You can create the secret with one click by using the created TLS private key and certificate.

TLS: ☒ Enable ☐ Exist secret ☒ Create secret

Cert

```
-----BEGIN CERTIFICATE-----
MIIDKzCCAOhOgAwIBAgIJAJCsMUrmv4M8MA0GCSqGSIb3DQEBCwUAMCwx
FDASBgNV
```

Key

```
-----BEGIN PRIVATE KEY-----
MIIEvQIBADANBgkqhkiG9w0BAQEFAASCBAKcwggSjAgEAAoIBAQD1lkopjVh
tFBWn
```

1. Log on to the master node and create `tls.key` and `tls.crt`.

```
openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout tls.key -out tls.crt -subj "/CN=foo.bar.com/O=foo.bar.com"
```

2. Run the `vim tls.key` and `vim tls.crt` to get the generated private key and certificate.
3. Copy the generated certificate and private key to the Cert and Key fields.

9. Adding the tags.

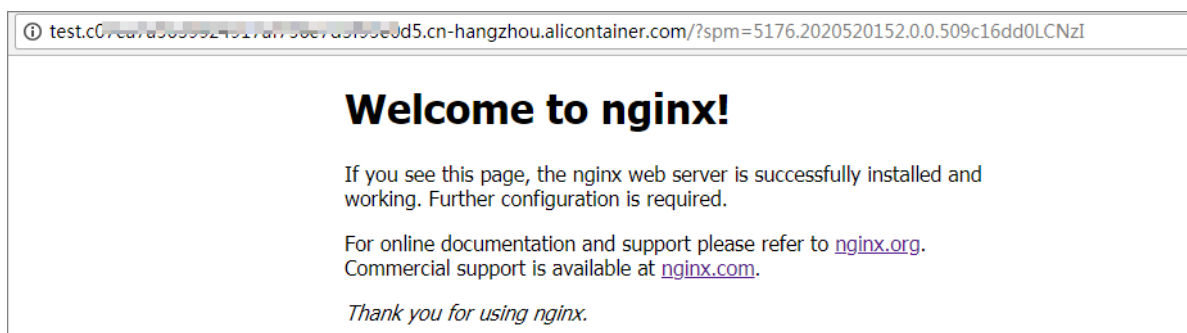
Add the corresponding tags for Ingress to indicate the characteristics of the Ingress.

10. Click **Create**.

The Ingress nginx-ingress is displayed on the Ingress page.

Name	endpoint	Rule	Time Created	Action
nginx-ingress		test.c07ea7a5639924917af756e7d3f95e0d5.cn-hangzhou.alicontainer... -> domain-nginx foo.bar.com/ -> new-nginx foo.bar.com/ -> old-nginx	07/11/2018,17:49:46	Details Update View YAML Delete

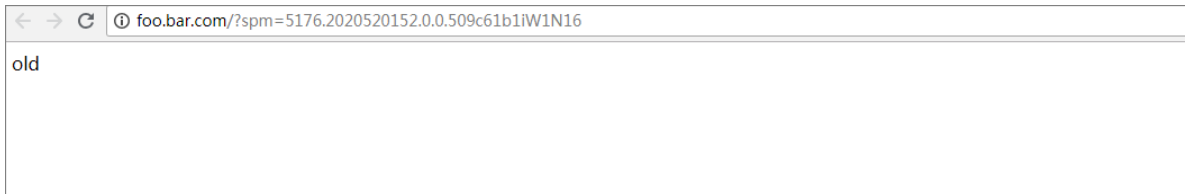
11. Click on the access domain name `test.[cluster-id].[region-id].alicontainer.com` in the route, and `foo.bar.com` to access the welcome page of nginx.



Click on the route address pointing the new-nginx service and find the page that points the old-nginx application.

**Note:**

Access the route address in the browser. By default, the request header does not have the `foo=^bar$`, so the traffic is directed to the old-nginx application.



12. Log on to the master node by using SSH. Run the following command to simulate the access result with a specific request header.

```
curl -H "Host: foo.bar.com" http://47.107.20.35
old
curl -H "Host: foo.bar.com" http://47.107.20.35
old
curl -H "Host: foo.bar.com" http://47.107.20.35 # Similar to
browser access requests
old
curl -H "Host: foo.bar.com" -H "foo: bar" http://47.107.20.35 #
Simulate an access request with a unique header, returning results
based on routing weight
new
curl -H "Host: foo.bar.com" -H "foo: bar" http://47.107.20.35
old
curl -H "Host: foo.bar.com" -H "foo: bar" http://47.107.20.35
old
curl -H "Host: foo.bar.com" -H "foo: bar" http://47.107.20.35
new
```

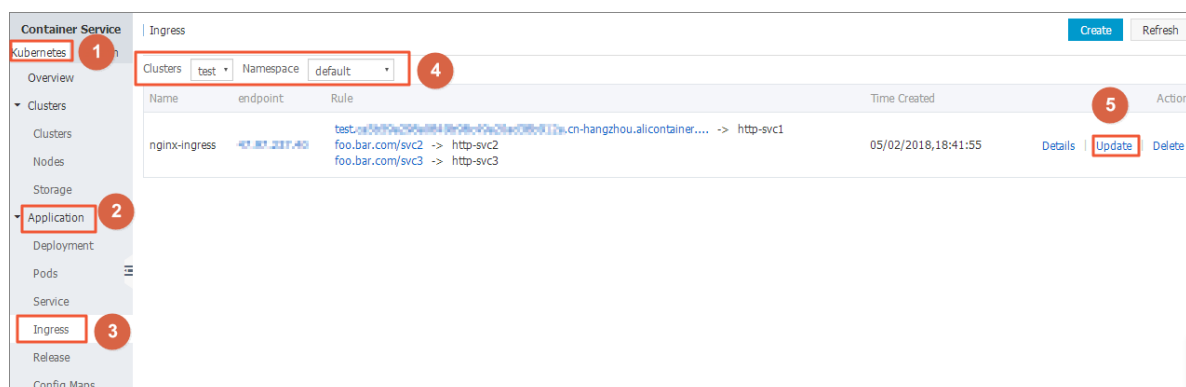
1.7.7 Update an Ingress

Prerequisites

- You have successfully created a Kubernetes cluster and Ingress controller is running normally in the cluster. For how to create a Kubernetes cluster, see [Create a Kubernetes cluster](#).
- You have successfully created an Ingress. For more information, see [Create an Ingress in Container Service console](#).

Procedure

- Log on to the [Container Service console](#).
- Click **Kubernetes > Application > > Ingress** in the left-side navigation pane.
- Select the cluster and namespace from the Clusters and Namespace drop-down lists. Click **Update** at the right of the Ingress.



4. Update the Ingress parameters in the displayed dialog box and then click **OK**. In this example, change `test.[cluster-id].[region-id].alicontainer.com` to `testv2.[cluster-id].[region-id].alicontainer.com`.



What's next

On the Ingress page, you can see a rule of this Ingress is changed.



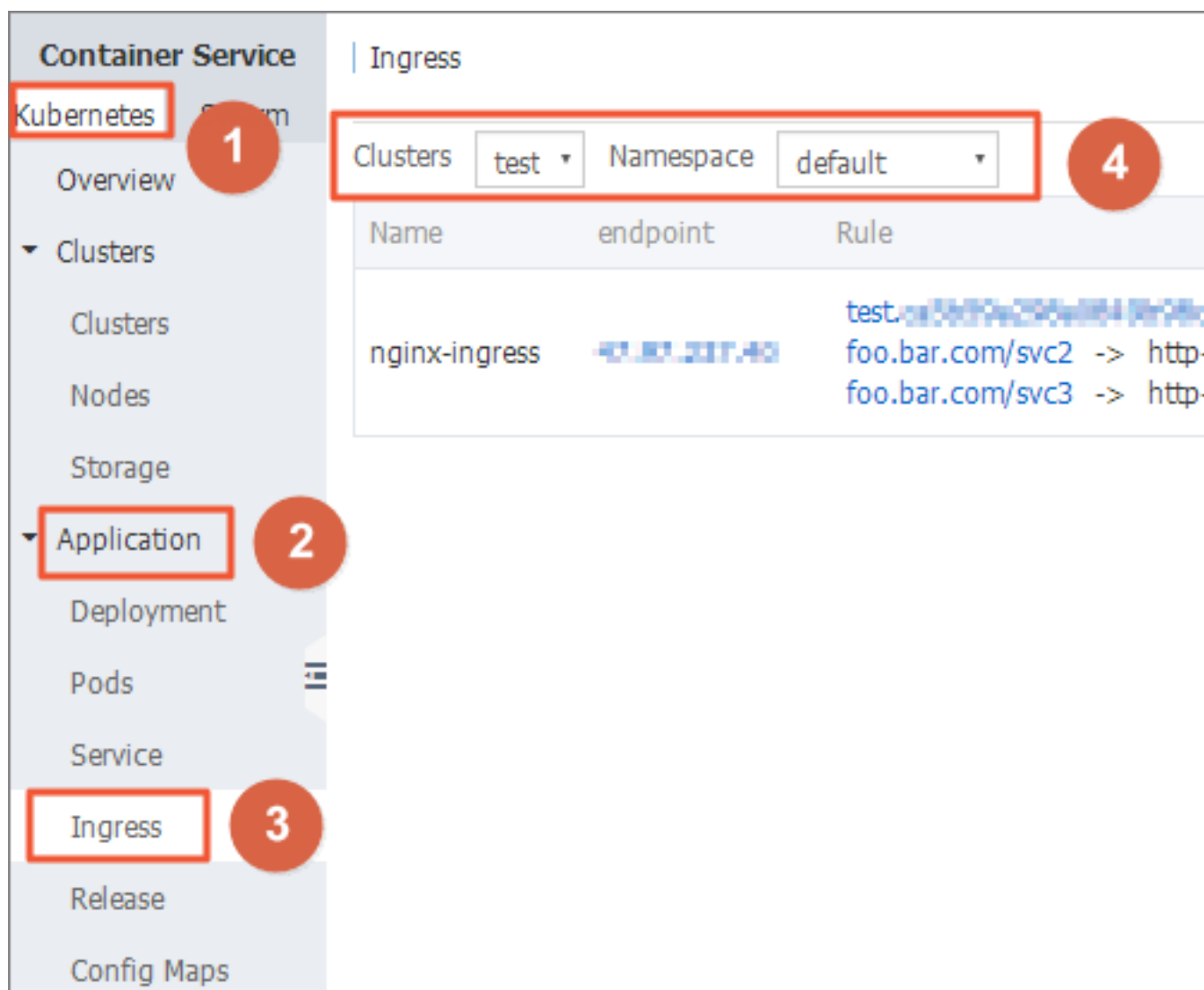
1.7.8 View Ingress details

Prerequisites

- You have successfully created a Kubernetes cluster and Ingress controller is running normally in the cluster. For how to create a Kubernetes cluster, see [Create a Kubernetes cluster](#).
- You have successfully created an Ingress. For more information, see [Create an Ingress in Container Service console](#).

Procedure

1. Log on to the [Container Service console](#).
2. Click **Kubernetes > Application > Ingress** in the left-side navigation pane.
3. Select the cluster and namespace from the Clusters and Namespace drop-down lists. Click **Details** at the right of the Ingress.



On the details page, you can view the overview and rules of the Ingress.

nginx-ingress

← Back to List

Refresh

Overview

Name:	nginx-ingress
Namespace:	default
Time Created:	2018-05-02T10:41:55Z
Label:	node-role.kubernetes.io/ingress:true
annotation:	nginx.ingress.kubernetes.io/rewrite-target:/
endpoint:	172.17.0.1:80

Rule

Domain	path	Name	service port
test.cn-hangzhou.aliyuncs.com	/svc1	http-svc1	80
foo.bar.com	/svc2	http-svc2	80
foo.bar.com	/svc3	http-svc3	80

1.7.9 Delete an Ingress

Prerequisites

- You have successfully created a Kubernetes cluster and Ingress controller is running normally in the cluster. For how to create a Kubernetes cluster, see [Create a Kubernetes cluster](#).
- You have successfully created an Ingress. For more information, see [Create an Ingress in Container Service console](#).

Procedure

1. Log on to the [Container Service console](#).
2. Click Kubernetes > **Application** > > **Ingress** in the left-side navigation pane.
3. Select the cluster and namespace from the Clusters and Namespace drop-down lists. Click **Delete** at the right of the Ingress.

Container Service | Ingress Create Refresh

Kubernetes 1

Overview

Clusters 2

Clusters

Nodes

Storage

Application 2

Deployment

Pods

Service

Ingress 3

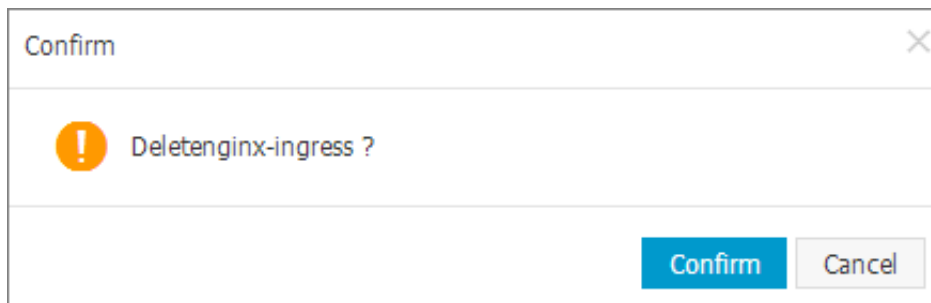
Release

Config Man

Clusters test Namespace default 4

Name	endpoint	Rule	Time Created	Action
nginx-ingress	192.168.1.100:80	test.foo.bar.com/svc2 -> http-svc2 foo.bar.com/svc3 -> http-svc3	05/02/2018,18:41:55	Details Update Delete 5

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



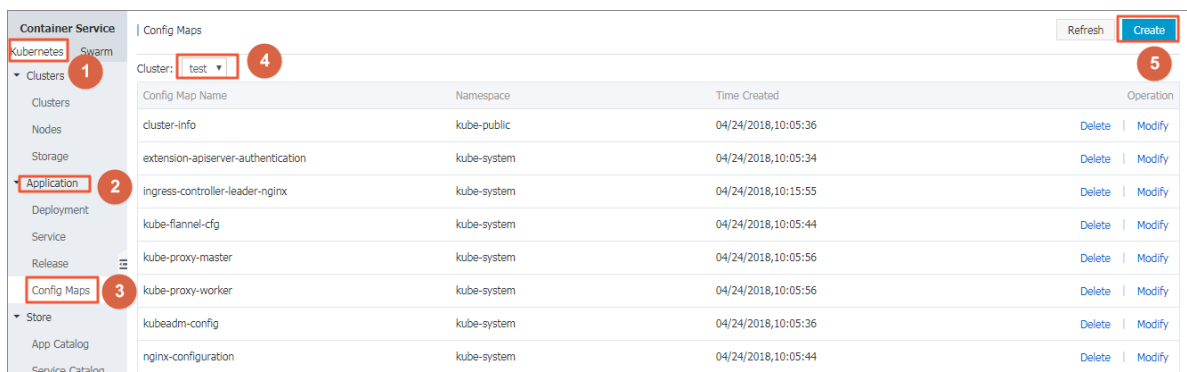
1.8 Config maps and Secrets

1.8.1 Create a config map

In the Container Service console, you can create a config map on the Config Maps page or by using a template.

Create a config map on Config Maps page

1. Log on to the [Container Service console](#).
2. Under Kubernetes, click **Application** > **Config Maps** in the left-side navigation pane.
3. Select the cluster from the Cluster drop-down list. Click **Create** in the upper-right corner.



4. Complete the settings and then click **OK**.
 - **Namespace:** Select the namespace to which the config map belongs. Config map is a Kubernetes resource object that must be applied to the namespace.
 - **Config Map Name:** Enter the config map name, which can contain lowercase letters, numbers, hyphens (-), and periods (.). The name cannot be empty. Other resource objects must reference the config map name to obtain the configuration information.
 - **Configuration:** Enter the Variable Name and the Variable Value. Then, click **Add** on the right. You can also click **Edit**, complete the configuration in the displayed dialog box, and click **OK**.

* Namespace:

default

* Config Map Name:

test-config

Name must consist of lowercase alphanumeric characters, '-' or '.'. Name cannot be empty.

Configuration:

Variable Name	Variable Value	Action
enemies	aliens	Edit Delete
lives	3	Edit Delete

Name

Value

Add

Variable key must be unique. Variable key and value cannot be empty.

Edit YAML file

OK

Cancel

In this example, configure the variables `enemies` and `lives` to pass the parameters `aliens` and `3` respectively.

YAML format

```
1 data:
2   enemies: aliens
3   lives: '3'
4 metadata:
5   name: test-config
6   namespace: default
7
```

* Configuration must be in YAML format.

OK

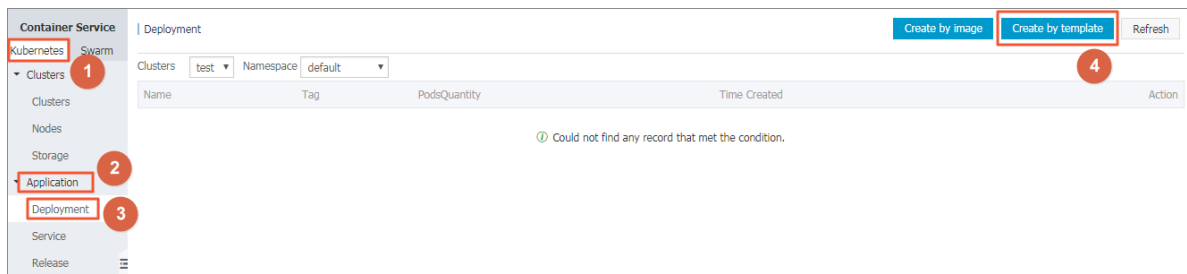
Cancel

5. You can view the config map test-config on the Config Maps page after clicking **OK**.

Config Maps				Refresh	Create
Cluster: test					
Config Map Name	Namespace	Time Created		Operation	
test	default	2018-02-09 03:30:31		Delete	Modify
test-config	default	2018-02-09 05:56:47		Delete	Modify

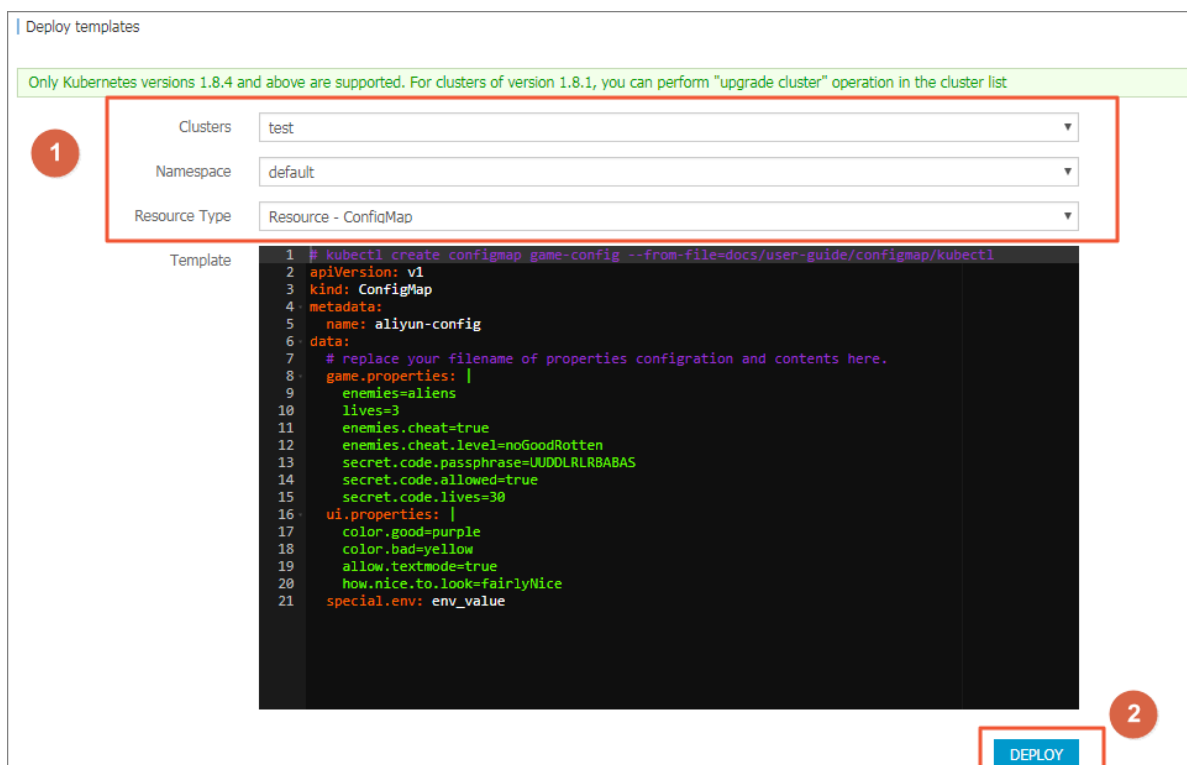
Create a config map by using a template

1. Log on to the [Container Service console](#).
2. Under Kubernetes, click **Application** > **Deployment** in the left-side navigation pane.
3. Click **Create by template** in the upper-right corner.



4. On the Deploy templates page, complete the settings and then click **DEPLOY**.

- **Clusters:** Select the cluster in which the config map is to be created.
- **Namespace:** Select the namespace to which the config map belongs. Config map is a Kubernetes resource object that must be applied to the namespace.
- **Resource Type:** You can write your own config map based on the Kubernetes YAML syntax rules, or select the sample template **resource-ConfigMap**. In the sample template, the config map is named as aliyun-config and includes two variable files `game.properties` and `ui.properties`. You can make modifications based on the sample template. Then, click **DEPLOY**.



5. After the successful deployment, you can view the config map aliyun-config on the Config Maps page.

Config Maps				Refresh	Create
Cluster: test					
Config Map Name	Namespace	Time Created		Operation	
aliyun-config	default	04/24/2018,15:41:32		Delete	Modify

1.8.2 Use a config map in a pod

You can use a config map in a pod in the following scenarios:

- Use a config map to define the pod environment variables.
- Use a config map to configure command line parameters.
- Use a config map in data volumes.

For more information, see [Configure a pod to use a ConfigMap](#).

Limits

To use a config map in a pod, make sure the config map and the pod are in the same cluster and namespace.

Create a config map

In this example, create a config map special-config, which includes two key-value pairs:

`SPECIAL_LEVEL: very` and `SPECIAL_TYPE: charm`.

Create a config map by using an orchestration template

1. Log on to the [Container Service console](#).
2. Under Kubernetes, click **Application** > **Deployment**. Click **Create by template** in the upper-right corner.
3. Select the cluster and namespace from the Clusters and Namespace drop-down lists. Select a sample template or Custom from the Resource Type drop-down list. Click **DEPLOY**.

You can use the following YAML sample template to create a config map.

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: special-config
  namespace: default
data:
  SPECIAL_LEVEL: very
  SPECIAL_TYPE: charm
```

Create a config map on Config Maps page

1. Log on to the [Container Service console](#).

2. Under Kubernetes, click **Application** > **Config Maps** in the left-side navigation pane.
3. Select the cluster and namespace from the Clusters and Namespace drop-down lists. Click **Create** in the upper-right corner.
4. Enter the Config Map Name. Enter the Variable Name and the Variable Value. Then, click **Add** on the right. Click **OK** after completing the configurations.

Container Service

Kubernetes

Swarm

Config Map

Overview

Clusters

Namespaces

Config Maps

Secrets

Services

Ingress

Load Balancers

Network Policies

Storage

Application

Deployment

Pods

Service

Ingress

Release

Config Maps

Store

App Catalog

Clusters

Namespace

* Config Map Name:

special-config

Configuration:

Variable Name	Variable Value	Action
SPECIAL_LEVEL	very	Edit Delete
SPECIAL_TYPE	charm	Edit Delete
<input type="text"/>	<input type="text"/>	<input type="button" value="Add"/>

Name must consist of lowercase alphanumeric characters, '-' or '.'. Name cannot be empty.

Variable key must be unique. Variable key and value cannot be empty.

[Edit YAML file](#)

OK

Cancel

Use a config map to define pod environment variables

Use config map data to define pod environment variables

1. Log on to the [Container Service console](#).
2. Under Kubernetes, click **Application > Deployment**. Click **Create by template** in the upper-right corner.
3. Select the cluster and namespace from the Clusters and Namespace drop-down lists. Select a sample template or Custom from the Resource Type drop-down list. Click **DEPLOY**.

You can define the environment variables in a pod. Use `valueFrom` to reference the value of `SPECIAL_LEVEL` to define the pod environment variables.

See the following orchestration example:

```
apiVersion: v1
kind: Pod
metadata:
  name: config-pod-1
spec:
  containers:
    - name: test-container
      image: busybox
      command: [ "/bin/sh", "-c", "env" ]
      env:
        - name: SPECIAL_LEVEL_KEY
          valueFrom:                                ##Use valueFrom to
specify env to reference the value of the config map.
          configMapKeyRef:
```

```

        name: special-config          ##The referenced
config map name.
        key: SPECIAL_LEVEL           ##The referenced
config map key.
    restartPolicy: Never

```

Similarly, to define the values of multiple config maps to the environment variable values of the pod, add multiple env parameters in the pod.

Configure all key-value pairs of a config map to pod environment variables

1. Log on to the [Container Service console](#).
2. Under Kubernetes, click **Application > Deployment** Click **Create by template** in the upper-right corner.
3. Select the cluster and namespace from the Clusters and Namespace drop-down lists. Select a sample template or Custom from the Resource Type drop-down list. Click **DEPLOY**.

To configure all the key-value pairs of a config map to the environment variables of a pod, use the envFrom parameter. The key in a config map becomes the environment variable name in the pod.

See the following orchestration example:

```

apiVersion: v1
kind: Pod
metadata:
  name: config-pod-2
spec:
  containers:
    - name: test-container
      image: busybox
      command: [ "/bin/sh", "-c", "env" ]
      envFrom:          ##Reference all the key-value pairs
in the config map special-config.
    - configMapRef:
        name: special-config
      restartPolicy: Never

```

Use a config map to configure command line parameters

1. Log on to the [Container Service console](#).
2. Under Kubernetes, click **Application > Deployment** Click **Create by template** in the upper-right corner.
3. Select the cluster and namespace from the Clusters and Namespace drop-down lists. Select a sample template or Custom from the Resource Type drop-down list. Click **DEPLOY**.

You can use the config map to configure the commands or parameter values in the container by using the environment variable replacement syntax `$(VAR_NAME)`.

See the following orchestration example:

```
apiVersion: v1
kind: Pod
metadata:
  name: config-pod-3
spec:
  containers:
    - name: test-container
      image: busybox
      command: [ "/bin/sh", "-c", "echo $(SPECIAL_LEVEL_KEY) $(SPECIAL_TYPE_KEY)" ]
      env:
        - name: SPECIAL_LEVEL_KEY
          valueFrom:
            configMapKeyRef:
              name: special-config
              key: SPECIAL_LEVEL
        - name: SPECIAL_TYPE_KEY
          valueFrom:
            configMapKeyRef:
              name: special-config
              key: SPECIAL_TYPE
      restartPolicy: Never
```

The output after running the pod is as follows:

```
very charm
```

Use a config map in data volumes

1. Log on to the [Container Service console](#).
2. Under the Kubernetes menu, click **Application Deployment** in the left-side navigation pane. Click **Create by template** in the upper-right corner.
3. Select the cluster and namespace from the Clusters and Namespace drop-down lists. Select a sample template or Custom from the Resource Type drop-down list. Click **DEPLOY**.

You can also use a config map in data volumes. Specifying the config map name under volumes stores the key-value pair data to the mountPath directory (*/etc/config* in this example). It finally generates a configuration file with key as the file name and values as the contents of the file.

Then, the configuration file with key as the name and value as the contents is generated.

```
apiVersion: v1
kind: Pod
metadata:
  name: config-pod-4
spec:
  containers:
    - name: test-container
      image: busybox
```

```

    command: [ "/bin/sh", "-c", "ls /etc/config/" ]    ##List the
    file names under this directory.
    volumeMounts:
      - name: config-volume
        mountPath: /etc/config
    volumes:
      - name: config-volume
        configMap:
          name: special-config
    restartPolicy: Never

```

Keys of the config map are output after running the pod.

```

SPECIAL_TYPE
SPECIAL_LEVEL

```

1.8.3 Update a config map

You can modify the configurations of a config map.

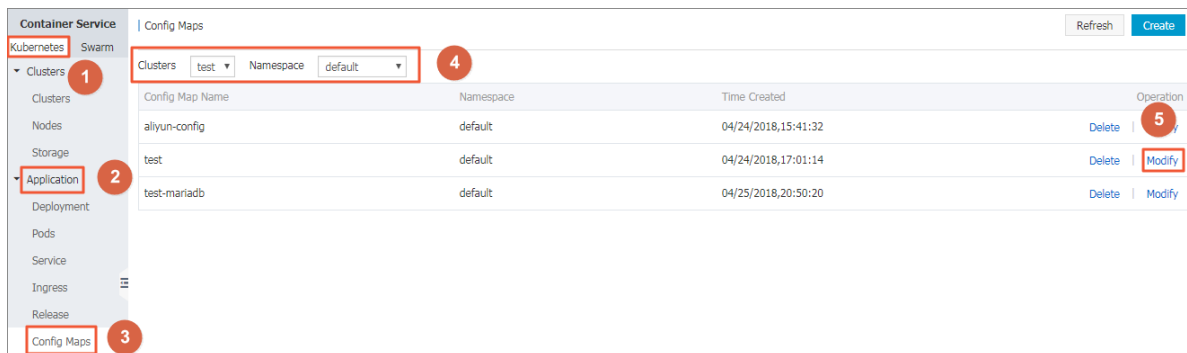


Note:

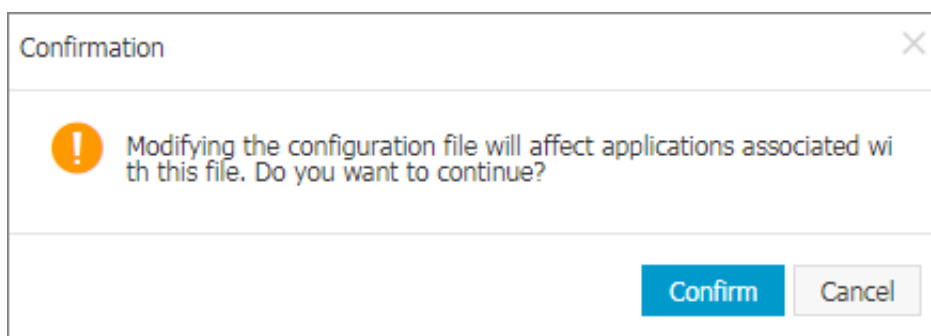
Updating a config map affects applications that use this config map.

Update a config map on Config Maps page

1. Log on to the [Container Service console](#).
2. Under Kubernetes, click **Application** > **Config Maps** in the left-side navigation pane.
3. Select the cluster and namespace from the Clusters and Namespace drop-down lists. Click **Modify** at the right of the config map.

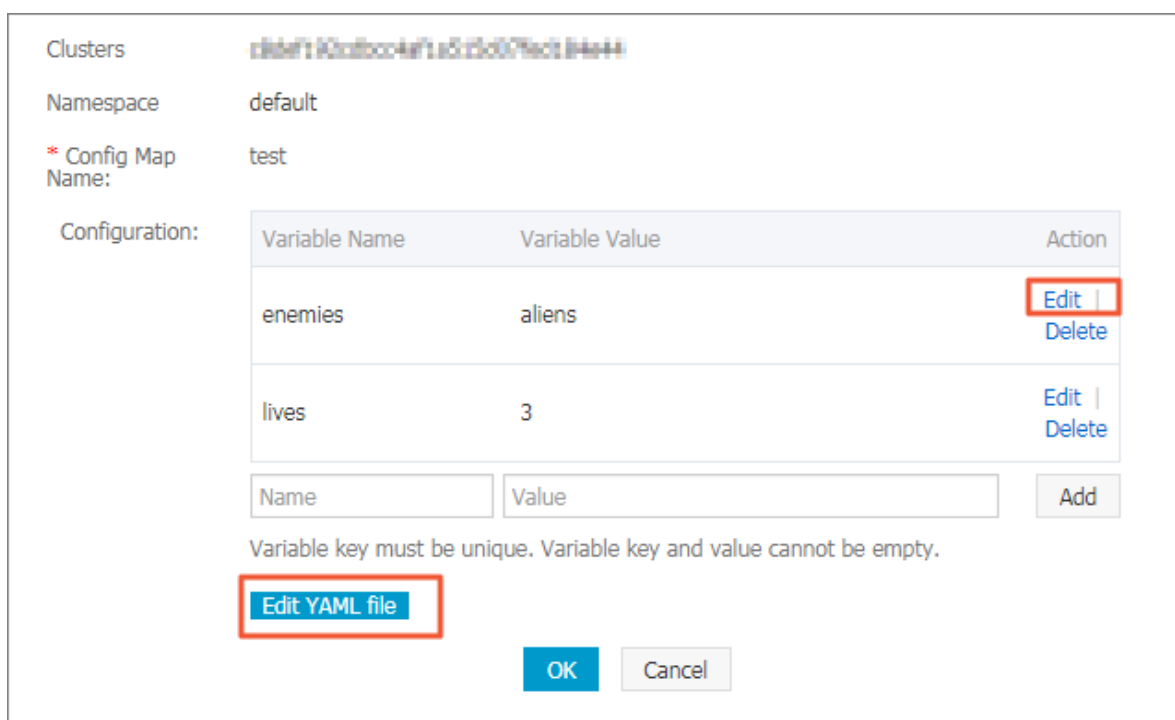


4. Click **Confirm** in the displayed dialog box.



5. Modify the configurations.

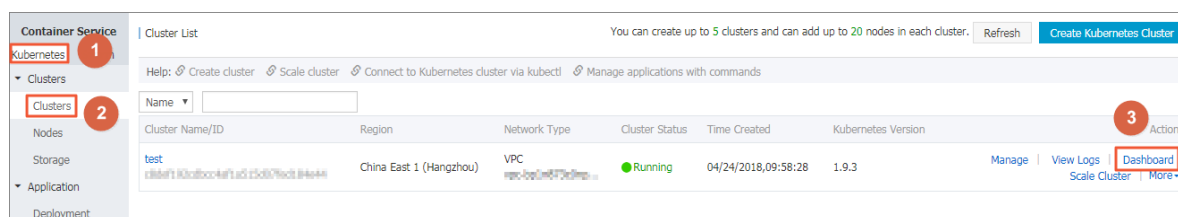
- Click **Edit** on the right of the configuration you want to modify. Update the configuration and then click **Save**.
- You can also click **Edit YAML file**. Click **OK** after making the modifications.



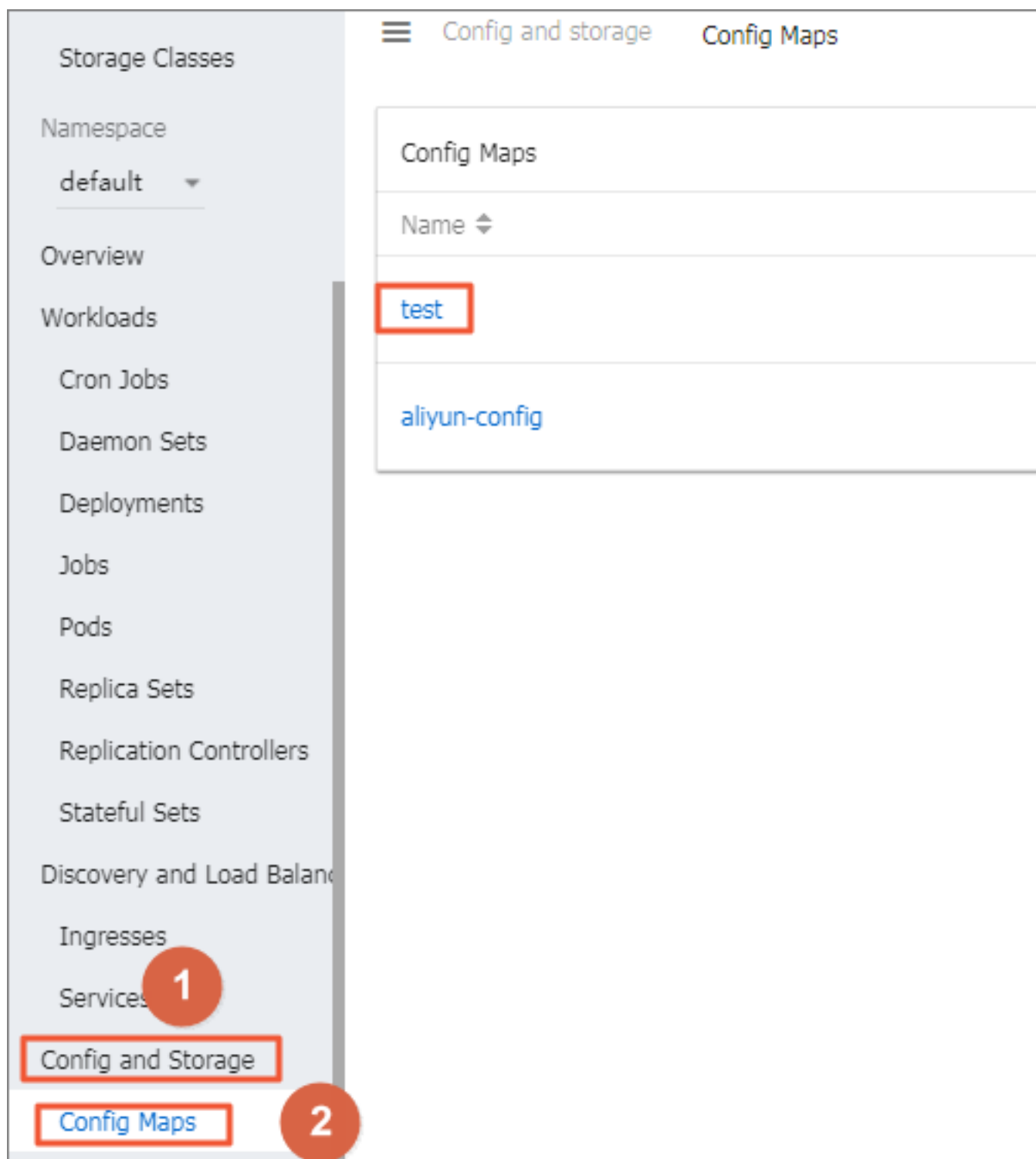
6. After modifying the configurations, click **OK**.

Update a config map in Kubernetes dashboard

1. Log on to the [Container Service console](#).
2. Under Kubernetes, click **Clusters** in the left-side navigation pane.
3. Click **Dashboard** at the right of the cluster.



4. Under Kubernetes, select a namespace, click **Config and Storage > Secrets** in the left-side navigation pane. Select the target secret and click **Actions > View/edit YAML**.



5. The Edit a Secret dialog box appears. Modify the configurations and then click **UPDATE**.

Edit a Config Map

```
1 {
2   "kind": "ConfigMap",
3   "apiVersion": "v1",
4   "metadata": {
5     "name": "test",
6     "namespace": "default",
7     "selfLink": "/api/v1/namespaces/default/configmaps/test",
8     "uid": "0a826463-479e-11e8-a84c-00163e101791",
9     "resourceVersion": "52788",
10    "creationTimestamp": "2018-04-24T09:01:14Z"
11  },
12  "data": {
13    "enemies": "aliens",
14    "lives": "3"
15  }
16 }
```

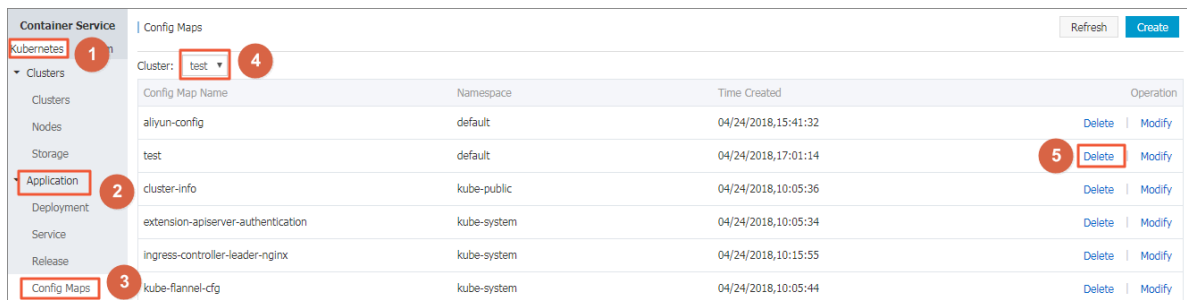
CANCEL COPY UPDATE

1.8.4 Delete a config map

You can delete a config map that is no longer in use.

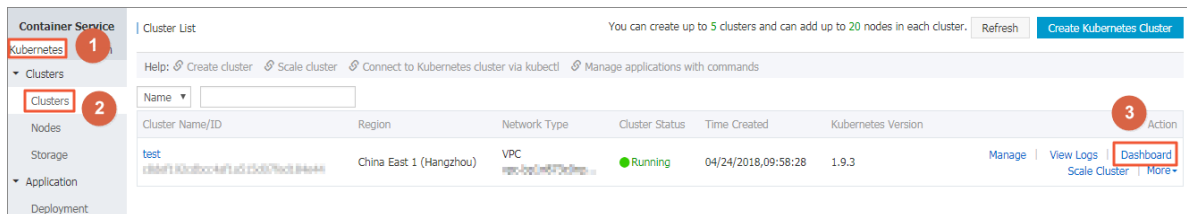
Delete a config map on Config Maps page

1. Log on to the [Container Service console](#).
2. Under Kubernetes, click **Application** > **Config Maps** in the left-side navigation pane.
3. Select the target cluster from the Cluster drop-down list. Click **Delete** at the right of the config map.

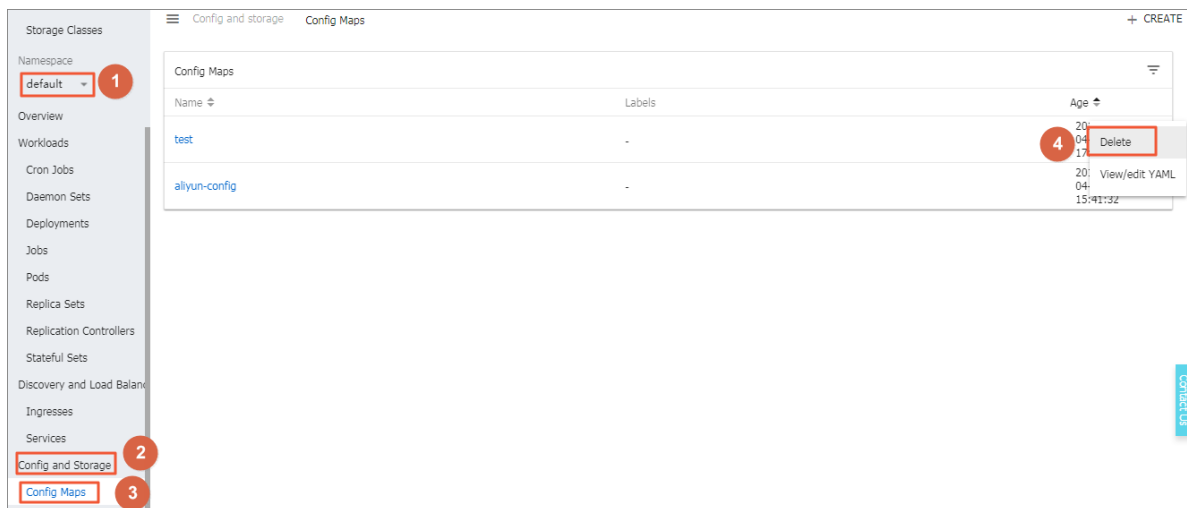


Delete a config map in Kubernetes dashboard

1. Log on to the [Container Service console](#).
2. Under Kubernetes, click **Clusters** in the left-side navigation pane.
3. Click **Clusters** in the left-side navigation pane, select the target cluster, and click **Dashboard** on the right.



4. Under Kubernetes, select a namespace, click **Config and Storage > Secrets** in the left-side navigation pane. Click the actions button on the right and click **Delete** in the drop-down list.



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

1.8.5 Create a secret

Prerequisites

You have created a Kubernetes cluster. For more information, see [Create a Kubernetes cluster](#).

Context

We recommend that you use secrets for sensitive configurations in Kubernetes clusters, such as passwords and certificates.

Secrets have many types. For example:

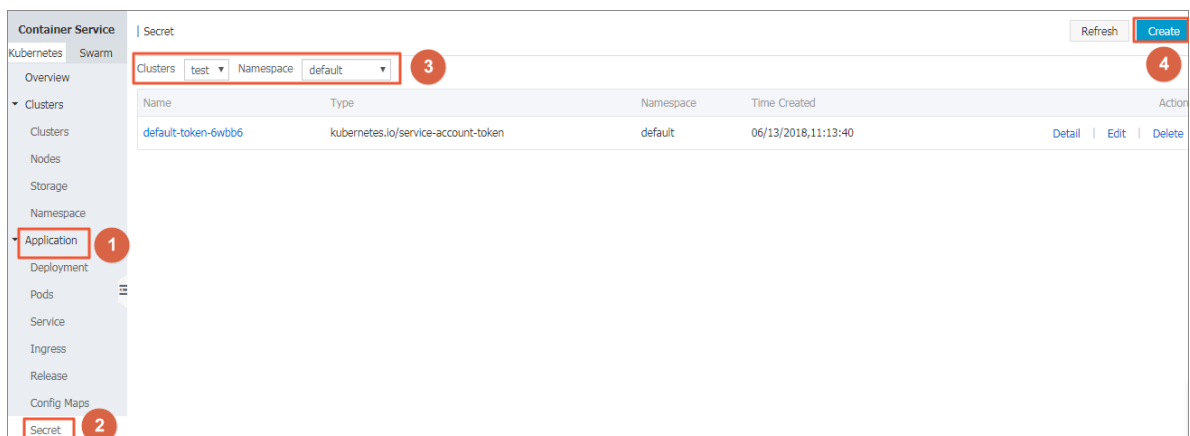
- **Service Account:** Automatically created by Kubernetes, which is used to access Kubernetes APIs and is automatically mounted to the pod directory `/run/secrets/kubernetes.io/serviceaccount`.
- **Opaque:** Secret in the base64 encoding format, which is used to store sensitive information such as passwords and certificates.

By default, you can only create secrets of the Opaque type in the Container Service console. Opaque data is of the map type, which requires the value to be in the base64 encoding format. Alibaba Cloud Container Service supports creating secrets with one click and automatically encoding the clear data to base64 format.

You can also create secrets manually by using command lines. For more information, see [Kubernetes secrets](#).

Procedure

1. Log on to the [Container Service console](#).
2. Under Kubernetes, click **Application > Secret** in the left-side navigation pane.
3. Select the cluster and namespace from the Clusters and Namespace drop-down lists. Click **Create** in the upper-right corner.



4. Complete the configurations to create a secret.



Note:

To enter the clear data of the secret, select the **Encode data values using Base64** check box.

Namespace: default

* Name: (1)
Name must consist of lowercase alphanumeric characters, '-' or '.'. Name cannot be empty.

* Data:

Name	Value
<input type="text" value="username"/> (2)	<input type="text" value="admin"/>
<input type="text" value="password"/> (3)	<input type="text" value="1f2d1e2e67df"/>

Names can only contain numbers, letters, "_", "-" and ".".

☒ Encode data values using Base64

1. Name: Enter the secret name, which must be 1–253 characters long, and can only contain lowercase letters, numbers, hyphens (-), and dots (.).
2. Configure the secret data. Click the **add** icon next to Name and enter the name and value of the secret, namely, the key-value pair. In this example, the secret contains two values: `username:admin` and `password:1f2d1e2e67df`.
3. Click **OK**.
5. The Secret page appears. You can view the created secret in the secret list.

Secret					<input type="button" value="Refresh"/>	<input type="button" value="Create"/>
Clusters	test	Namespace	default			
Name	Type	Namespace	Time Created	Action		
account	Opaque	default	06/13/2018,11:39:06	Detail	Edit	Delete
default-token-6wbb6	kubernetes.io/service-account-token	default	06/13/2018,11:13:40	Detail	Edit	Delete

1.8.6 View secret details

You can view the details of a created secret in the Container Service console.

Prerequisites

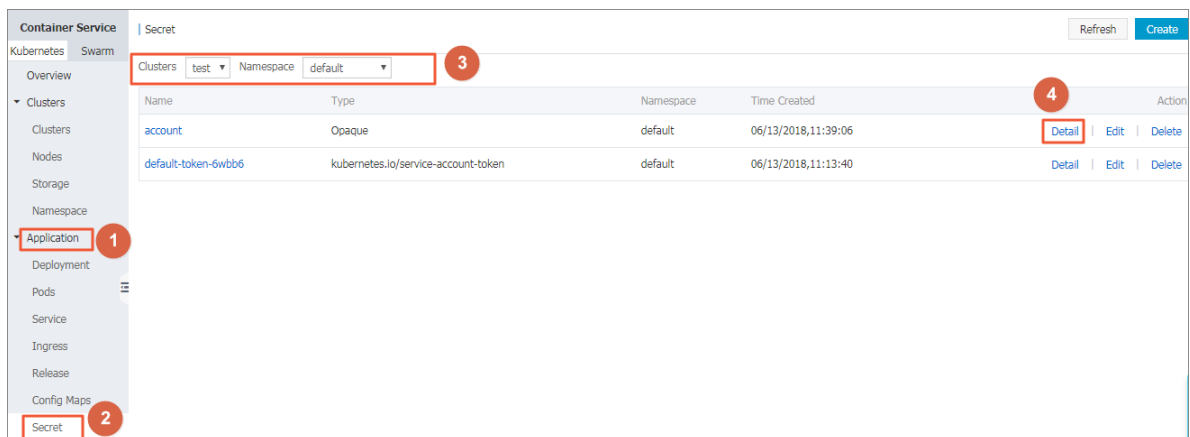
- You have created a Kubernetes cluster. For more information, see [Create a Kubernetes cluster](#).

- You have created a secret. For more information, see [Create a secret](#).

Context

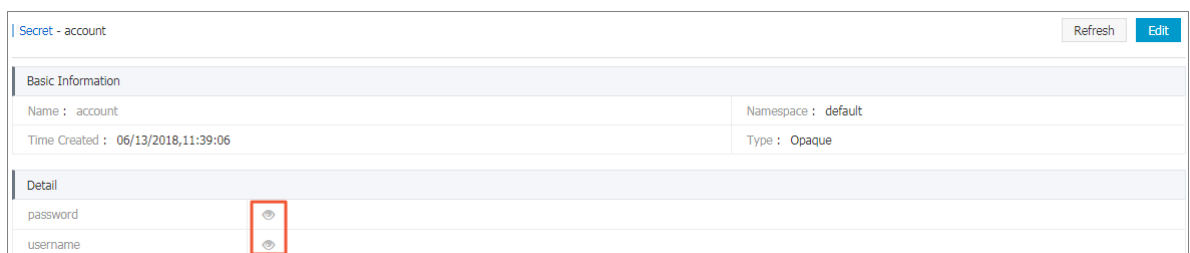
Procedure

- Log on to the [Container Service console](#).
- Under Kubernetes, click **Application** > **Secret** in the left-side navigation pane.
- Select the cluster and namespace from the Clusters and Namespace drop-down lists. Click **Detail** at the right of the secret.



- You can view the basic information of the secret, and the data that the secret contains.

Click the icon at the right of the data name under Detail to view the clear data.



1.8.7 Update a secret

You can update an existing secret directly in the Container Service console.

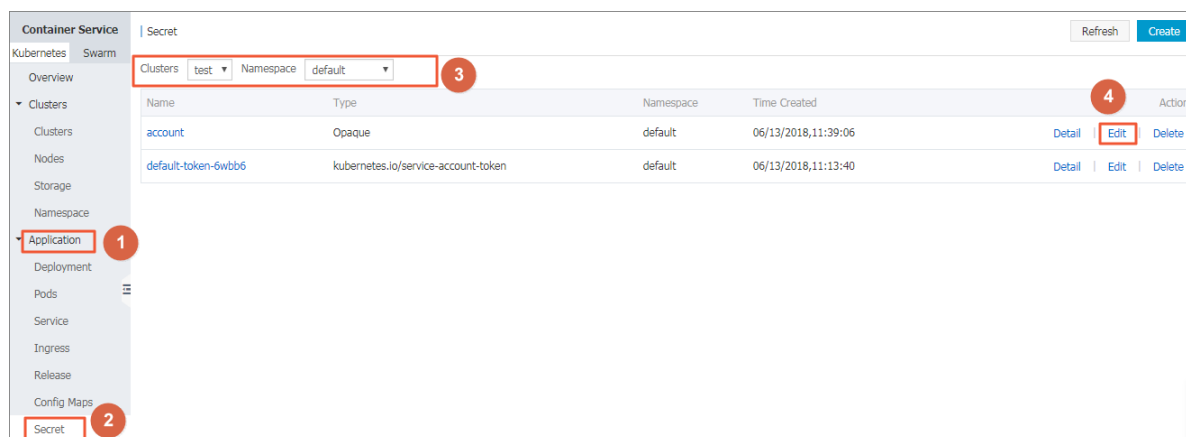
Prerequisites

- You have created a Kubernetes cluster. For more information, see [Create a Kubernetes cluster](#).
- You have created a secret. For more information, see [Create a secret](#).

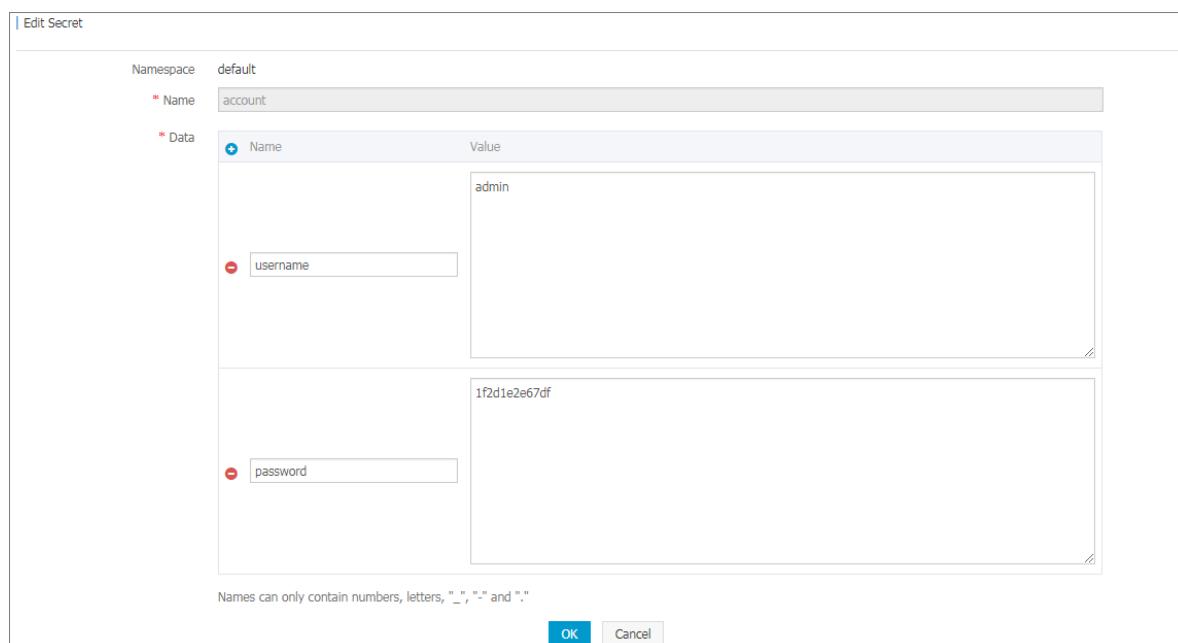
Context

Procedure

1. Log on to the [Container Service console](#).
2. Under Kubernetes, click **Application** > **Secret** in the left-side navigation pane.
3. Select the cluster and namespace from the Clusters and Namespace drop-down lists. Click **Edit** at the right of the secret.



4. Update the secret data on the Edit Secret page.



5. Click **OK**.

1.8.8 Delete a secret

You can delete an existing secret directly in the Container Service console.

Prerequisites

- You have created a Kubernetes cluster. For more information, see [Create a Kubernetes cluster](#).

- You have created a secret. For more information, see [Create a secret](#).

Context

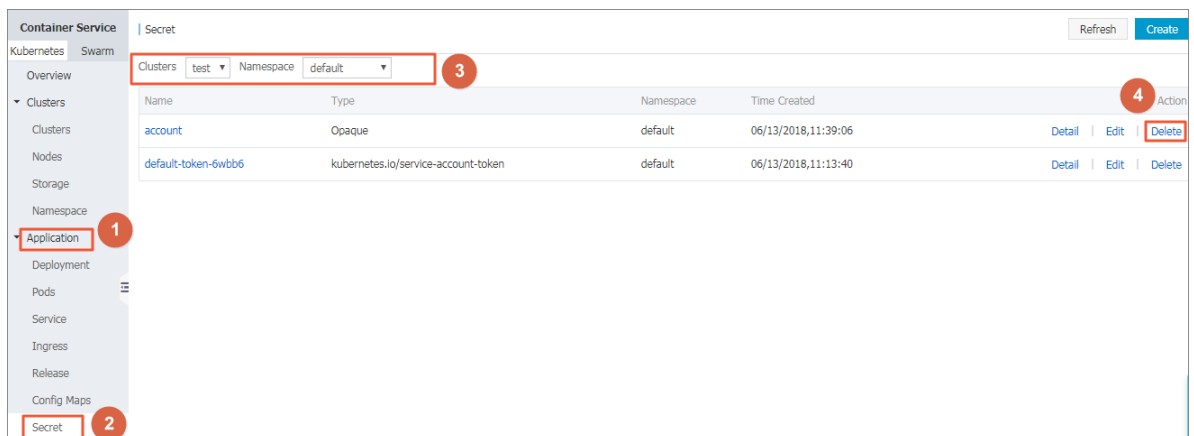


Note:

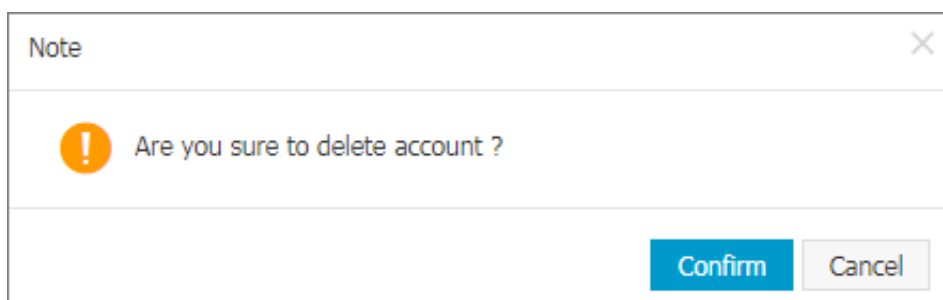
Do not delete the secret generated when the cluster is created.

Procedure

1. Log on to the [Container Service console](#).
2. Under Kubernetes, click **Application** > **Secret** in the left-side navigation pane.
3. Select the cluster and namespace from the Clusters and Namespace drop-down lists. Click **Delete** at the right of the secret.



4. Click Confirm in the displayed dialog box to delete the secret.



1.9 Storage

1.9.1 Overview

Container Service supports automatically binding Kubernetes pods to Alibaba Cloud cloud disks, NAS, and Object Storage Service (OSS).

Currently, static storage volumes and dynamic storage volumes are supported. See the following table for how each type of data volumes supports the static data volumes and dynamic data volumes.

Alibaba Cloud storage	Static data volume	Dynamic data volume
Alibaba Cloud cloud disk	You can use the cloud disk static storage volumes by: <ul style="list-style-type: none">• Using the volume method.• Using PV/PVC.	Supported.
Alibaba Cloud NAS	You can use the NAS static storage volumes by: <ul style="list-style-type: none">• Using flexvolume plug-in.<ul style="list-style-type: none">— Using the volume method.— Using PV/PVC.• Using NFS drive of Kubernetes.	Supported.
Alibaba Cloud OSS	You can use the OSS static storage volumes by: <ul style="list-style-type: none">• Using the volume method.• Using PV/PVC.	Not supported.

1.9.2 Install the plug-in

Deploy the Alibaba Cloud Kubernetes storage plug-in by using the following yaml configurations.

**Note:**

If your Kubernetes cluster is created before February 6th, 2018, install the Alibaba Cloud Kubernetes storage plug-in before using the data volumes. If your Kubernetes cluster is created after February 6th, 2018, you can directly use the data volumes without installing the Alibaba Cloud Kubernetes storage plug-in.

Limits

Currently, CentOS 7 operating system is supported.

Instructions

- Disable the `--enable-controller-attach-detach` option by using kubelet if you use the flexvolume. By default, Alibaba Cloud Kubernetes clusters have disabled this option.
- Deploy flexvolume in the kube-system user space.

Verify that the installation is complete

On the master node:

- Run the `kubectl get pod -n kube-system | grep flexvolume` command . Output is the list of running pods (number of nodes) .
- Run the `kubectl get pod -n kube-system | grep alicloud-disk-controller` command. Output is the list of running pods.

Installation example

Install flexvolume

```
apiVersion: apps/v1 # for versions before 1.8.0 use extensions/v1beta1
kind: DaemonSet
metadata:
  name: flexvolume
  namespace: kube-system
  labels:
    k8s-volume: flexvolume
spec:
  selector:
    matchLabels:
      name: acs-flexvolume
  template:
    metadata:
      labels:
        name: acs-flexvolume
    spec:
      hostPID: true
      hostNetwork: true
      tolerations:
        - key: node-role.kubernetes.io/master
          operator: Exists
          effect: NoSchedule
      containers:
        - name: acs-flexvolume
          image: registry.cn-hangzhou.aliyuncs.com/acs/flexvolume:v1.9.7-42e8198
          imagePullPolicy: Always
          securityContext:
            privileged: true
          env:
            - name: ACS_DISK
```



```

        value: "true"
      - name: ACS_NAS
        value: "true"
      - name: ACS_OSS
        value: "true"
    resources:
      limits:
        memory: 200Mi
      requests:
        cpu: 100m
        memory: 200Mi
    volumeMounts:
      - name: usrdir
        mountPath: /host/usr/
      - name: etcdDir
        mountPath: /host/etc/
      - name: logdir
        mountPath: /var/log/alicloud/
    volumes:
      - name: usrdir
        hostPath:
          path: /usr/
      - name: etcdDir
        hostPath:
          path: /etc/
      - name: logdir
        hostPath:
          path: /var/log/alicloud/

```

Install Disk provisioner

```

---
kind: StorageClass
apiVersion: storage.k8s.io/v1beta1
metadata:
  name: alicloud-disk-common
provisioner: alicloud/disk
parameters:
  type: cloud
---
kind: StorageClass
apiVersion: storage.k8s.io/v1beta1
metadata:
  name: alicloud-disk-efficiency
provisioner: alicloud/disk
parameters:
  type: cloud_efficiency
---
kind: StorageClass
apiVersion: storage.k8s.io/v1beta1
metadata:
  name: alicloud-disk-ssd
provisioner: alicloud/disk
parameters:
  type: cloud_ssd
---
kind: StorageClass
apiVersion: storage.k8s.io/v1beta1
metadata:
  name: alicloud-disk-available
provisioner: alicloud/disk

```

```

parameters:
  type: available
---
kind: ClusterRole
apiVersion: rbac.authorization.k8s.io/v1beta1
metadata:
  name: alicloud-disk-controller-runner
rules:
  - apiGroups: [""]
    resources: ["persistentvolumes"]
    verbs: ["get", "list", "watch", "create", "delete"]
  - apiGroups: [""]
    resources: ["persistentvolumeclaims"]
    verbs: ["get", "list", "watch", "update"]
  - apiGroups: ["storage.k8s.io"]
    resources: ["storageclasses"]
    verbs: ["get", "list", "watch"]
  - apiGroups: [""]
    resources: ["events"]
    verbs: ["list", "watch", "create", "update", "patch"]
---
apiVersion: v1
kind: ServiceAccount
metadata:
  name: alicloud-disk-controller
  namespace: kube-system
---
kind: ClusterRoleBinding
apiVersion: rbac.authorization.k8s.io/v1beta1
metadata:
  name: run-alicloud-disk-controller
subjects:
  - kind: ServiceAccount
    name: alicloud-disk-controller
    namespace: kube-system
roleRef:
  kind: ClusterRole
  name: alicloud-disk-controller-runner
  apiGroup: rbac.authorization.k8s.io
---
kind: Deployment
apiVersion: extensions/v1beta1
metadata:
  name: alicloud-disk-controller
  namespace: kube-system
spec:
  replicas: 1
  strategy:
    type: Recreate
  template:
    metadata:
      labels:
        app: alicloud-disk-controller
    spec:
      tolerations:
        - effect: NoSchedule
          operator: Exists
          key: node-role.kubernetes.io/master
        - effect: NoSchedule
          operator: Exists
          key: node.cloudprovider.kubernetes.io/uninitialized
      nodeSelector:

```

```
node-role.kubernetes.io/master: ""
serviceAccount: alicloud-disk-controller
containers:
- name: alicloud-disk-controller
  image: registry.cn-hangzhou.aliyuncs.com/acs/alibabacloud-disk-controller:v1.9.3-ed710ce
  volumeMounts:
  - name: cloud-config
    mountPath: /etc/kubernetes/
  - name: logdir
    mountPath: /var/log/alibabacloud/
volumes:
- name: cloud-config
  hostPath:
    path: /etc/kubernetes/
- name: logdir
  hostPath:
    path: /var/log/alibabacloud/
```

1.9.3 Use Alibaba Cloud cloud disks

You can use the Alibaba Cloud cloud disk storage volumes in Alibaba Cloud Container Service Kubernetes clusters.

Currently, Alibaba Cloud cloud disk provides the following two Kubernetes mount methods:

- [Static storage volumes](#)

You can use the cloud disk static storage volumes by:

- [Using the volume method](#)
- [Using PV/PVC](#)
- [Dynamic storage volumes](#)

**Note:**

The following requirements are imposed on the created cloud disk capacity:

- Basic cloud disk: Minimum 5Gi
- Ultra cloud disk: Minimum 20Gi
- SSD cloud disk: Minimum 20Gi

Static storage volumes

You can use Alibaba Cloud cloud disk storage volumes by using the volume method or PV/PVC.

Prerequisites

Before using cloud disk data volumes, you must create cloud disks in the Elastic Compute Service (ECS) console. For how to create cloud disks, see [Create a cloud disk](#).

Instructions

- The cloud disk is not a shared storage and can only be mounted by one pod at the same time.
- Apply for a cloud disk and obtain the disk ID before using cloud disk storage volumes. See [Create a cloud disk](#).
- `volumeId`: The disk ID of the mounted cloud disk, which must be the same as `volumeName` and PV Name.
- Only the cluster node that is in the same zone as the cloud disk can mount the cloud disk.

Use volume method

Use the `disk-deploy.yaml` file to create the pod.

```
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: nginx-disk-deploy
spec:
  replicas: 1
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
        - name: nginx-flexvolume-disk
          image: nginx
          volumeMounts:
            - name: "d-bp1j17ifxfasvts3tf40"
              mountPath: "/data"
      volumes:
        - name: "d-bp1j17ifxfasvts3tf40"
          flexVolume:
            driver: "alicloud/disk"
            fsType: "ext4"
            options:
              volumeId: "d-bp1j17ifxfasvts3tf40"
```

Use PV/PVC

Step 1 Create a cloud disk type PV

You can create the cloud disk type PV in the Container Service console or by using the yaml file.

Create PV by using yaml file

Use the `disk-pv.yaml` file to create the PV.



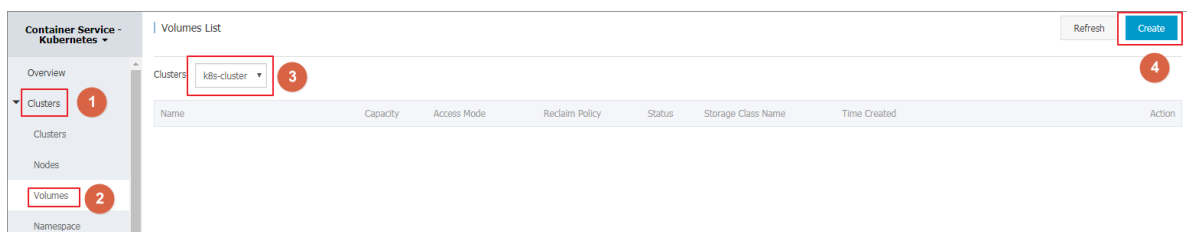
Note:

The PV name must be the same as the Alibaba Cloud cloud disk ID.

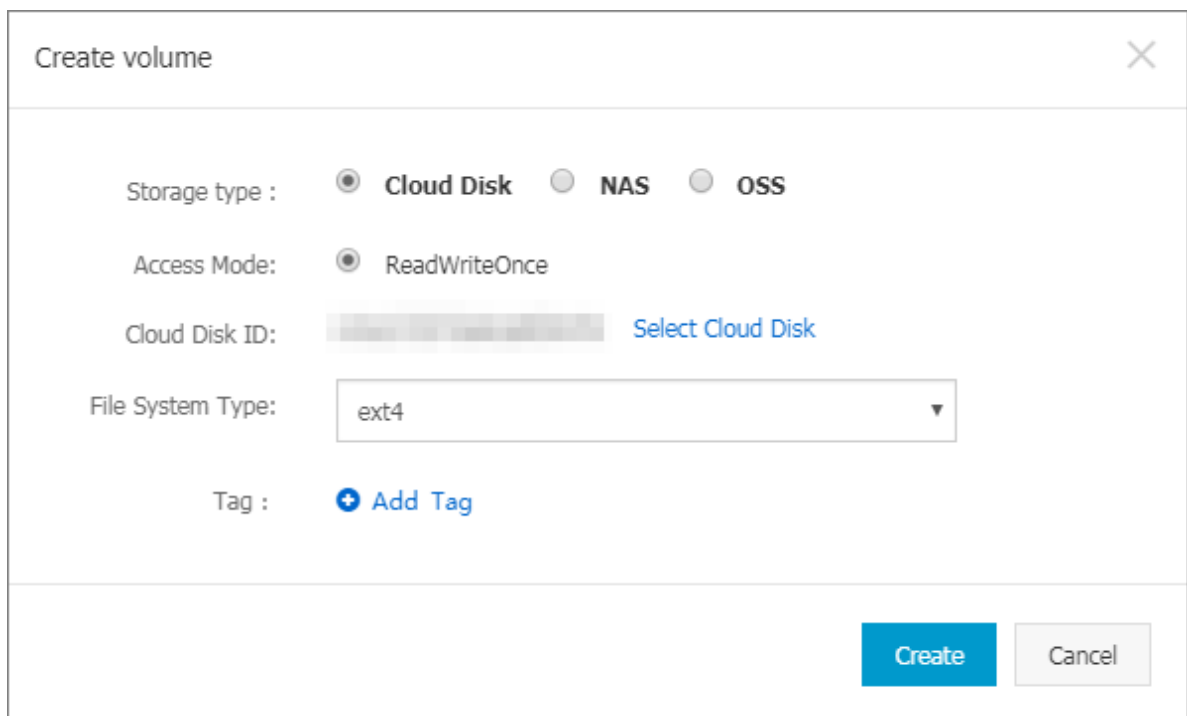
```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: d-bp1jl7ifxfasvts3tf40
  labels:
    failure-domain.beta.kubernetes.io/zone: cn-hangzhou-b
    failure-domain.beta.kubernetes.io/region: cn-hangzhou
spec:
  capacity:
    storage: 20Gi
  storageClassName: disk
  accessModes:
    - ReadWriteOnce
  flexVolume:
    driver: "alicloud/disk"
    fsType: "ext4"
    options:
      volumeId: "d-bp1jl7ifxfasvts3tf40"
```

Create cloud disk data volumes on the Container Service console

1. Log on to the [Container Service console](#).
2. Under Kubernetes, click **Clusters** > **Volumes** in the left-side navigation pane.
3. Select the cluster from the Clusters drop-down list and then click **Create** in the upper-right corner.



4. The Create Data Volume dialog box appears. Configure the data volume parameters.
 - **Type:** **cloud disk** in the example.
 - **Access Mode:** ReadWriteOnce by default.
 - **Cloud Disk ID:** Select the cloud disk to be mounted and is in the same region and zone as the cluster.
 - **File System Type:** You can select the data type in which data is stored to the cloud disk. The supported types include ext4, ext3, xfs, and vfat ext4 is selected by default.
 - **Tag:** Click Add Tag to add tags for this data volume.



The 'Create volume' dialog box contains the following fields and controls:

- Storage type :** Three radio buttons for **Cloud Disk** (selected), **NAS**, and **OSS**.
- Access Mode:** One radio button for **ReadWriteOnce**.
- Cloud Disk ID:** A text input field with a blurred value and a blue link **Select Cloud Disk** to its right.
- File System Type:** A dropdown menu currently showing **ext4**.
- Tag :** A blue plus icon followed by the text **Add Tag**.
- Buttons:** A blue **Create** button and a grey **Cancel** button at the bottom right.

5. Click **Create** after the configurations.

Step 2 Create PVC

Use the `disk-pvc.yaml` file to create the PVC.

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: pvc-disk
spec:
  accessModes:
    - ReadWriteOnce
  storageClassName: disk
  resources:
    requests:
      storage: 20Gi
```

Step 3 Create a pod

Use the `disk-pod.yaml` file to create the pod.

```
apiVersion: v1
kind: Pod
metadata:
  name: "flexvolume-alicloud-example"
spec:
  containers:
    - name: "nginx"
      image: "nginx"
      volumeMounts:
        - name: pvc-disk
          mountPath: "/data"
  volumes:
```

```
- name: pvc-disk
  persistentVolumeClaim:
    claimName: pvc-disk
```

Dynamic storage volumes

Dynamic storage volumes require you to manually create a Storage Class and specify the target type of cloud disk in the PVC by storage Class Name.

Create a StorageClass

```
kind: StorageClass
apiVersion: storage.k8s.io/v1beta1
metadata:
  name: alicloud-disk-common-hangzhou-b
provisioner: alicloud/disk
parameters:
  type: cloud_ssd
  regionid: cn-hangzhou
  zoneid: cn-hangzhou-b
```

Parameters:

- **provisioner:** Configured as alicloud/disk. The identifier is created by using the Alibaba Cloud Provisioner plug-in.
- **type:** The identifier of a cloud disk type. The supported cloud disk identifier are cloud, cloud_efficiency, cloud_ssd, and available. When this parameter is specifies as available, the system attempts to create an efficient, SSD, or basic cloud disk in turn until a cloud disk is created.
- **regionid:** The region where a cloud disk is to be created.
- **reclaimPolicy:** The cloud disk reclaim policy. The default is Delete. Retain is supported.
- **zoneid:** The zone where a cloud disk is to be created.

Create a service

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: disk-common
spec:
  accessModes:
    - ReadWriteOnce
  storageClassName: alicloud-disk-common-hangzhou-b
  resources:
    requests:
      storage: 20Gi
---
kind: Pod
apiVersion: v1
metadata:
  name: disk-pod-common
spec:
  containers:
```

```
- name: disk-pod
  image: nginx
  volumeMounts:
    - name: disk-pvc
      mountPath: "/mnt"
  restartPolicy: "Never"
  volumes:
    - name: disk-pvc
      persistentVolumeClaim:
        claimName: disk-common
```

Default options

By default, the cluster provides the following StorageClasses, which can be used in a single AZ cluster.

- alicloud-disk-common: Basic cloud disk.
- alicloud-disk-efficiency: High-efficiency cloud disk.
- alicloud-disk-ssd: SSD disk.
- alicloud-disk-available: Provides highly available options. The system first attempts to create a high-efficiency cloud disk. If the corresponding AZ high-efficiency cloud disk resources are sold out, the system tries to create an SSD cloud disk. If the SSD cloud disk is sold out, the system tries to create a basic cloud disk.

Create a multi-instance StatefulSet by using a cloud disk

Create a multi-instance StatefulSet by means of volumeClaimTemplates, which dynamically creates multiple PVCs and PVs, and binds them.

```
apiVersion: v1
kind: Service
metadata:
  name: nginx
  labels:
    app: nginx
spec:
  ports:
    - port: 80
      name: web
  clusterIP: None
  selector:
    app: nginx
---
apiVersion: apps/v1beta2
kind: StatefulSet
metadata:
  name: web
spec:
  selector:
    matchLabels:
      app: nginx
  serviceName: "nginx"
  replicas: 2
```



```
template:
  metadata:
    labels:
      app: nginx
  spec:
    containers:
      - name: nginx
        image: nginx
        ports:
          - containerPort: 80
            name: web
        volumeMounts:
          - name: disk-common
            mountPath: /data
    volumeClaimTemplates:
      - metadata:
          name: disk-common
        spec:
          accessModes: [ "ReadWriteOnce" ]
          storageClassName: "alicloud-disk-common"
          resources:
            requests:
              storage: 10Gi
```

1.9.4 Use Alibaba Cloud NAS

You can use the Alibaba Cloud NAS data volumes in Container Service Kubernetes clusters.

Currently, Alibaba Cloud NAS provides the following two Kubernetes mount methods:

- [Static storage volumes](#)

You can use the static storage volumes by:

- Using the flexvolume plug-in.
 - Using the volume method.
 - Using PV/PVC.
- Using NFS drive of Kubernetes.
- [Dynamic storage volumes](#)

Prerequisites

Before using NAS data volumes, you must create a file system in the NAS console and add the mount point of a Kubernetes cluster in the file system. The created NAS file system and your cluster must be in the same Virtual Private Cloud (VPC).

Static storage volumes

You can use Alibaba Cloud NAS file storage service by using the flexvolume plug-in provided by Alibaba Cloud or the NFS drive of Kubernetes.

Use flexvolume plug-in

Use the flexvolume plug-in and then you can use the Alibaba Cloud NAS data volumes by using the volume method or using PV/PVC.



Note:

- NAS is a shared storage and can provide shared storage service for multiple pods at the same time.
- server: The mount point of the NAS data disk.
- path: The mount directory for connecting to the NAS data volumes. You can mount NAS data volumes to a NAS sub-directory. The system automatically creates the sub-directory if the sub-directory does not exist and mounts the NAS data volumes to the created sub-directory.
- vers: Defines the version number of NFS mount protocol. 3.0 and 4.0 are supported.
- mode: Defines the access permission of the mount directory. The mount permission cannot be configured if you mount the NAS data volumes to the NAS root directory. If the NAS disk contains a huge amount of data, configuring the mode leads to the slow mounting or even the mounting failure.

Using the volume method.

Use the `nas-deploy.yaml` file to create the pod.

```
apiVersion: v1
kind: Pod
metadata:
  name: "flexvolume-nas-example"
spec:
  containers:
    - name: "nginx"
      image: "nginx"
      volumeMounts:
        - name: "nas1"
          mountPath: "/data"
  volumes:
    - name: "nas1"
      flexVolume:
        driver: "alicloud/nas"
        options:
          server: "0cd8b4a576-grs79.cn-hangzhou.nas.aliyuncs.com"
          path: "/k8s"
          vers: "4.0"
```

Using PV/PVC.

Step 1 Create PV

You can create NAS data volumes in the Container Service console or by using the YAML file.

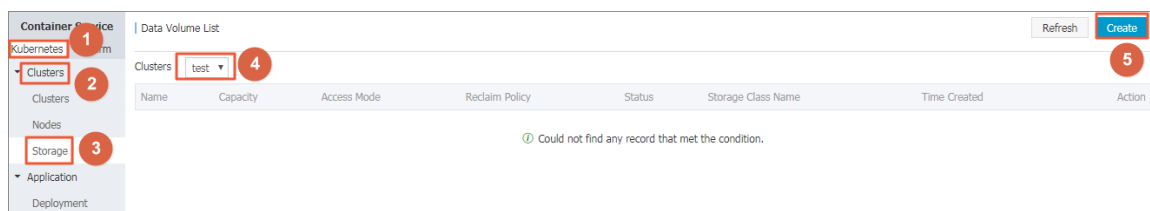
- Create PV by using YAML file

Use the `nas-pv.yaml` file to create the PV.

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: pv-nas
spec:
  capacity:
    storage: 5Gi
  storageClassName: nas
  accessModes:
    - ReadWriteMany
  flexVolume:
    driver: "alicloud/nas"
    options:
      server: "0cd8b4a576-uih75.cn-hangzhou.nas.aliyuncs.com"
      path: "/k8s"
      vers: "4.0"
```

- Create NAS data volumes in Container Service console

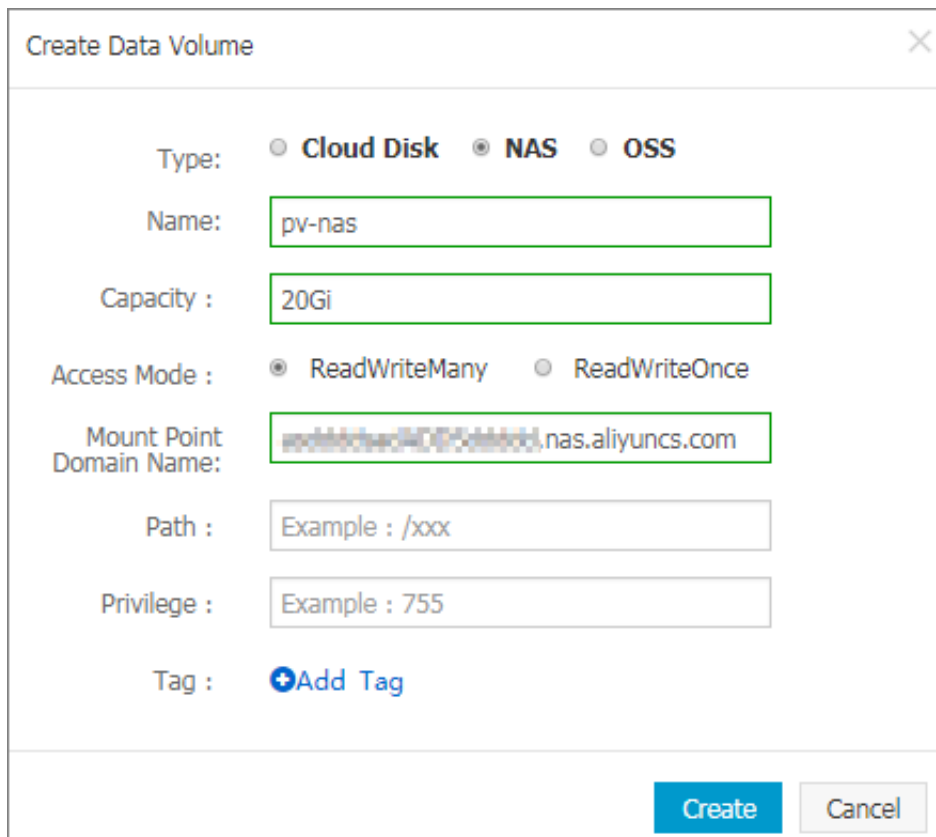
1. Log on to the [Container Service console](#).
2. Under Kubernetes, click **Clusters** > **Volumes** in the left-side navigation pane.
3. Select the cluster from the Clusters drop-down list and then click **Create** in the upper-right corner.



4. The Create Data Volume dialog box appears. Configure the data volume parameters.

- **Type:** Select NAS in this example.
- **Name:** Enter the name of the data volume you are about to create. The data volume name must be unique in the cluster. In this example, enter `pv-nas`.
- **Capacity:** Enter the capacity of the data volume to be created. Make sure the capacity cannot exceed the disk capacity.
- **Access Mode:** `ReadWriteMany` is selected by default.
- **Mount Point Domain Name:** Enter the mount address of the mount point in the NAS file system for the cluster.
- **Path:** The sub-directory under the NAS path, which starts with a forward slash `/`. The data volume is mounted to the specified sub-directory after being created.

- If this sub-directory does not exist in the NAS root directory, the data volume is mounted after the sub-directory is created by default.
- If this field is left empty, the data volume is mounted to the NAS root directory by default.
- **Privilege:** Configure the access permission of the mount directory, such as 755, 644, and 777.
 - You can only configure the privilege when the data volume is mounted to the NAS sub-directory, that is, you cannot configure the privilege if the data volume is mounted to the NAS root directory.
 - If this field is left empty, use the permissions of the NAS files by default.
- **Tag:** Click Add Tag to add tags for this data volume.



The image shows a 'Create Data Volume' dialog box with the following fields and options:

- Type:** Radio buttons for Cloud Disk, **NAS** (selected), and OSS.
- Name:** Text input field containing 'pv-nas'.
- Capacity :** Text input field containing '20Gi'.
- Access Mode :** Radio buttons for **ReadWriteMany** (selected) and ReadWriteOnce.
- Mount Point Domain Name:** Text input field containing 'acs-****-****.nas.aliyuncs.com'.
- Path :** Text input field with placeholder text 'Example : /xxx'.
- Privilege :** Text input field with placeholder text 'Example : 755'.
- Tag :** A blue '+Add Tag' button.

At the bottom right, there are 'Create' and 'Cancel' buttons.

5. Click **Create** after the configurations.

Step 2. Create PVC

Use the `nas-pvc.yaml` file to create the PVC

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
```

```

name: pvc-nas
spec:
  accessModes:
    - ReadWriteMany
  storageClassName: nas
  resources:
    requests:
      storage: 5Gi

```

Step 3 Create pod

Use the *nas-pod.yaml* file to create the pod.

```

apiVersion: v1
kind: Pod
metadata:
  name: "flexvolume-nas-example"
spec:
  containers:
    - name: "nginx"
      image: "nginx"
      volumeMounts:
        - name: pvc-nas
          mountPath: "/data"
  volumes:
    - name: pvc-nas
      persistentVolumeClaim:
        claimName: pvc-nas

```

Using NFS drive of Kubernetes.

Step 1 Create a NAS file system

Log on to the [NAS console](#) to create a NAS file system.



Note:

The created NAS file system and your cluster must be in the same region.

Assume that your mount point is `055f84ad83-ixxxx.cn-hangzhou.nas.aliyuncs.com`.

Step 2 Create PVC

You can create NAS data volumes in the Container Service console or by using an orchestration template.

- **Use an orchestration template**

Use the *nas-pv.yaml* file to create the PV.

Run the following commands to create a NAS type PersistentVolume.

```

root@master # cat << EOF |kubectl apply -f -
apiVersion: v1
kind: PersistentVolume

```

```

metadata:
  name: nas
spec:
  capacity:
    storage: 8Gi
  accessModes:
    - ReadWriteMany
  persistentVolumeReclaimPolicy: Retain
  nfs:
    path: /
    server: 055f84ad83-ixxxx.cn-hangzhou.nas.aliyuncs.com
EOF

```

- **Create NAS data volumes on Container Service console**

For more information, see Create NAS data volumes on Container Service console in [Use PV/PVC](#).

Step 2 Create PVC

Create a PersistentVolumeClaim to request to bind this PersistentVolume.

```

root@master # cat << EOF | kubectl apply -f -
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: nasclaim
spec:
  accessModes:
    - ReadWriteMany
  resources:
    requests:
      storage: 8Gi
EOF

```

Step 3 Create pod

Create an application to declare to mount and use this data volume.

```

root@master # cat << EOF | kubectl apply -f -
apiVersion: v1
kind: Pod
metadata:
  name: mypod
spec:
  containers:
    - name: myfrontend
      image: registry.aliyuncs.com/spacexnice/netdia:latest
      volumeMounts:
        - mountPath: "/var/www/html"
          name: mypd
  volumes:
    - name: mypd
      persistentVolumeClaim:
        claimName: nasclaim
EOF

```

Then, the NAS remote file system is mounted to your pod application.

Dynamic storage volumes

To use dynamic NAS storage volumes, you must manually install the drive plug-in and configure the NAS mount point.

Install the plug-in

```

apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: alicloud-nas
provisioner: alicloud/nas
---
apiVersion: v1
kind: ServiceAccount
metadata:
  name: alicloud-nas-controller
  namespace: kube-system
---
kind: ClusterRoleBinding
apiVersion: rbac.authorization.k8s.io/v1beta1
metadata:
  name: run-alicloud-nas-controller
subjects:
- kind: ServiceAccount
  name: alicloud-nas-controller
  namespace: kube-system
roleRef:
  kind: ClusterRole
  name: alicloud-disk-controller-runner
  apiGroup: rbac.authorization.k8s.io
---
kind: Deployment
apiVersion: extensions/v1beta1
metadata:
  name: alicloud-nas-controller
  namespace: kube-system
spec:
  replicas: 1
  strategy:
    type: Recreate
  template:
    metadata:
      labels:
        app: alicloud-nas-controller
    spec:
      tolerations:
      - effect: NoSchedule
        operator: Exists
        key: node-role.kubernetes.io/master
      - effect: NoSchedule
        operator: Exists
        key: node.cloudprovider.kubernetes.io/uninitialized
      nodeSelector:
        node-role.kubernetes.io/master: ""
      serviceAccount: alicloud-nas-controller
      containers:
      - name: alicloud-nas-controller
        image: registry.cn-hangzhou.aliyuncs.com/acs/alibabacloud-nas-controller:v1.8.4

```

```

    volumeMounts:
      - mountPath: /persistentvolumes
        name: nfs-client-root
    env:
      - name: PROVISIONER_NAME
        value: alicloud/nas
      - name: NFS_SERVER
        value: 0cd8b4a576-mm32.cn-hangzhou.nas.aliyuncs.com
      - name: NFS_PATH
        value: /
  volumes:
    - name: nfs-client-root
      nfs:
        server: 0cd8b4a576-mm32.cn-hangzhou.nas.aliyuncs.com
        path: /

```

Use dynamic storage volumes

```

apiVersion: apps/v1beta1
kind: StatefulSet
metadata:
  name: web
spec:
  serviceName: "nginx"
  replicas: 2
  volumeClaimTemplates:
    - metadata:
        name: html
      spec:
        accessModes:
          - ReadWriteOnce
        storageClassName: alicloud-nas
        resources:
          requests:
            storage: 2Gi
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
        - name: nginx
          image: nginx:alpine
          volumeMounts:
            - mountPath: "/usr/share/nginx/html/"
              name: html

```

1.9.5 Use Alibaba Cloud OSS

You can use the Alibaba Cloud Object Storage Service (OSS) data volumes in Alibaba Cloud Container Service Kubernetes clusters.

Currently, OSS static storage volumes are supported, while OSS dynamic storage volumes are not supported. You can use the OSS static storage volumes by:

- Using the volume method.

- Using PV/PVC.

Prerequisites

You must create a bucket in the OSS console before using the OSS static storage volumes.

Instructions

- OSS is a shared storage and can provide shared storage service for multiple pods at the same time.
- bucket: Currently, Container Service only supports mounting buckets and cannot mount the sub-directories or files under the bucket.
- url: The OSS endpoint, which is the access domain name for mounting OSS.
- akId: Your AccessKey ID.
- akSecret: Your AccessKey Secret.
- otherOpts: Customized parameter input in the format of `-o *** -o ***` is supported when mounting OSS.

Note

If your Kubernetes cluster is created before Feb 6th, 2018, [Install the plug-in](#) before using the data volumes. To use OSS data volumes, you must create the secret and enter the AccessKey information when deploying the flexvolume service.

Use OSS static storage volumes

Use volume method

Use the `oss-deploy.yaml` file to create the pod.

```
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: nginx-oss-deploy
spec:
  replicas: 1
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
        - name: nginx-flexvolume-oss
          image: nginx
          volumeMounts:
            - name: "oss1"
              mountPath: "/data"
      volumes:
        - name: "oss1"
          flexVolume:
```

```

driver: "alicloud/oss"
options:
  bucket: "docker"
  url: "oss-cn-hangzhou.aliyuncs.com"
  akId: ***
  akSecret: ***
  otherOpts: "-o max_stat_cache_size=0 -o allow_other"

```

Use PV/PVC (currently, dynamic pv is not supported)

Step 1 Create PV

You can create the PV in the Container Service console or by using the YAML file.

Create PV by using YAML file

Use the `oss-pv.yaml` file to create the PV.

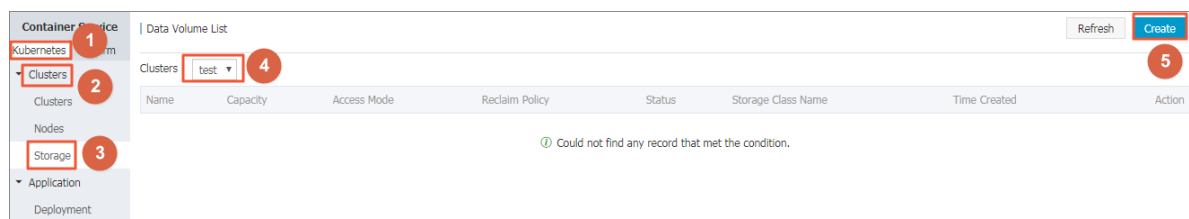
```

apiVersion: v1
kind: PersistentVolume
metadata:
  name: pv-oss
spec:
  capacity:
    storage: 5Gi
  accessModes:
    - ReadWriteMany
  storageClassName: oss
  flexVolume:
    driver: "alicloud/oss"
    options:
      bucket: "docker"
      url: "oss-cn-hangzhou.aliyuncs.com"
      akId: ***
      akSecret: ***
      otherOpts: "-o max_stat_cache_size=0 -o allow_other"

```

Create OSS data volumes in Container Service console

1. Log on to the [Container Service console](#).
2. Click **Kubernetes > Clusters > > Storage** in the left-side navigation pane.
3. Select the cluster from the Clusters drop-down list and then click **Create** in the upper-right corner.



4. The Create Data Volume dialog box appears. Configure the data volume parameters.

- **Type:** Select OSS in this example.

- **Name:** Enter the name of the data volume you are about to create. The data volume name must be unique in the cluster. In this example, enter pv-oss.
- **Capacity:** Enter the capacity of the data volume to be created.
- **Access Mode:** ReadWriteMany by default.
- **Access Key ID/Access Key Secret:** The AccessKey required to access OSS.
- **Bucket ID:** Select the OSS bucket name you want to use. Click **Select Bucket**. Select the bucket in the displayed dialog box and click **Select**.
- **Access Domain Name:** If the bucket and Elastic Compute Service (ECS) instance are in different regions, select **Internet**. If the bucket and ECS instance are in the same region, select Intranet or VPC according to the cluster network type. Select **VPC** if the network type is Virtual Private Cloud (VPC) or select **Intranet** if the network type is classic network.
- **Tag:** Click Add Tag to add tags for this data volume.

Create Data Volume

Type: ☐ Cloud Disk ☐ NAS ☒ OSS

Name:

pv-oss

Capacity :

20Gi

Access Mode :

☒ ReadWriteMany

Access Key ID:

xxxxxxxxxxxx

Access Key Secret:

xxxxxxxxxxxx

Optional Parameters:

For the formats of other parameters, refer to this document. Example: -o allow_other -o default_permission=666 -onoxattr

Bucket ID:

Select Bucket

Access Domain Name:

☐ Intranet ☐ Internet ☒ VPC ?

Tag :

+Add Tag

Create

Cancel

5. Click **Create** after completing the configurations.

Step 2 Create PVC

Use the `oss-pvc.yaml` file to create the PVC.

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: pvc-oss
spec:
  storageClassName: oss
  accessModes:
    - ReadWriteMany
  resources:
    requests:
      storage: 5Gi
```

Step 3 Create pod

Use the `oss-pod.yaml` file to create the pod.

```
apiVersion: v1
kind: Pod
metadata:
  name: "flexvolume-oss-example"
spec:
  containers:
    - name: "nginx"
      image: "nginx"
      volumeMounts:
        - name: pvc-oss
          mountPath: "/data"
  volumes:
    - name: pvc-oss
      persistentVolumeClaim:
        claimName: pvc-oss
```

Use OSS dynamic storage volumes

Currently not supported.

1.9.6 Create a persistent storage volume claim

You can create a persistent storage volume claim (PVC) by using the Container Service console.

Prerequisites

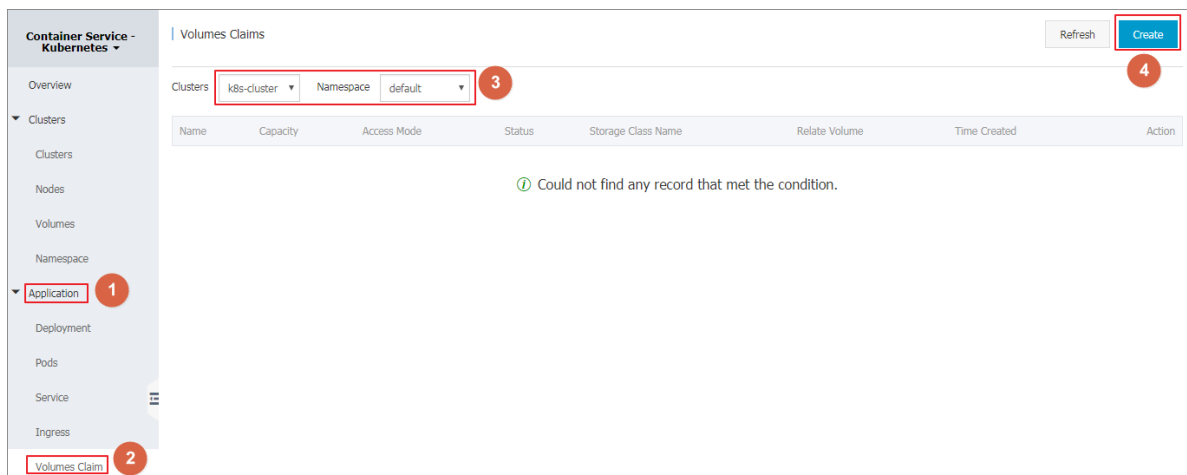
- You have created a Kubernetes cluster. For more information, see [Create a Kubernetes cluster](#).
- If you have already created a storage volume, use a cloud disk to create a cloud storage volume. For more information, see [Use Alibaba Cloud cloud disks](#).

By default, the storage claim is bound to the storage volume depending on the label `alicloud-pvname`. When the data volume is created by using the Container Service console, the storage volume is labeled by default. If the storage volume label does not exist, you must add a label before you select to bound this storage volume.

Context

Procedure

1. Log on to the [Container Service console](#).
2. Under Kubernetes, click **Application > Volumes Claim** in the left-side navigation pane to enter the Volumes Claims list page.
3. Select the target cluster and namespace, and click **Create** in the upper-right corner.



4. Complete the configurations in the Create Volume Claim dialog box, and click **Create**.

Create Volume Claims

Volume type :

☒ Cloud Disk
☐ NAS
☐ OSS

Name :

pv-disk

Name must start with a lowercase letter and can only contain lowercase letters, numbers, "." and "-"

Allocate mode :

☒ Existing volume

Existing volume :

d-bp12o50zzxanyje4nsew, 20Gi [Select Volume](#)

Capacity :

20Gi

Create

Cancel

- **Volume claim type:** Consistent with storage volume, including cloud, NAS, and OSS types.
- **Name:** Enter the storage volume claim name.
- **Distribution mode:** Currently, only existing storage volumes are supported.
- **Existing storage volume:** Select to bound the storage volume of this type.
- **Total:** Claim usage, cannot be greater than the total amount of storage volumes.



Note:

If a storage volume already exists in your cluster and is not used, but cannot be found in **Select Existing Storage Volume**, maybe the `alicloud-pvname` label is not defined.

If you cannot find an available storage volume, you can click **Clusters > Volumes** in the left-side navigation pane. Find the target storage volume, click **Label Management** on the right. Add the corresponding label `alicloud-pvname`, the value is the name of the storage volume. The cloud storage volume defaults to the cloud disk ID as the name of the storage volume.

Name	Value
alicloud-pvname	d-bp1-7330t00c7-emix3iv0e
failure-domain.beta.kubernetes.io/zone	cn-hangzhou-g
failure-domain.beta.kubernetes.io/region	cn-hangzhou

5. Return to the Volumes Claims list, you can see that the newly created storage claim appears in the list.

1.9.7 Using persistent storage volume claim

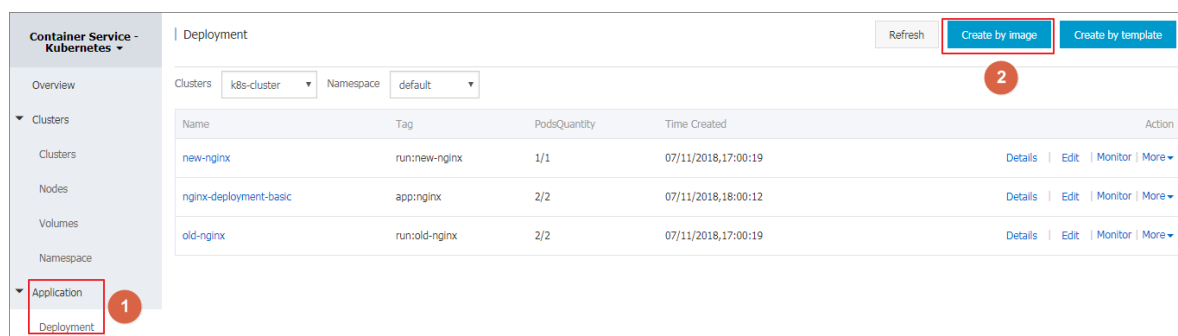
On the Container Service console, use an image or a template to deploy an application, so that you can use a persistent storage volume claim. In this example, an image is used to create an application. If you want to use a persistent storage volume claim with the template, see [Use Alibaba Cloud cloud disks](#).

Prerequisites

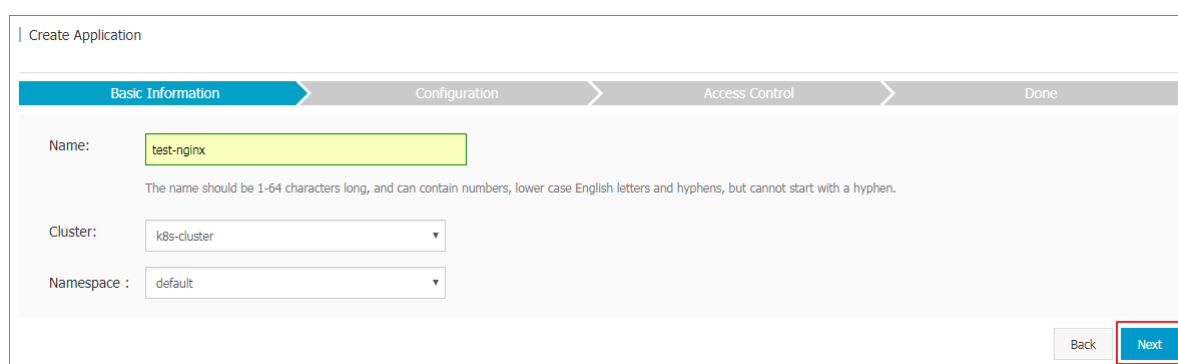
- You have created a Kubernetes cluster. For more information, see [Create a Kubernetes cluster](#).
- If you have already created a storage volume claim, use the cloud disk to create a cloud disk storage volume claim PVC disk. For more information, see [Create a persistent storage volume claim](#).

Procedure

1. Log on to the [Container Service console](#).
2. Under Kubernetes, click **Application > Deployment** in the left-side navigation pane. Enter the Deployment List page and click **Create by image** in the upper-right corner.



3. On the Basic Information page, configure the application name, deploy the cluster, and the namespace. Then click **Next**.



4. On the Application Configuration page, select Image. Then configure the cloud storage type of data volume, cloud disk, NAS, and OSS types are supported. In this example, use the cloud storage volume claim and click **Next**.

General

Image Name: Image Version:
[Select image](#) [Select image version](#)

Scale:

Resource Limit: CPU Memory
 Resource Request: CPU Memory

Volume

Data Volume: [+ Add local storage](#)

Storage type	Mount source	Container Path
+ Add cloud storage		
Disk	pvc-disk	/tmp

5. See [Create a service](#) to configure the test-nginx application, and click **Create**.
6. After the application is created, click **Apply > Container Group** in the left-side navigation pane. Find the container group to which the application belongs, and click **Details**.

Container Service - Kubernetes

Overview

Clusters

Clusters

Nodes

Volumes

Namespace

Application (1)

Deployment

Pods (2)

Pods

Clusters: Namespace: (3)

Name	Status	Pod IP	Node	Time Created	CPU	Memory
test-nginx-deployment-76dbb97577-5klvr	Running			07/12/2018, 15:34:56	0	0

Detail (4) More

7. On the Container Group details page, click **Storage** to view the container group is properly bound to the PVC disk.

Pods - test-nginx-deployment-76dbb97577-5klvr Refresh

Overview

Name : test-nginx-deployment-76dbb97577-5klvr	Namespace : default
Status : Running	Time Created : 07/12/2018,15:34:56
Node : cn-hangzhou.i-bp153888e85yy0cyw659	Pod IP : 10.0.1.197
Tag : app: test-nginx pod-template-hash: 3286653133	

Container Events Created by Init Containers **Volumes** Logs

Name	Type	Details
volume-1531380735073	persistentVolumeClaim	claimName: pvc-disk
default-token-w689p	secret	defaultMode: 420 secretName: default-token-w689p

1.10 Logs

1.10.1 Overview

The Alibaba Cloud Container Service Kubernetes cluster provides you with multiple methods to manage application logs.

- With the open-source Fluentd-pilot project provided by Alibaba Cloud Container Service, you can conveniently [#unique_108](#) to better use the features provided by Alibaba Cloud Log Service such as log statistics and analysis.
- [Configure Log4jAppender for Kubernetes and Log Service.](#)
- [A solution to log collection problems of Kubernetes clusters by using log-pilot, Elasticsearch, and Kibana.](#)

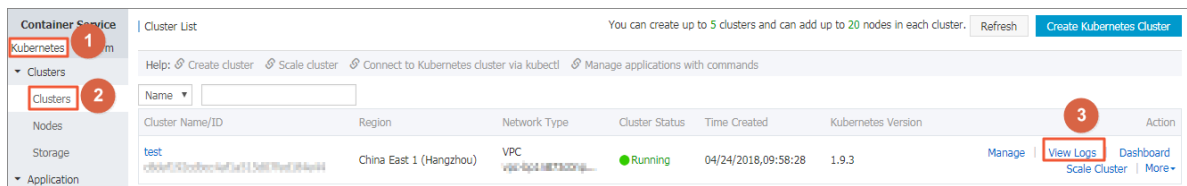
1.10.2 View cluster logs

Context

You can view the cluster operation logs by using the simple log service of Container Service.

Procedure

1. Log on to the [Container Service console](#).
2. Click Kubernetes > **Clusters** in the left-side navigation pane.
3. Click **View Logs** at the right of the cluster.



View the cluster operation information.

Cluster Logs: test Back to Cluster List		Refresh
Detailed resource deployment logs: Stack Events		
Time	Information	
04/24/2018,13:55:38	c8def192cdbcc4af1a515d07fed184e44 Start to client.DescribeTemplate	
04/24/2018,13:55:35	c8def192cdbcc4af1a515d07fed184e44 Start describeStackInfo	
04/24/2018,11:27:38	c8def192cdbcc4af1a515d07fed184e44 Start to client.DescribeTemplate	
04/24/2018,11:27:36	c8def192cdbcc4af1a515d07fed184e44 Start describeStackInfo	
04/24/2018,11:27:08	c8def192cdbcc4af1a515d07fed184e44 Start to client.DescribeTemplate	
04/24/2018,11:27:06	c8def192cdbcc4af1a515d07fed184e44 Start describeStackInfo	
04/24/2018,11:26:55	c8def192cdbcc4af1a515d07fed184e44 Start to client.DescribeTemplate	
04/24/2018,11:26:54	c8def192cdbcc4af1a515d07fed184e44 Start describeStackInfo	
04/24/2018,11:25:02	c8def192cdbcc4af1a515d07fed184e44 Start to client.DescribeTemplate	
04/24/2018,11:25:00	c8def192cdbcc4af1a515d07fed184e44 Start describeStackInfo	
04/24/2018,10:16:42	c8def192cdbcc4af1a515d07fed184e44 Set up k8s DNS configuration successfully	
04/24/2018,10:15:36	c8def192cdbcc4af1a515d07fed184e44 Start describeStackInfo	
04/24/2018,10:15:36	c8def192cdbcc4af1a515d07fed184e44 Stack CREATE completed successfully:	
04/24/2018,10:15:34	c8def192cdbcc4af1a515d07fed184e44 Start describeStackInfo	

1.10.3 Configure Log4jAppender for Kubernetes and Log Service

Log4j is an open-source project of Apache, which consists of three important components: log level, log output destination, and log output format. By configuring Log4jAppender, you can set the log output destination to console, file, GUI component, socket server, NT event recorder, or UNIX Syslog daemon.

This document introduces how to configure a YAML file to output Alibaba Cloud Container Service Kubernetes cluster logs to Alibaba Cloud Log Service, without modifying the application codes. In this document, deploy a sample API application in the Kubernetes cluster for demonstration.

Prerequisites

- You have activated Container Service and created a Kubernetes cluster.

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

- Enable AccessKey or Resource Access Management (RAM). Make sure you have sufficient access permissions. Use the AccessKey in this example.

Step 1 Configure Log4jAppender in Alibaba Cloud Log Service

1. Log on to the [Log Service console](#).
2. On the Project List page, click **Create Project** in the upper-right corner. Complete the configurations and then click **Confirm** to create the project.

In this example, create a project named k8s-log4j and select the same region (China East 1 (Hangzhou)) as the Kubernetes cluster.



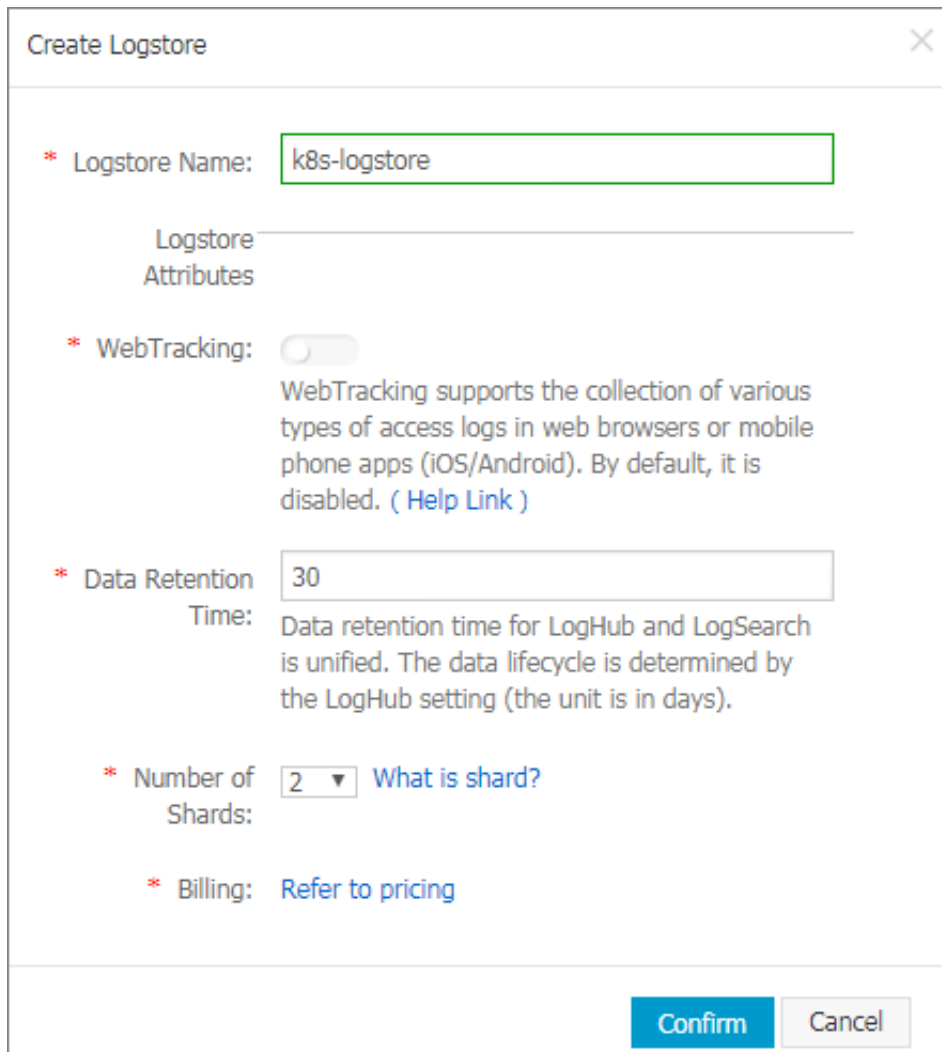
Note:

Generally, create a Log Service project in the same region as the Kubernetes cluster. When the Kubernetes cluster and Log Service project are in the same region, log data is transmitted by using the intranet, which saves the Internet bandwidth cost and time of data transmission because of different regions, and implements the best practice of real-time collection and quick query.

3. After being created, the project k8s-log4j is displayed on the Project List page. Click the project name.
4. The **Logstore List** page appears. Click **Create** in the upper-right corner.

5. Complete the configurations and then click **Confirm**.

In this example, create a Logstore named k8s-logstore.

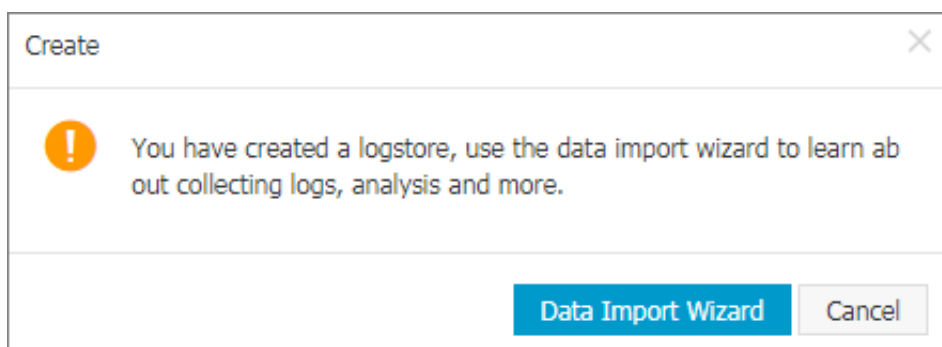


The 'Create Logstore' dialog box contains the following fields and options:

- Logstore Name:** A text input field containing 'k8s-logstore'.
- Logstore Attributes:** A section header.
- WebTracking:** A toggle switch that is currently turned off. Below it, a description states: 'WebTracking supports the collection of various types of access logs in web browsers or mobile phone apps (iOS/Android). By default, it is disabled. ([Help Link](#))'.
- Data Retention Time:** A text input field containing '30'. Below it, a description states: 'Data retention time for LogHub and LogSearch is unified. The data lifecycle is determined by the LogHub setting (the unit is in days).'.
- Number of Shards:** A dropdown menu showing '2' and a link that says 'What is shard?'.
- Billing:** A link that says 'Refer to pricing'.

At the bottom right, there are two buttons: 'Confirm' (highlighted in blue) and 'Cancel'.

6. Then, a dialog box asking you to use the data import wizard appears.

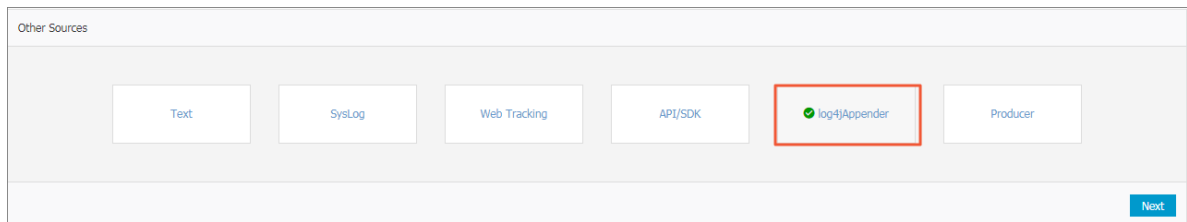


The 'Create' dialog box displays a warning icon (an orange circle with an exclamation mark) and the following text: 'You have created a logstore, use the data import wizard to learn about collecting logs, analysis and more.'

At the bottom right, there are two buttons: 'Data Import Wizard' (highlighted in blue) and 'Cancel'.

7. Click Data Import Wizard. In the Select Data Source step, select log4jAppender under **Other Sources** and then complete the configurations as instructed on the page.

Use the default configurations in this example. Configure the settings according to the specific scenarios of log data.



Step 2 Configure Log4jAppender in the Kubernetes cluster

In this example, use the sample YAML files [demo-deployment](#) and [demo-service](#) for demonstration.

1. Connect to your Kubernetes cluster.

For more information, see [Access Kubernetes clusters by using SSH](#) or [Connect to a Kubernetes cluster by using kubectl](#).

2. Obtain the `demo-deployment.yaml` file and configure the environment variable `JAVA_OPTS` to collect logs from the Kubernetes cluster.

The sample orchestration of the `demo-deployment.yaml` file is as follows:

```
apiVersion: apps/v1beta2
kind: Deployment
metadata:
  name: log4j-appender-demo-spring-boot
  labels:
    app: log4j-appender
spec:
  replicas: 1
  selector:
    matchLabels:
      app: log4j-appender
  template:
    metadata:
      labels:
        app: log4j-appender
    spec:
      containers:
        - name: log4j-appender-demo-spring-boot
          image: registry.cn-hangzhou.aliyuncs.com/jaegertracing/log4j-appender-demo-spring-boot:0.0.2
          env:
            - name: JAVA_OPTS ##Note
              value: "-Dproject={your_project} -Dlogstore={your_logstore} -Dendpoint={your_endpoint} -Daccess_key_id={your_access_key_id} -Daccess_key={your_access_key_secret}"
          ports:
            - containerPort: 8080
```

Wherein:

- `-Dproject`: The name of the used Alibaba Cloud Log Service project. In this example, it is `k8s-log4j`.
- `-Dlogstore`: The name of the used Alibaba Cloud Log Service Logstore. In this example, it is `k8s-logstore`.
- `-Dendpoint`: The service endpoint of Log Service. You must configure your service endpoint according to the region where the Log Service project resides. For more information, see [Service endpoint](#). In this example, it is `cn-hangzhou.log.aliyuncs.com`.
- `-Daccess_key_id`: Your AccessKey ID.
- `-Daccess_key`: Your AccessKey Secret.

3. Run the following command in the command line to create the deployment:

```
kubectl create -f demo-deployment.yaml
```

4. Obtain the `demo-service.yaml` file and run the following command to create the service.

No need to modify the configurations in the `demo-service.yaml` file.

```
kubectl create -f demo-service.yaml
```

Step 3 Test to generate Kubernetes cluster logs

You can run the `kubectl get` command to view the deployment status of the resource object.

Wait until the deployment and the service are successfully deployed. Then, run the `kubectl get svc` command to view the external access IP of the service, that is, the EXTERNAL-IP.

```
$ kubectl get svc
NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE
log4j-appender-demo-spring-boot-svc LoadBalancer 172.21. XX.XX 120.55
. XXX.XXX 8080:30398/TCP 1h
```

In this example, test to generate Kubernetes cluster logs by running the `login` command, wherein, `K8S_SERVICE_IP` is the EXTERNAL-IP.



Note:

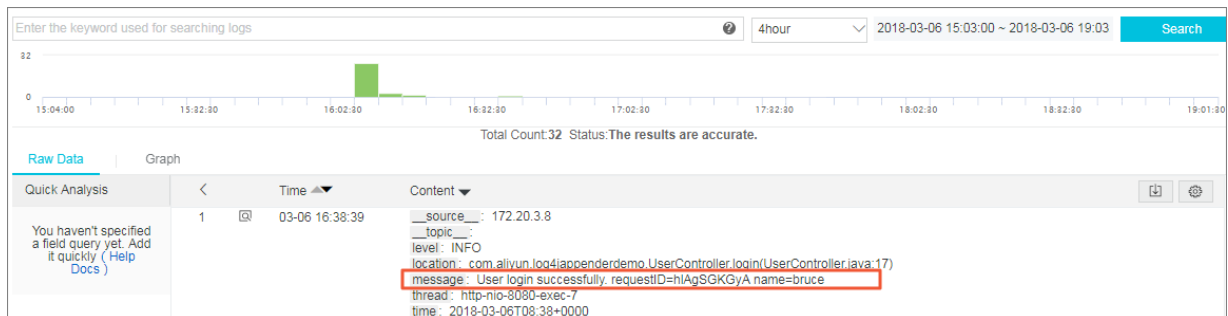
See [GitHub log4j-appender-demo](#) to view the complete collection of APIs.

```
curl http://${K8S_SERVICE_IP}:8080/login? name=bruce
```

Step 4 View logs in Alibaba Cloud Log Service

Log on to the [Log Service console](#).

Click the project name and click **Search** at the right of the Logstore k8s-logstore to view the output logs of the Kubernetes cluster.



The output content of the log corresponds to the preceding command. This example demonstrates how to output the logs of the sample application to Alibaba Cloud Log Service. By completing the preceding steps, you can configure Log4JAppender in Alibaba Cloud and implement advanced functions such as collecting logs in real time, filtering data, and querying logs by using Alibaba Cloud Log Service.

1.10.4 A solution to log collection problems of Kubernetes clusters by using log-pilot, Elasticsearch, and Kibana

Requirements for logs of distributed Kubernetes clusters always bother developers. This is mainly because of the characteristics of containers and the defects of log collection tools.

- Characteristics of containers:
 - Many collection targets: The characteristics of containers cause the number of collection targets is large, which requires to collect the container logs and container stdout. Currently, no good tool can collect file logs from containers dynamically. Different data sources have different collection softwares. However, no one-stop collection tool exists.
 - Difficulty caused by auto scaling: Kubernetes clusters are in the distributed mode. The auto scaling of services and the environment brings great difficulty to log collection. You cannot configure the log collection path in advance, the same as what you do in the traditional virtual machine (VM) environment. The dynamic collection and data integrity are great challenges.
- Defects of current log collection tools:
 - Lack the capability to dynamically configure log collection: The current log collection tools require you to manually configure the log collection method and path in advance. These tools cannot dynamically configure the log collection because they cannot automatically detect the lifecycle changes or dynamic migration of containers.

- Log collection problems such as logs are duplicate or lost: Some of the current log collection tools collect logs by using the tail method. Logs may be lost in this way. For example, the application is writing logs when the log collection tool is being restarted. Logs written during this period may be lost. Generally, the conservative solution is to collect logs of 1 MB or 2 MB previous to the current log by default. However, this may cause the duplicate log collection.
- Log sources without clear marks: An application may have multiple containers that output the same application logs. After all the application logs are collected to a unified log storage backend, you cannot know a log is generated on which application container of which node when querying logs.

This document introduces log-pilot, a tool to collect Docker logs, and uses the tool together with Elasticsearch and Kibana to provide a one-stop solution to log collection problems in the Kubernetes environment.

Introduction on log-pilot

Log-pilot is an intelligent tool used to collect container logs, which not only collects container logs and outputs these logs to multiple types of log storage backends efficiently and conveniently, but also dynamically discovers and collects log files from containers.

Log-pilot uses declarative configuration to manage container events strongly and obtain the stdout and file logs of containers, which solves the problem of auto scaling. Besides, log-pilot has the functions of automatic discovery, maintenance of checkpoint and handle, and automatic tagging for log data, which effectively deals with the problems such as dynamic configuration, duplicate logs, lost logs, and log source marking.

Currently, log-pilot is completely open-source in GitHub. The project address is <https://github.com/AliyunContainerService/log-pilot>. You can know more implementation principles about it.

Declarative configuration for container logs

Log-pilot supports managing container events, can dynamically listen to the event changes of containers, parse the changes according to the container labels, generate the configuration file of log collection, and then provide the file to collection plug-in to collect logs.

For Kubernetes clusters, log-pilot can dynamically generate the configuration file of log collection according to the environment variable `aliyun_logs_$name = $path`. This environment variable contains the following two variables:

- One variable is `$name`, a custom string which indicates different meanings in different scenarios. In this scenario, `$name` indicates index when collecting logs to Elasticsearch.
- The other is `$path` which supports two input modes, `stdout` and paths of log files within containers, respectively corresponding to the standard output of logs and log files within containers.
 - `Stdout` indicates to collect standard output logs from containers. In this example, to collect Tomcat container logs, configure the label `aliyun.logs.catalina=stdout` to collect standard output logs of Tomcat.
 - The path of a log file within a container also supports wildcards. To collect logs within the Tomcat container, configure the environment variable `aliyun_logs_access=/usr/local/tomcat/logs/*.log`. To not use the keyword `aliyun`, you can use the environment variable `PILOT_LOG_PREFIX`, which is also provided by log-pilot, to specify the prefix of your declarative log configuration. For example, `PILOT_LOG_PREFIX: "aliyun,custom"`.

Besides, log-pilot supports multiple log parsing formats, including none, JSON, CSV, Nginx, apache2, and regexp. You can use the `aliyun_logs_$name_format=<format>` label to tell log-pilot to use what format to parse logs when collecting logs.

Log-pilot also supports custom tags. If you configure `aliyun_logs_$name_tags="K1=V1,K2=V2"` in the environment variable, `K1=V1` and `K2=V2` are collected to log output of the container during the log collection. Custom tags help you tag the log generation environment for convenient statistics, routing, and filter of logs.

Log collection mode

In this document, deploy a log-pilot on each machine and collect all the Docker application logs from the machines.

Compared with deploying a logging container on each pod, the most obvious advantage of this solution is less occupied resources. The larger the cluster scale is, the more obvious the advantage is. This solution is also recommended in the community.

Prerequisites

You have activated Container Service and created a Kubernetes cluster. In this example, create a Kubernetes cluster in China East 1 (Hangzhou).

Step 1 Deploy Elasticsearch

1. Connect to your Kubernetes cluster. For more information, see [#unique_58](#) or [#unique_114](#).
2. Deploy the resource object related to Elasticsearch first. Then, enter the following orchestration template. This orchestration template includes an elasticsearch-api service, an elasticsearch-discovery service, and a status set of Elasticsearch. All of these objects are deployed under the namespace kube-system.

```
kubectl apply -f https://acs-logging.oss-cn-hangzhou.aliyuncs.com/elasticsearch.yml
```

3. After the successful deployment, corresponding objects are under the namespace kube-system. Run the following commands to check the running status:

```
$ kubectl get svc,StatefulSet -n=kube-system
NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE
svc/elasticsearch-api ClusterIP 172.21.5.134 <none> 9200/TCP 22h
svc/elasticsearch-discovery ClusterIP 172.21.13.91 <none> 9300/TCP 22h
...
NAME DESIRED CURRENT AGE
statefulsets/elasticsearch 3 3 22h
```

Step 2 Deploy log-pilot and the Kibana service

1. Deploy the log-pilot log collection tool. The orchestration template is as follows:

```
kubectl apply -f https://acs-logging.oss-cn-hangzhou.aliyuncs.com/log-pilot.yml
```

2. Deploy the Kibana service. The sample orchestration template contains a service and a deployment.

```
kubectl apply -f https://acs-logging.oss-cn-hangzhou.aliyuncs.com/kibana.yml
```

Step 3 Deploy the test application Tomcat

After deploying the log tool set of Elasticsearch + log-pilot + Kibana, deploy a test application Tomcat to test whether or not logs can be successfully collected, indexed, and displayed.

The orchestration template is as follows:

```
apiVersion: v1
kind: Pod
metadata:
  name: tomcat
  namespace: default
  labels:
    name: tomcat
spec:
```

```

containers:
- image: tomcat
  name: tomcat-test
  volumeMounts:
  - mountPath: /usr/local/tomcat/logs
    name: accesslogs
  env:
  - name: aliyun_logs_catalina
    value: "stdout" ##Collect standard output logs.
  - name: aliyun_logs_access
    value: "/usr/local/tomcat/logs/catalina. *.log" ## Collect log
files within the container.
  volumes:
  - name: accesslogs
    emptyDir: {}

```

The Tomcat image is a Docker image that both uses stdout and file logs. In the preceding orchestration, the log collection configuration file is dynamically generated by defining the environment variable in the pod. See the following descriptions for the environment variable:

- `aliyun_logs_catalina=stdout` indicates to collect stdout logs from the container.
- `aliyun_logs_access=/usr/local/tomcat/logs/catalina. *.log` indicates to collect all the log files whose name matches `catalina. *.log` under the directory `/usr/local/tomcat/logs/` from the container.

In the Elasticsearch scenario of this solution, the `$name` in the environment variable indicates index. In this example, `$name` is `catalina` and `access`.

Step 4 Expose the Kibana service to Internet

The Kibana service deployed in the preceding section is of the NodePort type, which cannot be accessed from the Internet by default. Therefore, create an Ingress in this document to access the Kibana service from Internet and test whether or not logs are successfully indexed and displayed.

1. Create an Ingress to access the Kibana service from Internet. In this example, use the simple routing service to create an Ingress. For more information, see [#unique_115](#). The orchestration template of the Ingress is as follows:

```

apiVersion: extensions/v1beta1
kind: Ingress
metadata:
  name: kibana-ingress
  namespace: kube-system #Make sure the namespace is the same as
that of the Kibana service.
spec:
  rules:
  - http:
    paths:
    - path: /
      backend:

```

```

        serviceName: kibana #Enter the name of the Kibana service
    .
        servicePort: 80 #Enter the port exposed by the Kibana
service.

```

2. After the Ingress is successfully created, run the following commands to obtain the access address of the Ingress:

```

$ kubectl get ingress -n=kube-system
NAME HOSTS ADDRESS PORTS AGE
shared-dns * 120.55.150.30 80 5m

```

3. Access the address in the browser as follows.

4. Click **Management** in the left-side navigation pane. Then, click **Index Patterns > Create Index Pattern**. The detailed index name is the `$name` variable suffixed with a time string. You can create an index pattern by using the wildcard `*`. In this example, use `$name*` to create an index pattern.

You can also run the following commands to enter the corresponding pod of Elasticsearch and list all the indexes of Elasticsearch:

```

$ kubectl get pods -n=kube-system #Find the corresponding pod of
Elasticsearch.
...
$ kubectl exec -it elasticsearch-1 bash #Enter a pod of Elasticsea
rch.
...
$ curl 'localhost:9200/_cat/indices? v' ## List all the indexes.
health status index uuid pri rep docs.count docs.deleted store.size
pri.store.size
green open .kibana x06jj19PS4Cim6Ajo51PWg 1 1 4 0 53.6kb 26.8kb
green open access-2018.03.19 txd3tG-NR6-guqmMEKKzEw 5 1 143 0 823.
5kb 411.7kb
green open catalina-2018.03.19 ZgtWd16FQ7qqJNNWXxFPcQ 5 1 143 0 915
.5kb 457.5kb

```

5. After successfully creating the indexes, click **Discover** in the left-side navigation pane, select the created index and the corresponding time range, and then enter the related field in the search box to query logs.

Then, you have successfully tested the solution to log collection problems of Alibaba Cloud Kubernetes clusters based on log-pilot, Elasticsearch, and Kibana. By using this solution, you can deal with requirements for logs of distributed Kubernetes clusters effectively, improve the Operation and Maintenance and operational efficiencies, and guarantee the continuous and stable running of the system.

1.11 Monitoring

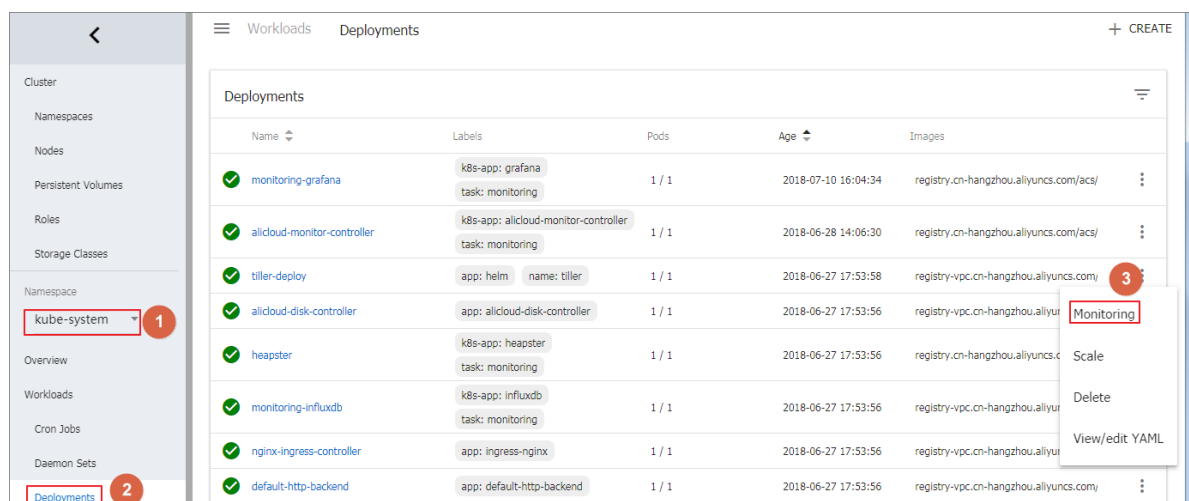
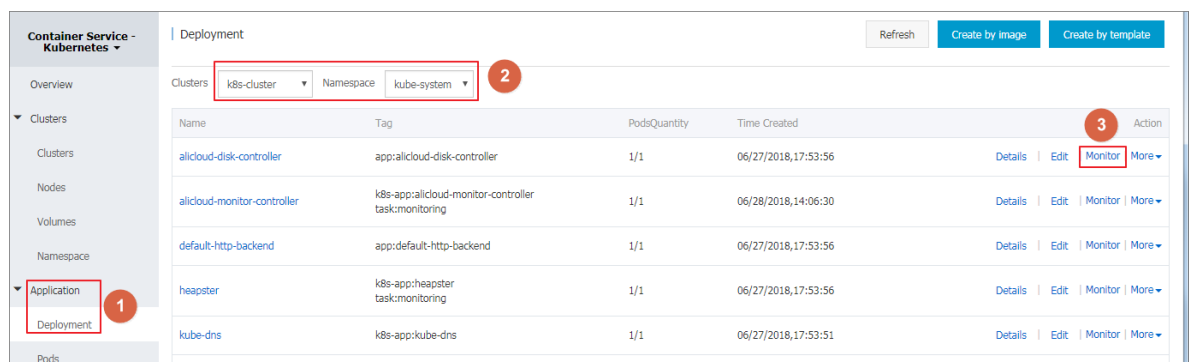
1.11.1 Integration and usage with CloudMonitor

Prerequisites

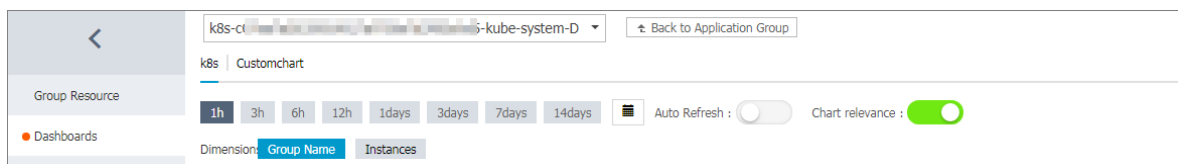
Check whether `alicloud-monitor-controller` has been deployed in the `kube-system` namespace. If not, upgrade the version of the cluster.

Procedure

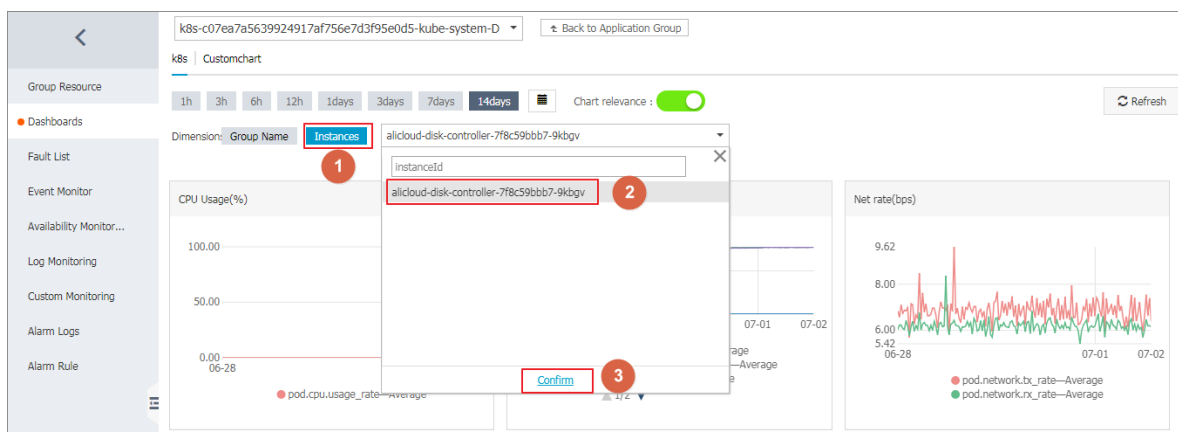
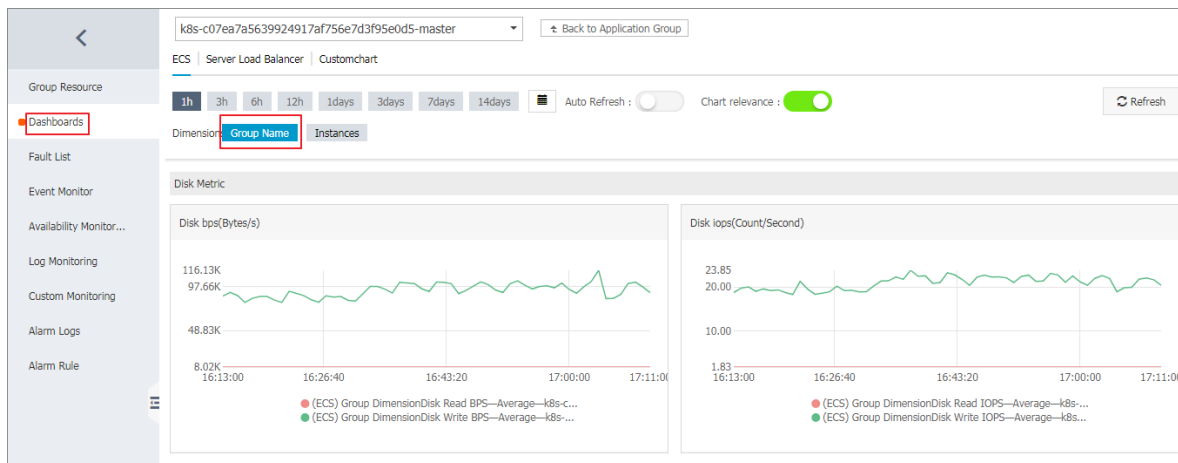
1. Log on to the [Container Service console](#).
2. Under Kubernetes, click **Deployment** in the left-side navigation pane.
3. Select the target deployment, click **Monitor** on the right. You can also click **Monitor** on the Deployment page of the built-in kubernetes dashboard.



In this case, you jump to the corresponding Application group details page of CloudMonitor.



4. Application group supports monitoring in two dimensions: **group** and **instance**.



5. For alarm settings, the index of group level starts with **group**, and the instance level index starts with **pod**.

Upgrade cluster version

1. Log on to the [Container Service console](#).
2. Under Kubernetes, click **Application > Deployment** in the left-side navigation pane to enter the Deployment List page. Click **Create by template** in the upper-right corner.

Name	Tag	PodsQuantity	Time Created	Action
new-nginx	run:new-nginx	1/1	07/11/2018,17:00:19	Details Edit Monitor More
nginx-deployment-basic	app:nginx	2/2	07/11/2018,18:00:12	Details Edit Monitor More
old-nginx	run:old-nginx	2/2	07/11/2018,17:00:19	Details Edit Monitor More
test-nginx-deployment	app:test-nginx	1/1	07/12/2018,15:34:56	Details Edit Monitor More

3. Select the target cluster, kube-system namespace, and use the following sample template. Then click **Create**.



Note:

Replace **REGION** and **CLUSTER_ID** with your actual cluster information, and redeploy heapster yaml template.

Clusters

k8s-cluster

Namespace

kube-system

Resource Type

Resource - basic Deployment

Template

```

1  apiVersion: extensions/v1beta1
2  kind: Deployment
3  metadata:
4    name: heapster
5    namespace: kube-system
6  spec:
7    replicas: 1
8    template:
9      metadata:
10     labels:
11       task: monitoring
12       k8s-app: heapster
13     annotations:
14       scheduler.alpha.kubernetes.io/critical-pod: ''
15     spec:
16       serviceAccount: admin
17     containers:
18     - name: heapster
19       image: registry.##REGION##.aliyuncs.com/acs/heapster-amd64:v1.5.1.1
20       imagePullPolicy: IfNotPresent
21       command:
22       - /heapster
23       - --source=kubernetes:https://kubernetes.default
24       - --historical-source=influxdb:http://monitoring-influxdb:8086
25       - --sink=influxdb:http://monitoring-influxdb:8086
26       - --sink=socket:tcp://monitor.csk.##REGION##.aliyuncs.com:8093?clusterId=##CLUSTER_ID##&public=true

```

DEPLOY

An example of heapster template is as follows. If you have an earlier version of the heapster in the cluster, you can log on to the Kubernetes cluster and run the `kubectl apply -f xxx.yaml` command to upgrade it.

```

apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: heapster
  namespace: kube-system
spec:
  replicas: 1
  template:
    metadata:
      labels:
        task: monitoring
        k8s-app: heapster
      annotations:
        scheduler.alpha.kubernetes.io/critical-pod: ''
    spec:
      serviceAccount: admin
      containers:
      - name: heapster

```

```

      image: registry. ##REGION##.aliyuncs.com/acs/heapster-amd64:
v1.5.1.1
      imagePullPolicy: IfNotPresent
      command:
      - /heapster
      - --source=kubernetes:https://kubernetes.default
      - --historical-source=influxdb:http://monitoring-influxdb:
8086
      - --sink=influxdb:http://monitoring-influxdb:8086
      - --sink=socket:tcp://monitor.csk. ##REGION##.aliyuncs.com:
8093? clusterId=##CLUSTER_ID##&public=true

```

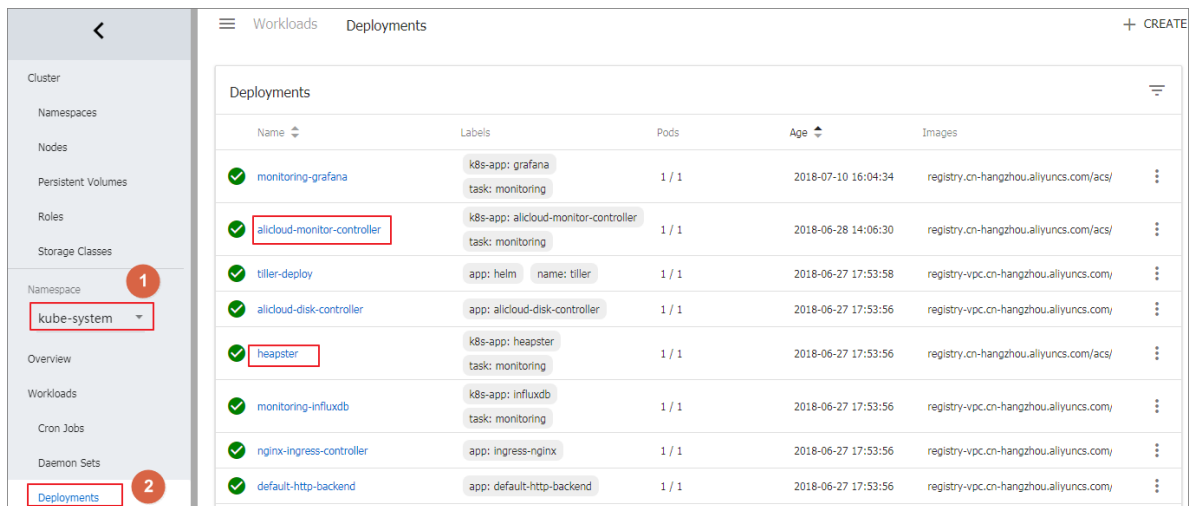
The example layout of alicloud-monitor-controller is as follows. Run the `kubectl create -f xxx.yaml` command to deploy alicloud-monitor-controller.

```

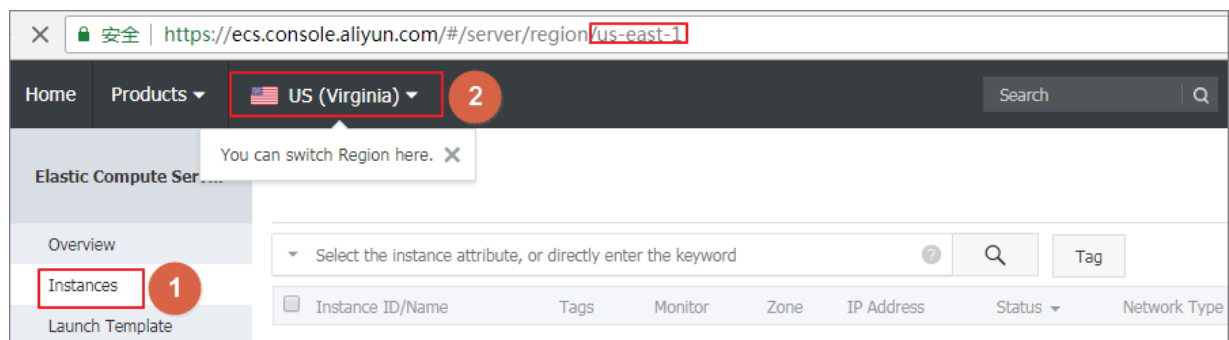
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: alicloud-monitor-controller
  namespace: kube-system
spec:
  replicas: 1
  template:
    metadata:
      labels:
        task: monitoring
        k8s-app: alicloud-monitor-controller
      annotations:
        scheduler.alpha.kubernetes.io/critical-pod: ''
    spec:
      hostNetwork: true
      tolerations:
        - effect: NoSchedule
          operator: Exists
          key: node-role.kubernetes.io/master
        - effect: NoSchedule
          operator: Exists
          key: node.cloudprovider.kubernetes.io/uninitialized
      serviceAccount: admin
      containers:
        - name: alicloud-monitor-controller
          image: registry. ##REGION##.aliyuncs.com/acs/alibabacloud-
monitor-controller:v1.0.0
          imagePullPolicy: IfNotPresent
          command:
            - /alicloud-monitor-controller
            - agent
            - --regionId=##REGION##
            - --clusterId=##CLUSTER_ID##
            - --logtostderr
            - --v=4

```

4. Go to the Kubernetes console. In the kube-system namespace, you can see that the two deployments are running, and the upgrade is complete.



If you do not know the REGION information, you can go to the ECS console and select the region where your cluster resides. The last segment of the page URL address is REGION.



1.11.2 Use Grafana to display monitoring data

Prerequisites

- You have successfully created a Kubernetes cluster. For more information, see [Create a Kubernetes cluster](#).
- In this example, use the Grafana with built-in monitoring templates and the image address is `registry.cn-hangzhou.aliyuncs.com/acs/grafana:5.0.4`.

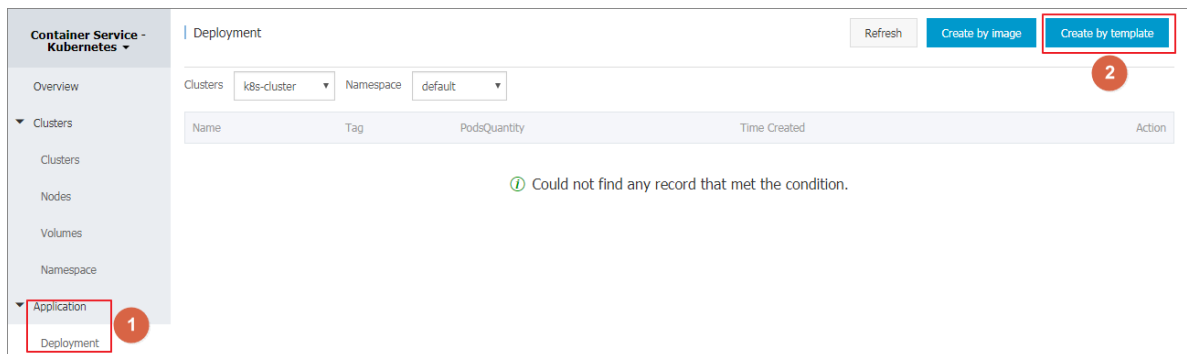
Context

Among Kubernetes monitoring solutions, compared with open-source solutions such as Prometheus, the combination of Heapster + InfluxDB + Grafana is more simple and direct. Heapster not only collects monitoring data in Kubernetes, but also is relied on by the monitoring interface of the console and the POD auto scaling of HPA. Therefore, Heapster is an essential component of Kubernetes. An Alibaba Cloud Kubernetes cluster has the built-in Heapster +

InfluxDB combination. To display the monitoring data, you must configure an available Grafana and the corresponding dashboard.

Procedure

1. Log on to the [Container Service console](#).
2. Click Kubernetes > **Application** > > **Deployment** in the left-side navigation pane.
3. Click **Create by template** in the upper-right corner.



4. Configure the template to create the deployment and service of Grafana. After completing the configurations, click **DEPLOY**.
 - Clusters: Select a cluster.
 - Namespace: Select the namespace to which the resource object belongs, which must be **kube-system**.
 - Resource Type: Select Custom in this example. The template must contain a deployment and a service.

Deploy templates

Only Kubernetes versions 1.8.4 and above are supported. For clusters of version 1.8.1, you can perform "upgrade cluster" operation in the cluster list

Clusters

Namespace

Resource Type

1

2

DEPLOY

Template

```

1 apiVersion: extensions/v1beta1
2 kind: Deployment
3 metadata:
4   name: monitoring-grafana
5   namespace: kube-system
6 spec:
7   replicas: 1
8   template:
9     metadata:
10    labels:
11      task: monitoring
12      k8s-app: grafana
13    spec:
14      containers:
15        - name: grafana
16          image: registry.cn-hangzhou.aliyuncs.com/acs/grafana:5.0.4
17          ports:
18            - containerPort: 3000
19              protocol: TCP
20          volumeMounts:
21            - mountPath: /var
22              name: grafana-storage
23          env:
24            - name: INFLUXDB_HOST
25              value: monitoring-influxdb
26          volumes:
27            - name: grafana-storage

```

The orchestration template in this example is as follows:

```

apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: monitoring-grafana
  namespace: kube-system
spec:
  replicas: 1
  template:
    metadata:
      labels:
        task: monitoring
        k8s-app: grafana
    spec:
      containers:
        - name: grafana
          image: registry.cn-hangzhou.aliyuncs.com/acs/grafana:5.0.4
          ports:
            - containerPort: 3000
              protocol: TCP
          volumeMounts:
            - mountPath: /var
              name: grafana-storage
          env:
            - name: INFLUXDB_HOST
              value: monitoring-influxdb
          volumes:
            - name: grafana-storage
              emptyDir: {}

apiVersion: v1
kind: Service

```

```

metadata:
  name: monitoring-grafana
  namespace: kube-system
spec:
  ports:
  - port: 80
    targetPort: 3000
  type: LoadBalancer
  selector:
    k8s-app: grafana

```

5. Go back to the Deployment page after the successful deployment. Select the cluster from the Clusters drop-down list and then select kube-system from the Namespace drop-down list to view the deployed applications.

Deployment

Refresh Create by image Create by template

Clusters: k8s-cluster Namespace: kube-system

Name	Tag	PodsQuantity	Time Created	Action
alicloud-disk-controller	app:alicloud-disk-controller	1/1	06/27/2018,17:53:56	Details Edit Monitor More
alicloud-monitor-controller	k8s-app:alicloud-monitor-controller task:monitoring	1/1	06/28/2018,14:06:30	Details Edit Monitor More
default-http-backend	app:default-http-backend	1/1	06/27/2018,17:53:56	Details Edit Monitor More
heapster	k8s-app:heapster task:monitoring	1/1	06/27/2018,17:53:56	Details Edit Monitor More
kube-dns	k8s-app:kube-dns	1/1	06/27/2018,17:53:51	Details Edit Monitor More
monitoring-grafana	k8s-app:grafana task:monitoring	1/1	07/10/2018,16:04:34	Details Edit Monitor More

6. Click the name monitoring-grafana to view the deployment status. Wait until the running status changes to Running.

Deployment monitoring-grafana Back to List Refresh

Overview

Name:	monitoring-grafana
Namespace:	kube-system
Time Created:	2018-04-27 17:54:26
Label:	k8s-app:grafana task:monitoring
annotation:	deployment.kubernetes.io/revision:1
Selector:	k8s-app:grafana task:monitoring
Strategy:	RollingUpdate
Status:	Updated:1 , Unavailable:0 , Replica:1

Pods RelatedService

Name	Status	Image	Events
monitoring-grafana-675dc8448c-drfrq	Running	registry.cn-hangzhou.aliyuncs.com/acs/grafana:5.0.4	

7. Click Kubernetes > **Application** > > **Service** in the left-side navigation pane. Select the cluster from the Clusters drop-down list and kube-system from the Namespace drop-down list to view the external endpoint.

The external endpoint is automatically created by using the LoadBalancer type service. For developers who require more secure access policies, we recommend that you increase the security by adding the external endpoint to the IP whitelist or configuring the certificate.

Container Service - Kubernetes

Service List

Refresh

Create

Overview

Clusters

Clusters

Nodes

Volumes

Namespace

Application

Deployment

Pods

Service

Ingress

Clusters

k8s-cluster

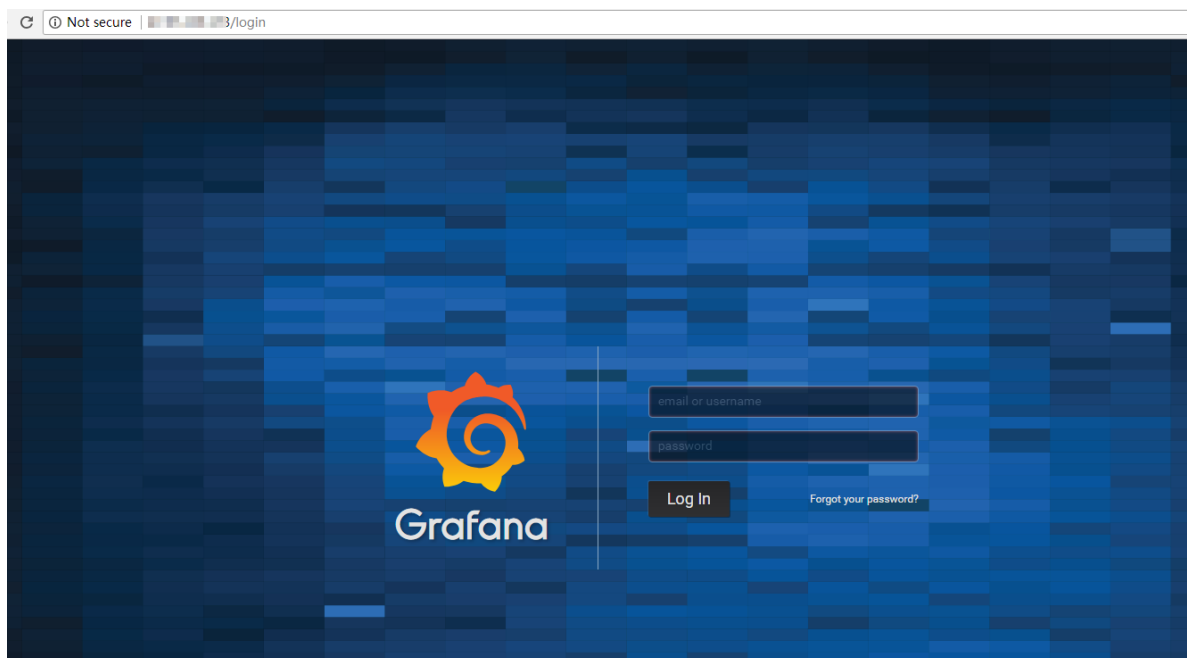
Namespace

kube-system

Name	Type	Time Created	ClustersIP	Internalendpoint	externalendpoint	Action
default-http-backend	ClusterIP	06/27/2018,17:53:56		default-http-backend:80 TCP	-	Details Update View YAML Delete
heapster	ClusterIP	06/27/2018,17:53:56		heapster:80 TCP	-	Details Update View YAML Delete
kube-dns	ClusterIP	06/27/2018,17:53:51		kube-dns:53 UDP kube-dns:53 TCP	-	Details Update View YAML Delete
monitoring-grafana	LoadBalancer	07/10/2018,16:04:34		monitoring-grafana:80 TCP monitoring-grafana:32746 TCP	:80	Details Update View YAML Delete
monitoring-influxdb	ClusterIP	06/27/2018,17:53:56		monitoring-influxdb:8086 TCP		Details Update View YAML Delete
nginx-ingress-lb	LoadBalancer	06/27/2018,17:53:56		nginx-ingress-lb:80 TCP nginx-ingress-lb:30883 TCP nginx-ingress-lb:443 TCP nginx-ingress-lb:32380 TCP	:80 :443	Details Update View YAML Delete

- Click the **external endpoint** at the right of the monitoring-grafana service to log on to the Grafana monitoring page.

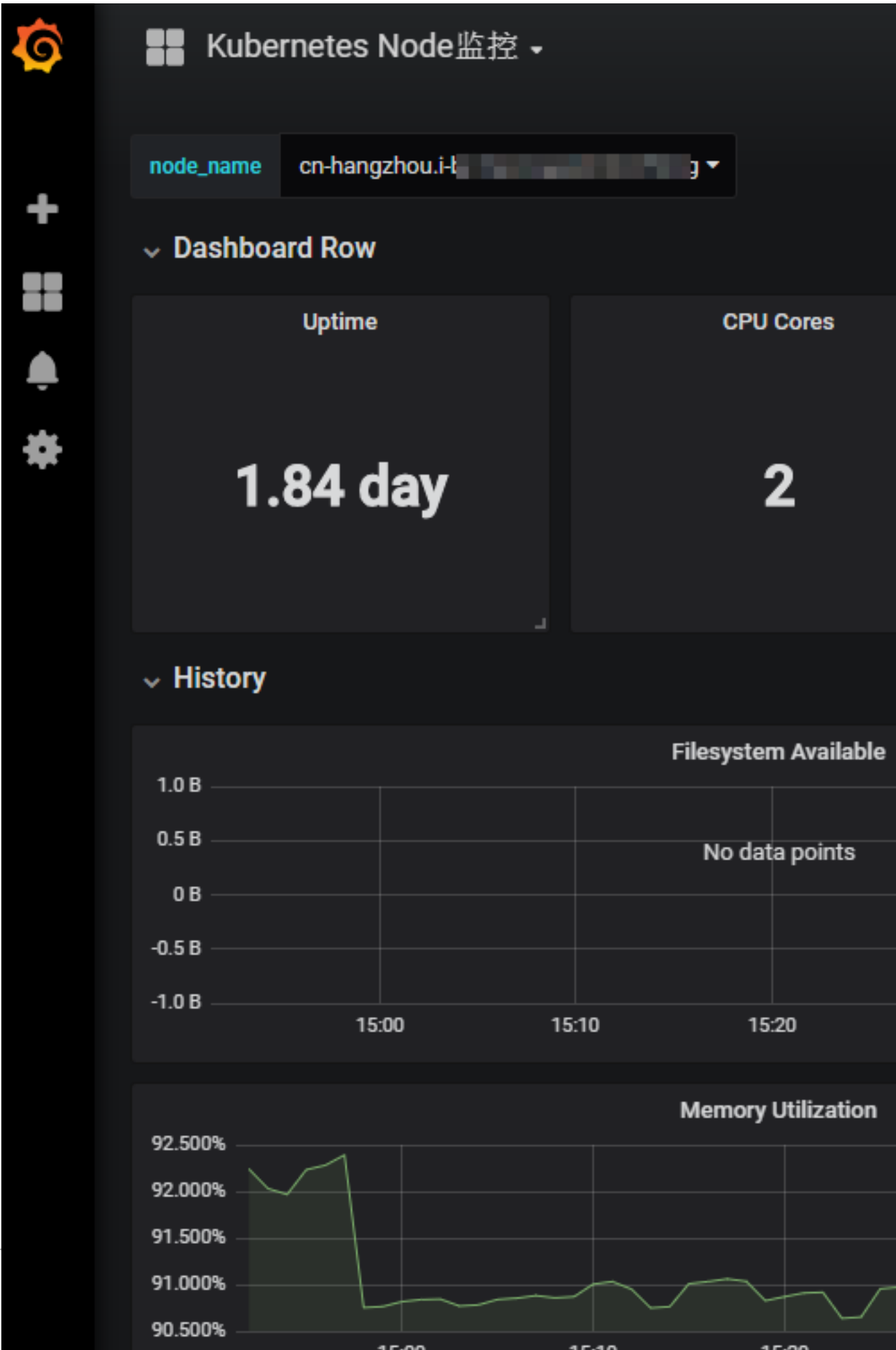
By default, the username and password of Grafana are both admin. We recommend that you change the password after the login.

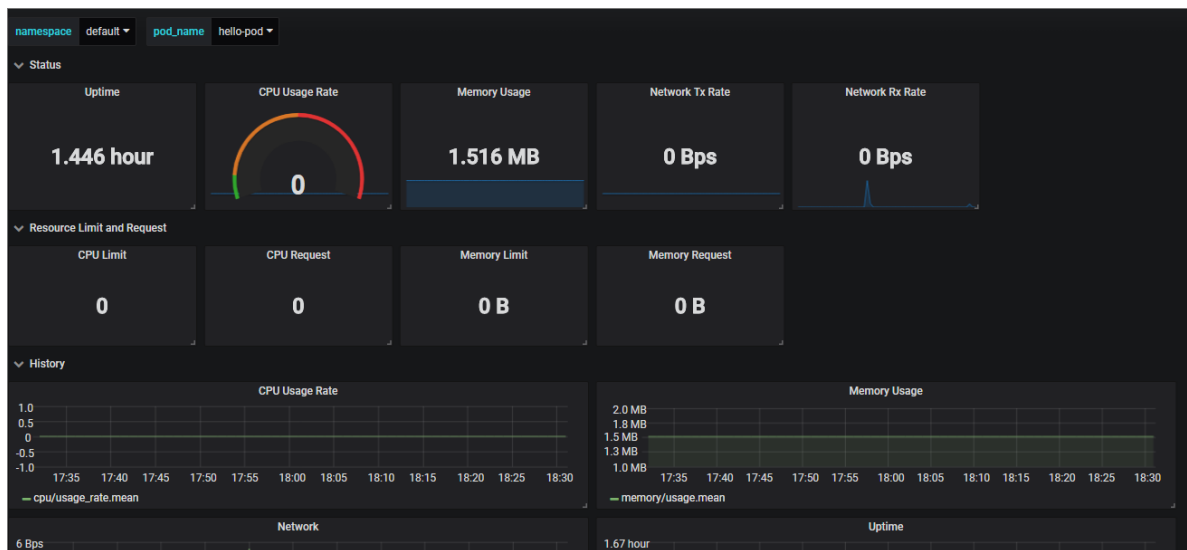


- Select the built-in monitoring templates to view the monitoring dashboards of the pod and node.

In this example, the Grafana has two built-in templates, one for displaying physical resources at the node level, and one for displaying resources related to the pod. Developers can also add

custom dashboards for more complicated display and send alarm notifications about resources based on Grafana.





1.11.3 Use an HPA auto scaling container

Alibaba Cloud Container Service supports the rapid creation of HPA-enabled applications on the console interface to achieve auto scaling of container resources. You can also configure it by defining the yaml configuration of Horizontal Pod Autoscaling (HPA).

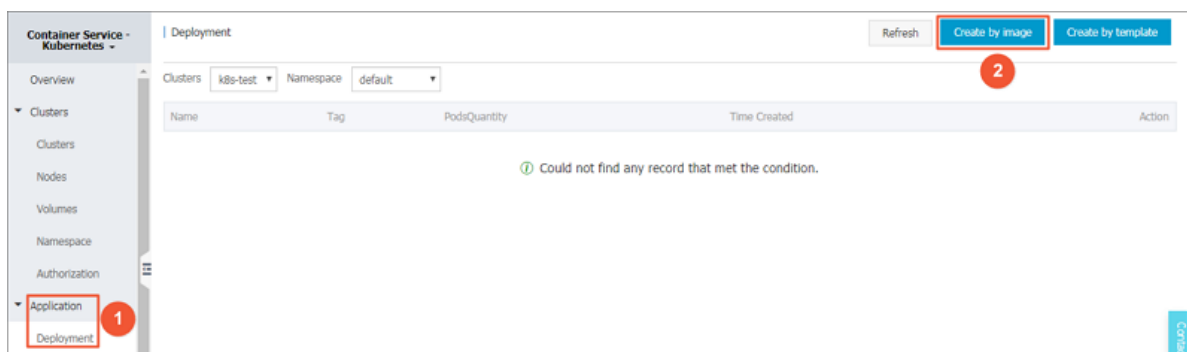
Prerequisites

- You have created a Kubernetes cluster. For more information, see [Create a Kubernetes cluster](#).
- You have successfully connected to the master node of the Kubernetes cluster.

Method 1 Create an HPA application in the Container Service console

In Alibaba Cloud Container Service, HPA has been integrated. You can easily create it through the Container Service console.

- Log on to the [Container Service console](#).
- Under Kubernetes, click **Application > Deployment** in the left-side navigation pane. Click **Create by image** in the upper-right corner.



3. Enter the application name, select the cluster and namespace, and click **Next**.
4. Configure the application settings. Set the number of replicas, select the **Enable** box for Automatic Scaling, and configure the settings for scaling.
 - **Metric:** CPU and memory. Configure a resource type as needed.
 - **Condition:** The percentage value of resource usage. The container begins to expand when the resource usage exceeds this value.
 - **Maximum Replicas:** The maximum number of replicas that the deployment can expand to.
 - **Minimum Replicas:** The minimum number of replicas that the deployment can contract to.

The screenshot shows the 'Configuration' tab of a deployment configuration page. It includes the following fields and values:

- Replicas:** 1
- Auto Scaling:** ☒ Enable
- Metric:** CPU Usage (selected from a dropdown)
- Condition:** Usage 50 %
- Maximum Replicas:** 10 (Range: 2-100)
- Minimum Replicas:** 1 (Range: 1-100)

5. Configure the container. Select an image and configure the required resources. Click **Next**.



Note:

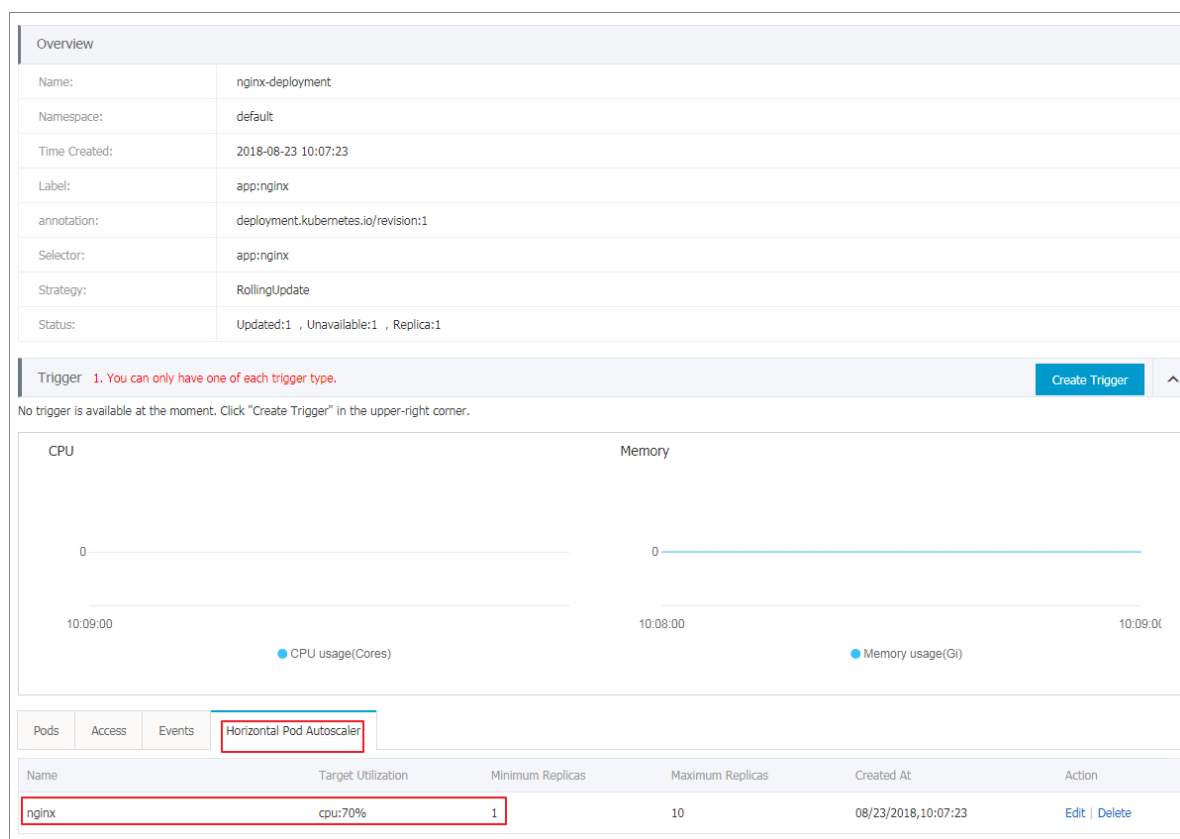
You must configure the required resources for the deployment. Otherwise, container auto scaling cannot be achieved.

The screenshot shows the 'container0' configuration page. It includes the following fields and values:

- Image Name:** nginx
- Image Version:** latest
- Always pull image:** ☐
- Resource Limit:** CPU 500m, Memory 128Mi
- Resource Request:** CPU 500m, Memory 128Mi
- Init Container:** ☐

6. In the Access Control page, do not configure any settings in this example. Click **Create** directly.

Now a deployment that supports HPA has been created. You can view the auto scaling group information in the details of your deployment.



7. In the actual environment, the application scales according to the CPU load. You can also verify auto scaling in the test environment. By performing a CPU pressure test on the pod, you can find that the pod can complete the horizontal expansion in half a minute.

Pods	Access	Events	Horizontal Pod Autoscaler
Name	Status	Image	
k8s-hpa-deployment-f87696b8b-jpr15	Running	nginx:latest	

Method 2 Use kubectl commands to configure container auto scaling

You can also manually create an HPA by using an orchestration template and bind it to the deployment object to be scaled. Use the `kubectl` command to complete the container auto scaling configuration.

The following is an example of an Nginx application. Execute the `kubectl create -f xxx.yaml` command to create an orchestration template for the deployment as follows:

```
apiVersion: apps/v1beta2 # for versions before 1.8.0 use apps/v1beta1
kind: Deployment
metadata:
  name: nginx
  labels:
    app: nginx
spec:
  replicas: 2
  selector:
```

```

    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
        - name: nginx
          image: nginx:1.7.9 # replace it with your exactly <image_name:
tags>
      ports:
        - containerPort: 80
      resources:
        requests:                                ##This parameter must be
configured. Otherwise, the HPA cannot operate.
        cpu: 500m

```

Create an HPA. Configure an object to which the current HPA is bound by using **scaleTargetRef**. In this example, the object is the deployment named nginx.

```

apiVersion: autoscaling/v2beta1
kind: HorizontalPodAutoscaler
metadata:
  name: nginx-hpa
  namespace: default
spec:
  scaleTargetRef:                                ##Bind the HPA to a
deployment named nginx
    apiVersion: apps/v1beta2
    kind: Deployment
    name: nginx
  minReplicas: 1
  maxReplicas: 10
  metrics:
    - type: Resource
      resource:
        name: cpu
        targetAverageUtilization: 50

```



Note:

The HPA needs to configure the request resource for the pod. The HPA does not operate without the request resource.

Warnings similar to the following are displayed when you execute **kubectl describe hpa [name]**:

```

Warning  FailedGetResourceMetric  2m (x6 over 4m)  horizontal-pod
-autoscaler  missing request for cpu on container nginx in pod default
/nginx-deployment-basic-75675f5897-mqzs7

```

```
Warning FailedComputeMetricsReplicas 2m (x6 over 4m) horizontal-pod-autoscaler failed to get cpu utilization: missing request for cpu on container nginx in pod default/nginx-deployment-basic-75675f5
```

After creating the HPA, execute the `kubectl describe hpa [name]` command again. You can see the following message, which indicates that the HPA is running normally.

```
Normal SuccessfulRescale 39s horizontal-pod-autoscaler New size: 1; reason: All metrics below target
```

When the usage of Nginx pod exceeds 50% set in this example, the container expands horizontally. When the usage of Nginx pod drops below 50%, the container contracts.

1.11.4 Monitor resources and send alarm notifications by using resource groups

Prerequisites

- [Create a Kubernetes cluster](#) if you do not have one.
- The Kubernetes version must be 1.8.4 or later. If the Kubernetes version of your cluster is earlier than 1.8.4, you must upgrade the cluster first and then upgrade the monitoring service to create a resource alarm group quickly.

Context

In the infrastructure Operation & Maintenance (O&M) of IT system, monitoring and alarm notifications guarantee the reliability and security, and facilitate the daily O&M, system monitoring, problem troubleshooting, and debugging.

For Kubernetes, the traditional container monitoring solution monitors resources and sends alarm notifications by using the statically configured monitoring agent or centralized server, which has a big problem. For example, containers are mostly scheduled in the resource pool and information required to identify the monitoring objects is missing if you deploy the monitoring agent on the host. Compared with traditional applications, the lifecycle of containers is shorter, and the abstract upper concepts of containers such as replica sets and deployments of Kubernetes do not have a good way to abstract the collected data reversely. This makes the container monitoring data cannot be effectively aggregated or be used to send alarm notifications. The original monitoring and alarm rules may not take effect if the application is released.

Alibaba Cloud Container Service Kubernetes clusters deeply integrate with cloud monitoring, abstract the logical concepts by using application groups, and unify the logical concepts and physical concepts on monitoring data and lifecycle. Besides, Alibaba Cloud cloud monitoring

service provides many features and custom tools, which allows you to monitor the Kubernetes resources and send alarm notifications.

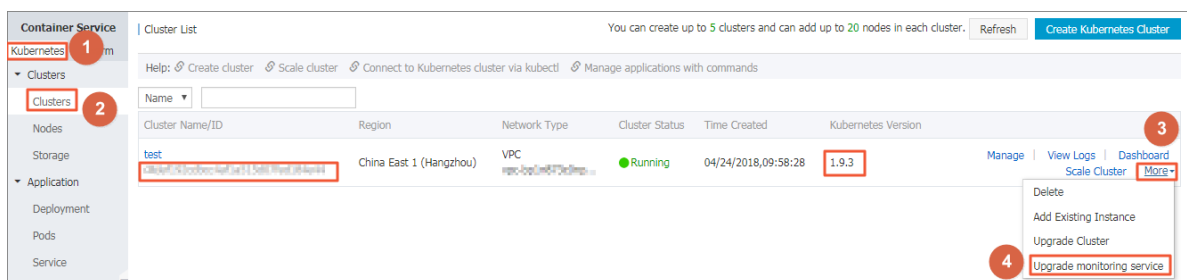
Procedure

1. Log on to the [Container Service console](#).
2. Under Kubernetes, click **Clusters** in the left-side navigation pane.
3. View the information such as Kubernetes version and cluster ID, and then click **More** > **Upgrade monitoring service** at the right of the cluster to create the resource alarm group quickly.

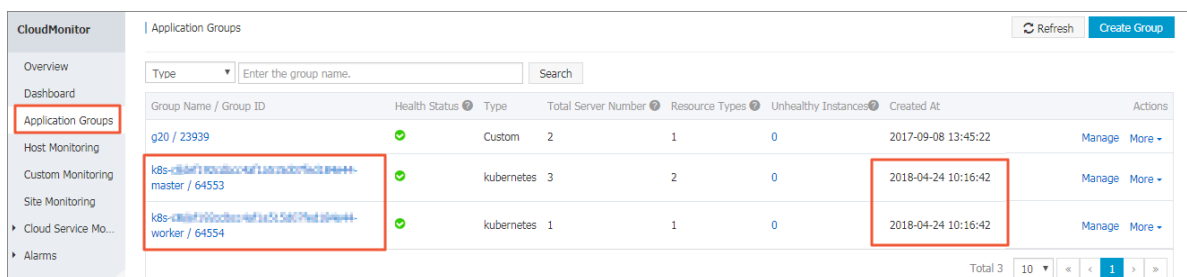


Note:

Click **Upgrade Cluster** to upgrade the cluster first if your Kubernetes version is earlier than 1.8.4.



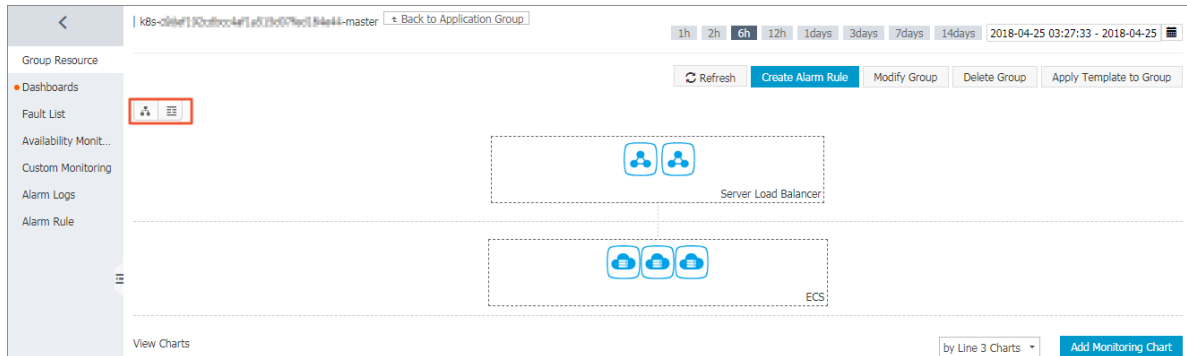
4. Log on to the [CloudMonitor console](#). In the left-hand navigation bar, click **Application Groups**. In Application Groups, you can see the Kubernetes resource groups that contains cluster ID information.



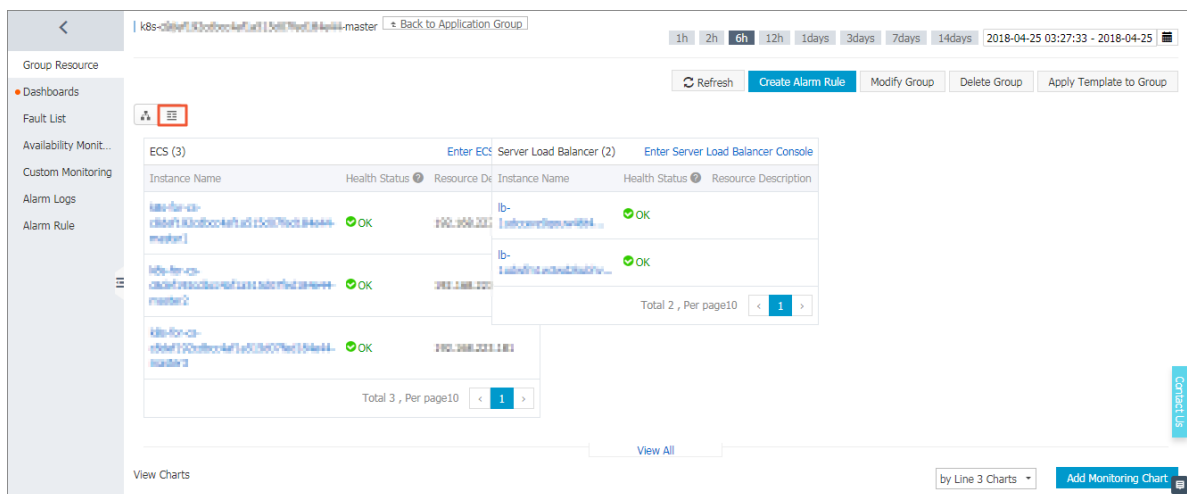
5. Click the **Group Name** to go to the detailed group page. You can view the monitoring views in the group. Take the Kubernetes master group as an example. The topology view of the group is displayed by default.

Kubernetes clusters have two kinds of nodes: worker nodes and master nodes. Master nodes generally have applications used to manage and control data deployed and the overall requirement on resources focuses on the strong robustness. Worker nodes are generally used

to schedule pods and the overall requirement on resources focuses on scheduling capability. Container Service automatically creates two resource groups, a master group and a worker group, when you create the resource alarm group. The master group includes the master nodes and the related Server Load Balancer instances. The worker group includes all the worker nodes.



6. To view the detailed machine information in the group, click the list view icon. Then, you can view the detailed information of each cloud product in the group.



7. Click **Add Monitoring Chart** in the lower-right corner to add more monitoring charts.

Add View

1 Chart Type

Line Area Heat Map Pie Chart

2 Select Metrics

Dashboards Log Custom

Monitoring Monitoring

ECS ECS Heat Map Gradient Range : 0

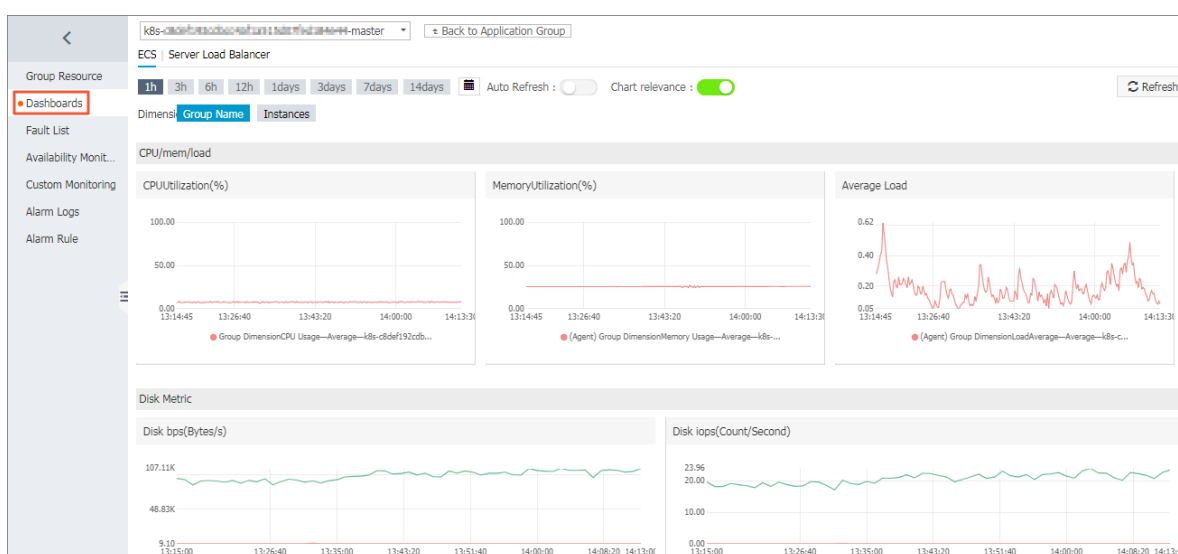
auto

No Data

Metrics : Heartbeat not Detected status

Resource : k8s-cbdf1932db-132dbf1932db-master/k8s-for-cs-cbdf1932db-132dbf1932db-master

- Click **Dashboards** in the left-side navigation pane to view the detailed monitoring metrics of each cloud product in the group.



9. Click **Alarm Rule** in the left-side navigation pane. The list of existing alarm rules in the group is displayed. By default, the health check to the core components of all nodes is set in the master group.

1. Click **Create Alarm Rule** to create an alarm rule for this group according to the business requirements.

The screenshot shows the CloudMonitor console interface. On the left, the 'Alarm Rule' menu item is highlighted with a red circle and the number 1. In the top right area, the 'Create Alarm Rule' button is highlighted with a red circle and the number 2. Below this, a table displays the list of existing alarm rules for the master group. All rules have a status of 'OK'.

Rule Name	Status (All)	Enable	Dimensions (All)	Alarm Rules	Product Name (All)	Notification Contact	Actions
kube-controller-TelnetStatus.Value	OK	Enabled	Group Dimension:k8s-c8def192cdccc4af1a515d07fed184e44-master	1minute Value>400 it alarms 3 times To alarm	CloudMonitor-Availability Monitoring	Default Co... View	Modify View Disable
kube-apiserver-TelnetStatus.Value	OK	Enabled	Group Dimension:k8s-c8def192cdccc4af1a515d07fed184e44-master	1minute Value>400 it alarms 3 times To alarm	CloudMonitor-Availability Monitoring	Default Co... View	Modify View Disable
kubelet-TelnetLatency.Average	OK	Enabled	Group Dimension:k8s-c8def192cdccc4af1a515d07fed184e44-master	1minute Average>10000 it alarms 3 times To alarm	CloudMonitor-Availability Monitoring	Default Co... View	Modify View Disable
kube-proxy-TelnetStatus.Value	OK	Enabled	Group Dimension:k8s-c8def192cdccc4af1a515d07fed184e44-master	1minute Value>400 it alarms 3 times To alarm	CloudMonitor-Availability Monitoring	Default Co... View	Modify View Disable
kube-schedule-TelnetLatency.Average	OK	Enabled	Group Dimension:k8s-c8def192cdccc4af1a515d07fed184e44-master	1minute Average>10000 it alarms 3 times To alarm	CloudMonitor-Availability Monitoring	Default Co... View	Modify View Disable
kubelet-TelnetStatus.Value	OK	Enabled	Group Dimension:k8s-c8def192cdccc4af1a515d07fed184e44-master	1minute Value>400 it alarms 3 times To alarm	CloudMonitor-Availability Monitoring	Default Co... View	Modify View Disable

2. The Create Alarm Rule page appears. Complete the configurations to create an alarm rule.

- Select the related resource, such as Elastic Compute Service (ECS).
- Select whether or not to use a template to create an alarm rule. If yes, select an existing alarm template from the Select Template drop-down list. You can also click **Create Alarm Template** to create a new custom alarm template. For more information, see .
- Configure the notification method (such as DingTalk Robot, email, and SMS) to obtain the Kubernetes cluster status in time.

Create Alarm Rule

Back to

1

Related Resource

Products :

k8s

Resource Range :

Application Group

Create Application Group Improving the efficiency of maintenance Best Practice

Group Name :

k8s-c2174629b4ea049d887da6e71d...

2

Set Alarm Rules

Use Template :

Yes No

Select Template :

常用基础模板

Create Alarm Template

常用基础模板_cpu_total

Host.cpu.total

1m

it alar...

>

90

%

常用基础模板_diskusage_utili

Host.disk.utilization

1m

it alar...

>

90

%

常用基础模板_memory_used

Host.mem.usedutilization L...

1m

it alar...

>

90

%

常用基础模板_InternetOutRat

Internet Outbound Bandwi...

1m

it alar...

>

90

%

常用基础模板_agent_heartbe

Heartbeat not Detected

1m

Mute for :

24h

Effective Period :

00:00

To:

23:59

3

Notification Method

Notification Contact :

Contact Group

All

Search

Quickly create a contact group

Selected Groups 1 count

All

云账号报警联系人

3. Click **Confirm** to create the alarm rule. The created alarm rule is displayed on the Alarm Rule page.

Back to Application Group

1h 2h 6h 12h 1days 3days 7days 14days 2018-04-25 19:26:17 - 2018-04-2

Group Resource

Enter the alarm rule name.

Search

Refresh

Create Alarm Rule

Modify Group

Delete Group

Apply Template to Group

Rule Name	Status (All)	Enable	Dimensions (All)	Alarm Rules	Product Name (All)	Notification Contact	Actions
test_CPUUtilization	OK	Enabled	Group Dimension:k8s-c8def192cdbcc4af1a515d07fed184e44-master	1minute CPU Usage Average>=90 % it alarms 1 times To alarm	ECS	Default Co...	View Modify Disable Delete
kube-controller-TelnetStatus.Value	OK	Enabled	Group Dimension:k8s-c8def192cdbcc4af1a515d07fed184e44-master	1minute Value>400 it alarms 3 times To alarm	CloudMonitor-Availability Monitoring	Default Co...	View Modify Disable

What's next

You can click the buttons in the left-side navigation pane to use other features that meet your resource monitoring requirements, such as fault list, event monitoring, availability monitoring, and log monitoring.

1.12 Manage a release

1.12.1 Manage a Helm-based release

The Alibaba Cloud Kubernetes service enriches the cloud marketplace, in which the app catalog and service catalog functions integrate with the Helm package management tool and allow you to build the application in the cloud quickly. A chart can be released multiple times, which requires you to manage the release versions. Therefore, the Alibaba Cloud Kubernetes service provides the release function, which allows you to manage the applications released by using Helm in the Container Service console.

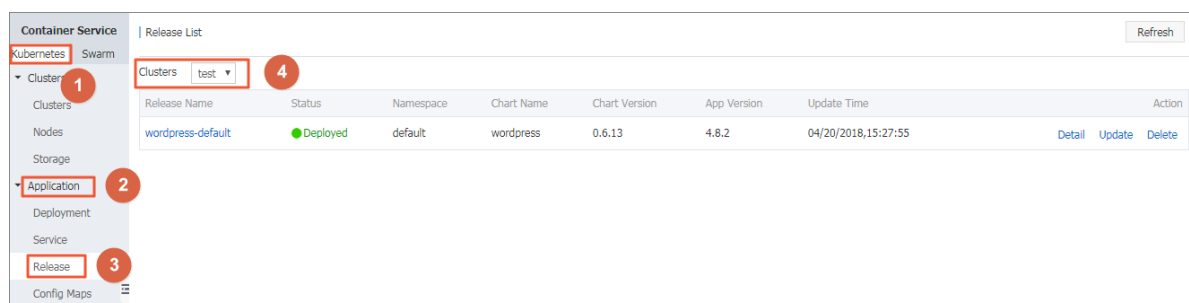
Prerequisites

- You have created a Kubernetes cluster successfully. For how to create a Kubernetes cluster, see [Create a Kubernetes cluster](#).
- You have used the app catalog function or service catalog function to install a Helm application. For more information, see [Simplify Kubernetes application deployment by using Helm](#). In this document, use the wordpress-default application as an example.

View release details

- Log on to the [Container Service console](#).
- Under Kubernetes, click **Application** > **Release** in the left-side navigation pane. Select the cluster from the Clusters drop-down list.

You can view the applications released by using the Helm package management tool and their services in the selected cluster.



- Take the wordpress-default as an example. Click **Detail** at the right of the release.

You can view the release details such as the current version and history version of this release (in this example, the current version is 1 and no history version exists). You can also view the resource information of wordpress-default, such as the resource name and resource type, and view the YAML information.

**Note:**

Click the resource name and you are redirected to the Kubernetes dashboard to view the detailed running status of this resource.

The screenshot shows the 'Container Service' console with the 'Release List' for 'wordpress-default'. The left sidebar shows the navigation menu with 'Application' > 'Release' selected. The main panel displays the 'Current Version' of the release, including the release name, namespace, and deployment time. Below this is a table listing the resources created by the release.

Resource	Kind	Values
wordpress-default-mariadb	Secret	View YAML
wordpress-default-wordpress	Secret	View YAML
wordpress-default-mariadb	ConfigMap	View YAML
wordpress-default-mariadb	PersistentVolumeClaim	View YAML
wordpress-default-wordpress	PersistentVolumeClaim	View YAML
wordpress-default-mariadb	Service	View YAML
wordpress-default-wordpress	Service	View YAML
wordpress-default-mariadb	Deployment	View YAML
wordpress-default-wordpress	Deployment	View YAML

4. Click the **Values** tab to view the parameter configurations for installing the Helm package.

The screenshot shows the 'Container Service' console with the 'Values' tab selected for the 'wordpress-default' release. The left sidebar shows the navigation menu with 'Application' > 'Release' selected. The main panel displays the 'Current Version' of the release, including the release name, namespace, and deployment time. Below this is a table listing the resources created by the release. The 'Values' tab is active, showing the Helm values for the release.

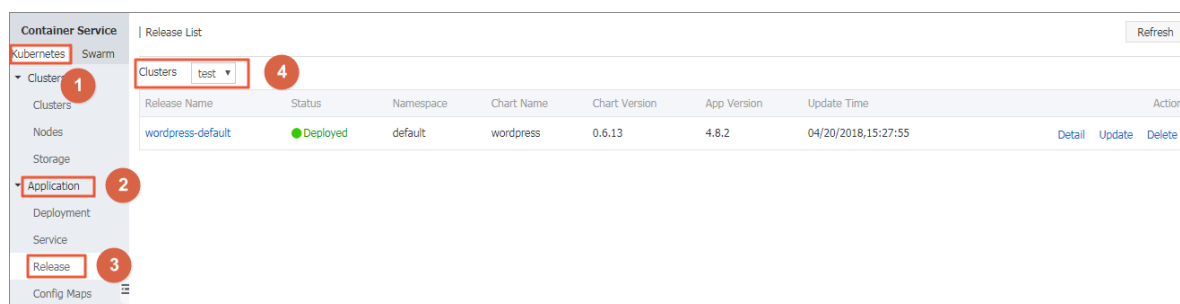
```

1 ## Bitnami WordPress image version
2 ## ref: https://hub.docker.com/r/bitnami/wordpress/tags/
3 ##
4 image: bitnami/wordpress:4.8.2-r0
5
6 ## Specify a imagePullPolicy
7 ## ref: http://kubernetes.io/docs/user-guide/images/#pre-pulling-images
8 ##
9 imagePullPolicy: IfNotPresent
10
11 ## User of the application
12 ## ref: https://github.com/bitnami/bitnami-docker-wordpress#environment-variables
13 ##
14 wordpressUsername: user
15
16 ## Application password
17 ## Defaults to a random 10-character alphanumeric string if not set
18 ## ref: https://github.com/bitnami/bitnami-docker-wordpress#environment-variables
19 ##
20 # wordpressPassword:
21
22 ## Admin email
23 ## ref: https://github.com/bitnami/bitnami-docker-wordpress#environment-variables
24 ##
25 wordpressEmail: user@example.com
  
```

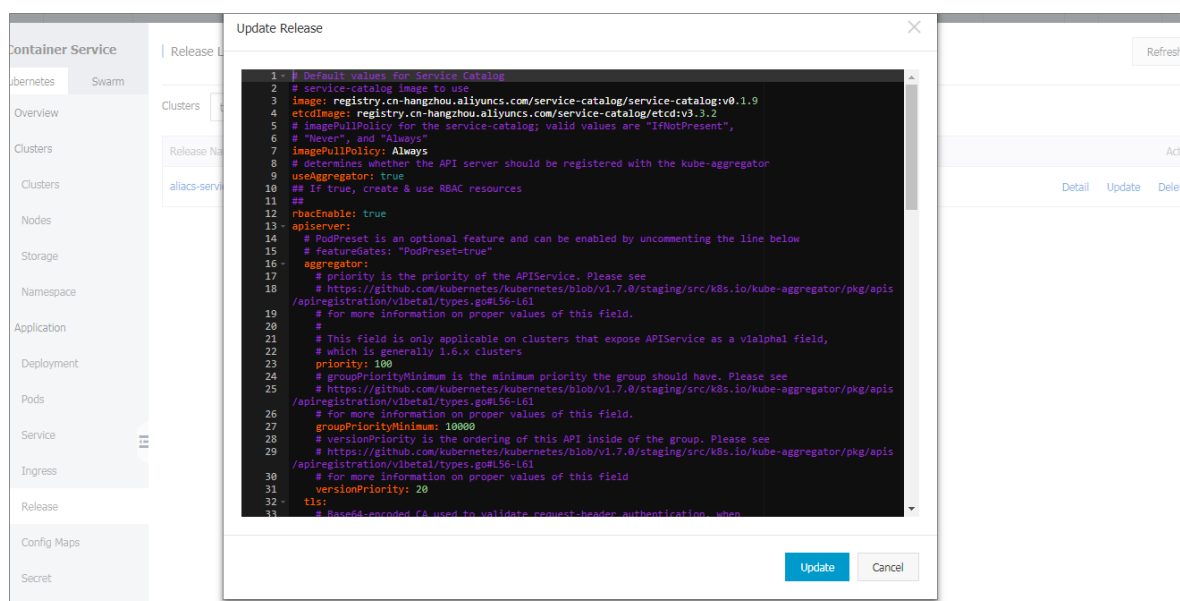
Update a release version

1. Log on to the [Container Service console](#).
2. Under Kubernetes, click **Application** > **Release** in the left-side navigation pane. Select the cluster from the Clusters drop-down list.

You can view the applications released by using the Helm package management tool and their services in the selected cluster.



3. Take the wordpress-default as an example. Click **Update** at the right of the release to update the release. The Update Release dialog box appears.



4. Modify the parameters and then click **Update**.

Update Release

```

1  ## Bitnami WordPress image version
2  ## ref: https://hub.docker.com/r/bitnami/wordpress/tags/
3  ##
4  image: bitnami/wordpress:4.8.2-r0
5
6  ## Specify a imagePullPolicy
7  ## ref: http://kubernetes.io/docs/user-guide/images/#pre-pulling-images
8  ##
9  imagePullPolicy: IfNotPresent
10
11 ## User of the application
12 ## ref: https://github.com/bitnami/bitnami-docker-wordpress#environment
13 ##
14 wordpressUsername: user
15
16 ## Application password
17 ## Defaults to a random 10-character alphanumeric string if not set
18 ## ref: https://github.com/bitnami/bitnami-docker-wordpress#environment
19 ##
20 # wordpressPassword:
21
22 ## Admin email
23 ## ref: https://github.com/bitnami/bitnami-docker-wordpress#environment
24 ##

```

Update

Cancel

The current version is changed to 2 and you can find version 1 under History Version. To roll back the release, click **Rollback**.

Current Version

Release Name : wordpress-default

Namespace : default

Deployed at : 04/20/2018,17:45:35

Current Version : 2

Time Updated : 04/20/2018,17:45:46

Resource	Kind	Values
wordpress-default-mariadb	Secret	View YAML
wordpress-default-wordpress	Secret	View YAML
wordpress-default-mariadb	ConfigMap	View YAML
wordpress-default-mariadb	PersistentVolumeClaim	View YAML
wordpress-default-wordpress	PersistentVolumeClaim	View YAML
wordpress-default-mariadb	Service	View YAML
wordpress-default-wordpress	Service	View YAML
wordpress-default-mariadb	Deployment	View YAML
wordpress-default-wordpress	Deployment	View YAML

History Version

Version : 1

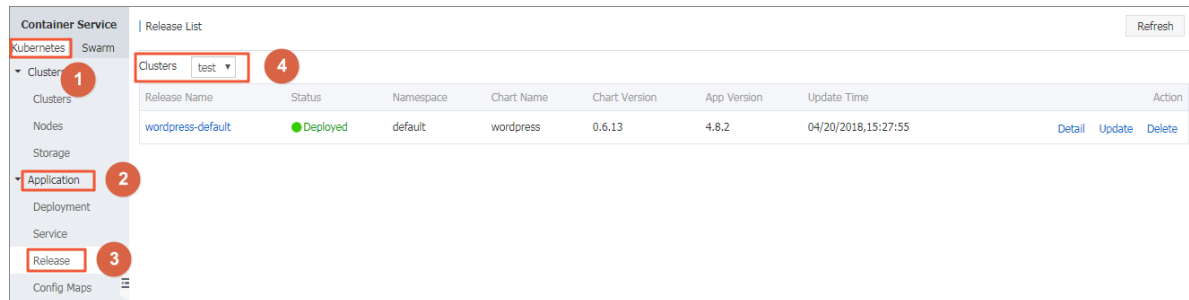
[Rollback](#)

Time Updated : 04/20/2018,17:45:35

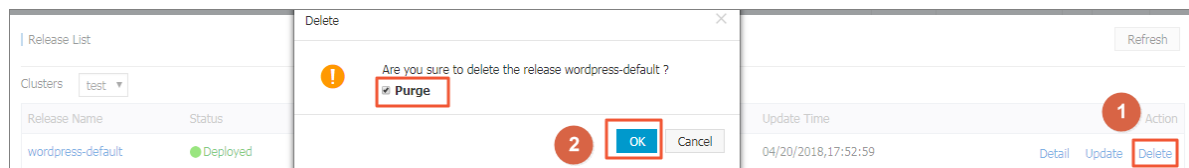
Delete a release

1. Log on to the [Container Service console](#).
2. Under Kubernetes, click **Application** > **Release** in the left-side navigation pane. Select the cluster from the Clusters drop-down list.

You can view the applications released by using the Helm package management tool and their services in the selected cluster.



3. Take the wordpress-default as an example. Click **Delete** at the right of the release to delete the release. The Delete dialog box appears.



4. Select the **Purge** check box to clear the release records if necessary. Click **OK** to delete the wordpress-default application and its resources such as the services and deployments.

1.12.2 Use batch release on Alibaba Cloud Container Service for Kubernetes

You can use Alibaba Cloud Container Service for Kubernetes to release application versions in batches, achieving fast version verification and rapid iteration of applications.

Context



Note:

The latest Kubernetes cluster has installed alicloud-application-controller by default. For older versions of clusters, only versions of 1.9.3 and later are currently supported, and you can upgrade old versions of clusters through the prompt link on the console.

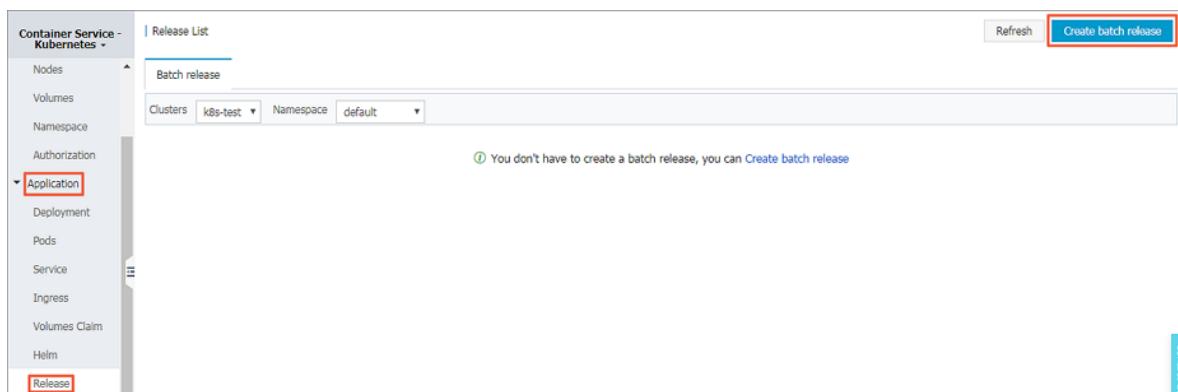
Procedure

1. Log on to the [Container Service console](#).

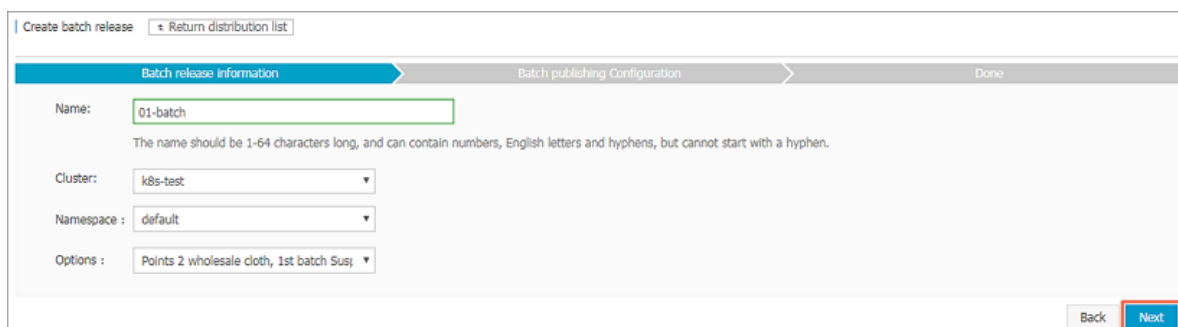
2. Under Kubernetes, click **Application** > **Release** in the left-side navigation pane. Click **Create batch release** in the upper-right corner.

**Note:**

If the button is gray, you can upgrade the cluster by following the upgrade link.



3. Configure batch release information, including the application name, cluster, namespace, and options. Click **Next**.



4. On the batch publishing configuration page, configure the backend pod and service, and then click **Update** to create an application.

Batch release information | **Batch publishing Configuration** | Done

The wizard mode

General

Image Name: Image Version: ☐ Always pull image

Scale:

Resource Limit: CPU: Memory:

Resource Request: CPU: Memory:

Access Control

Service: ☐ ClusterIp ☐ NodePort ☒ Server Load Balancer Load Balancer will be created

Port Mapping:

Port	target port	Protocol
<input type="text" value="80"/>	<input type="text" value="80"/>	<input type="text" value="TCP"/>

[Add](#)

[Prev](#) [Update](#)

Overview

- General
- Access Control
- Container
- Volume
- Environment

[Advance Config](#)

5. Return to the release list, an application is displayed in the **Not started** status. Click **Detail** on the right.

Release List Refresh Create batch release

Batch release

Clusters: Namespace:

Release Name	Namespace	Update Time	Status	Action
01-batch	default	2018-09-03 16:00:56	Not started	Detail Update Delete

6. On the application detail page, you can view more information. Click **Change Configuration** in the upper-right corner of the page to make a batch release change.

Release List - nginx Refresh Change Configuration

Details **History**

Overview

Release Name:	nginx
Release Type:	Batch Release
Created At:	2018-09-28 16:24:04
RelatedService:	batchrelease-nginx-svc
Status:	Not Started

Not Started **In Progress** **Completed** Refresh Continue Roll Back Complete

Name	App Version	Status	Pod IP	Time Created	Action
batchrelease-nginx-0	v1	Running	172.16.1.159	09/28/2018,16:24:04	Terminal Logs
batchrelease-nginx-1	v1	Running	172.16.1.160	09/28/2018,16:24:07	Terminal Logs
batchrelease-nginx-2	v1	Running	172.16.1.161	09/28/2018,16:24:10	Terminal Logs
batchrelease-nginx-3	v1	Running	172.16.1.162	09/28/2018,16:24:12	Terminal Logs

[Contact Us](#)

7. Configure changes for the new version of the application, and then click **Update**.

8. By default, you return to the release list page, where you can view the batch release status of the application. After completing the first deployment, click **Detail**.

Release Name	Namespace	Update Time	Status	Action
nginx	default	2018-09-28 16:24:04	Wait for Deployment to Complete (Total Batches: 2. Currently, batch 0 is in process and the batch status is Deploying)	Details Update Delete

9. You can see that the Not Started list is has two pods and the Completed list has two pods, which indicates that the first batch has been completed in batch release. Click **Continue**, you can release the second batch of pods. Click **Roll Back** to roll back to the previous version.

Name	App Version	Status	Pod IP	Time Created	Action
batchrelease-nginx-2	v2	Running	172.16.1.164	09/28/2018,16:26:57	Terminal Logs
batchrelease-nginx-3	v2	Running	172.16.1.163	09/28/2018,16:26:53	Terminal Logs

10. When completing the release, click **History** to roll back to history versions.



What's next

You can use batch release to quickly verify your application version without traffic consumption. Batch release is more resource-saving than blue-green release. Currently, batch release can be performed on only web pages. The yaml file editing is to be opened later to support more complex operations.

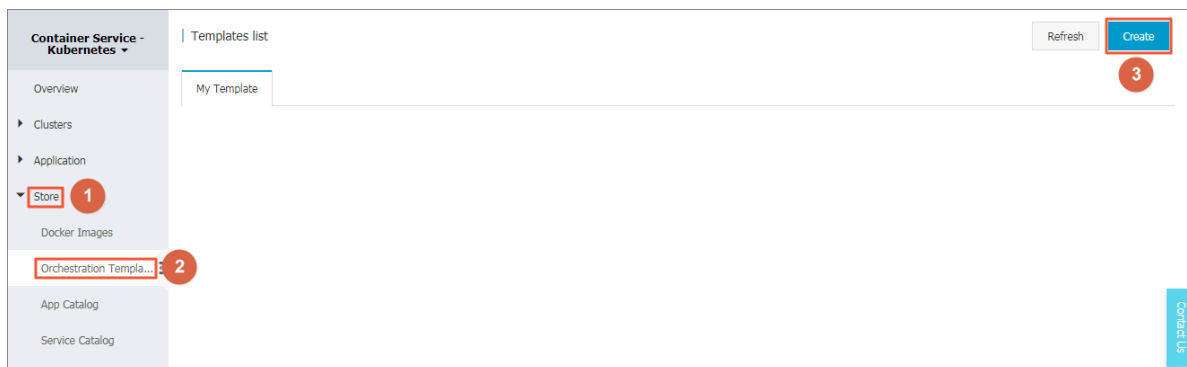
1.13 Template

1.13.1 Create an orchestration template

You can use multiple methods to create orchestration templates through the Container Service console.

Procedure

1. Log on to the [Container Service console](#).
2. Under Kubernetes, click **Store > Orchestration Templates** in the left-side navigation pane. Click **Create** in the upper-right corner.



3. In the displayed dialog box, configure the orchestration template, and then click **Save**. In this example, build a tomcat application template that contains a deployment and a service.
 - **Name:** Set the template name.
 - **Description:** Enter the description for the template. This parameter is optional.
 - **Template:** Configure the template that conforms to Kubernetes yaml syntax rules. The template can contain multiple resource objects that are separated by `---`.

Create

Name:

tomcat

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

Description:

tomcat application

Template:

```

1  apiVersion: apps/v1beta2 # for versions before 1.8.0 use
   apps/v1beta1
2  kind: Deployment
3  metadata:
4    name: tomcat-deployment
5    labels:
6      app: tomcat
7  spec:
8    replicas: 1
9    selector:
10   matchLabels:
11     app: tomcat
12   template:
13     metadata:
14       labels:
15         app: tomcat
16     spec:
17       containers:
18       - name: tomcat
19         image: tomcat # replace it with your exactly
20         <image_name:tags>
21         ports:
22         - containerPort: 8080

```

Save

Cancel

4. After the template is created, the **Template List** page is displayed. You can see the template under **My template**.

Templates list

Refresh Create

My Template

tomcat

tomcat application

Details

Resource Type / Resource Name

Deployment: tomcat-deployment

Service: tomcat-svc

Create Application ➔

5. Optional: You can also click **Application > Deployment** in the left-hand navigation pane, and click **Create by template** to enter the **Deploy templates** page. Save one of orchestration templates built-in Container Service as your custom template.
- a) Select a built-in template and click **Save Template**.

Clusters: test-sls

Namespace: default

Resource Type: Resource - basic Deployment

Template:

```

1 apiVersion: apps/v1beta2 # for versions before 1.8.0 use apps/v1beta1
2 kind: Deployment
3 metadata:
4   name: nginx-deployment-basic
5   labels:
6     app: nginx
7 spec:
8   replicas: 2
9   selector:
10    matchLabels:
11      app: nginx
12   template:
13     metadata:
14       labels:
15         app: nginx
16     spec:
17       # nodeSelector:
18       #   env: test-team
19     containers:
20     - name: nginx
21       image: nginx:1.7.9 # replace it with your exactly <image_name:tags>
22       ports:
23       - containerPort: 80
  
```

Add Deployment
Deploy with exist template

Save Template DEPLOY

- b) In the displayed dialog box, configure the name, description, and template. After completing the configurations, click **Save**.



Note:

You can modify the built-in template.

- c) Click **Store** > **Orchestration Template**, the created template is displayed under **My Template**.

Templates list			Refresh	Create
My Template				
	tomcat tomcat application Details	Resource Type / Resource Name Deployment: tomcat-deployment Service: tomcat-svc	Create Application ➔	
	nginx Details	Resource Type / Resource Name Deployment: nginx-deployment-basic	Create Application ➔	

What's next

You can quickly create an application by using the orchestration template under **My Template**.

1.13.2 Edit an orchestration template

You can edit an orchestration arrangement template.

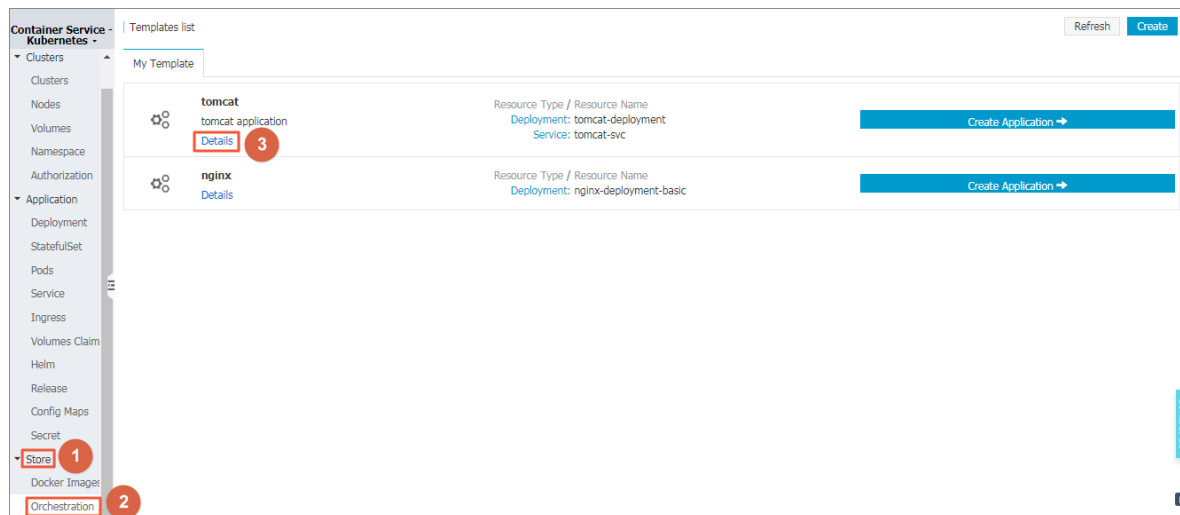
Prerequisites

You have created an orchestration template, see [Create an orchestration template](#).

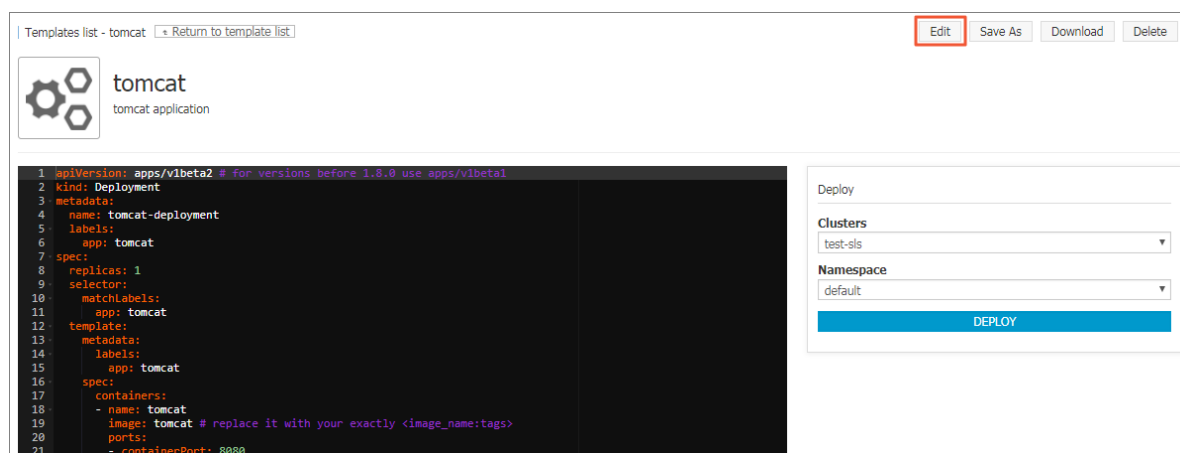
Procedure

1. Log on to the [Container Service console](#).

2. Under Kubernetes, click **Store > Orchestration Templates**. Existing orchestration templates are displayed under **My Template**.
3. Select a template and click **Details**.



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



5. In the displayed dialog box, edit the name, description, and template, and click **Save**.

Modify template

Name:

tomcat-V2

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

Description:

tomcat application

Template:

```

1  apiVersion: apps/v1beta2 # for versions
   before 1.8.0 use apps/v1beta1
2  kind: Deployment
3  metadata:
4    name: tomcat-deployment
5    labels:
6      app: tomcat
7  spec:
8    replicas: 1
9    selector:
10   matchLabels:
11     app: tomcat
12   template:
13     metadata:
14       labels:
15         app: tomcat
16     spec:
17       containers:
18       - name: tomcat
19         image: tomcat # replace it with your
   exactly <image_name:tags>
20       ports:
21       - containerPort: 8080
22

```

Save

Cancel

6. Back to the **Template List** page, under **My Template**, you can see the template is changed.

Templates list		Refresh	Create
My Template			
	tomcat-V2	Resource Type / Resource Name	Create Application →
	tomcat application Details	Deployment: tomcat-deployment Service: tomcat-svc	
	nginx	Resource Type / Resource Name	Create Application →
	Details	Deployment: nginx-deployment-basic	

1.13.3 Save an existing orchestration template as a new one

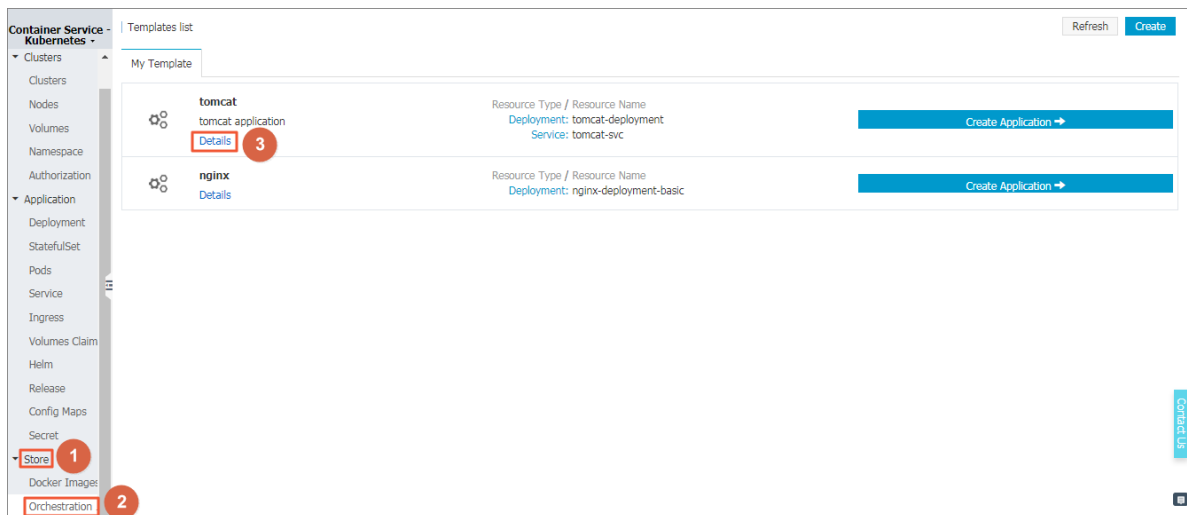
You can save an existing template as a new one.

Prerequisites

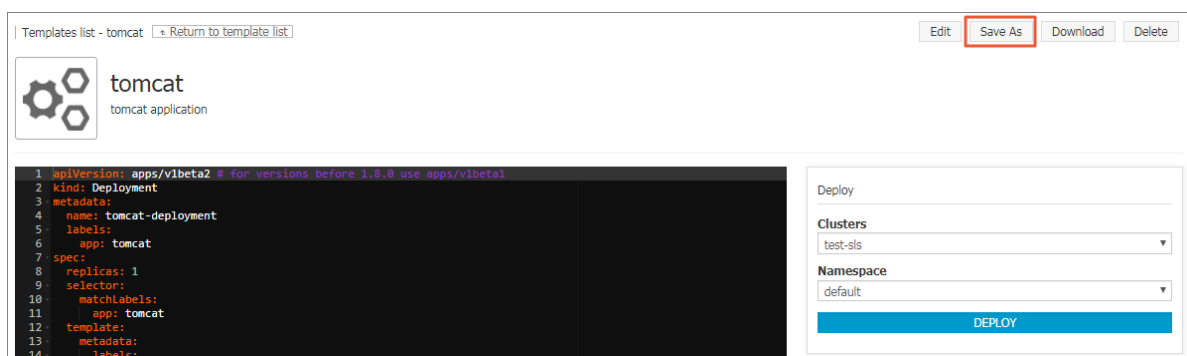
You have created an orchestration template, see [Create an orchestration template](#).

Procedure

1. Log on to the [Container Service console](#).
2. Under Kubernetes, click **Store > Orchestration Templates**. Existing orchestration templates are displayed under **My Template**.
3. Select a template and click **Details**.



4. You can modify the template and click **Save as** in the upper-right corner.



5. In the displayed dialog box, configure the template name and click **OK**.

Save Template As

Name:

tomcat-V3

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

OK

Cancel


6. Back to the **Template List** page, you can see that the saved template is displayed under **My Template**.

Templates list

Refresh

Create

My Template



tomcat-V3
tomcat application
[Details](#)

Resource Type / Resource Name
[Deployment](#): tomcat-deployment
[Service](#): tomcat-svc

Create Application →

1.13.4 Download an orchestration template

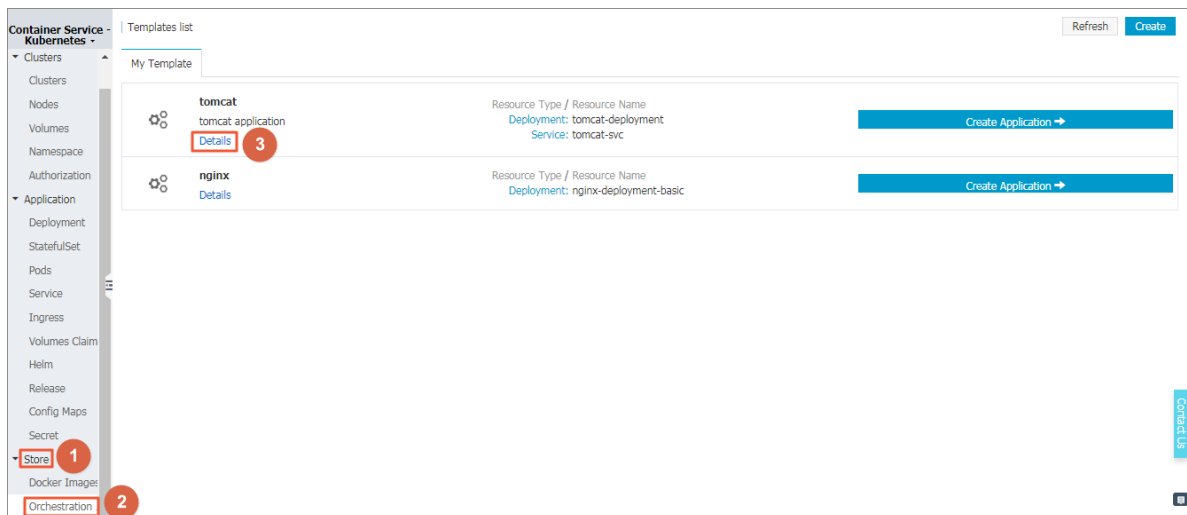
You can download an existing orchestration template.

Prerequisites

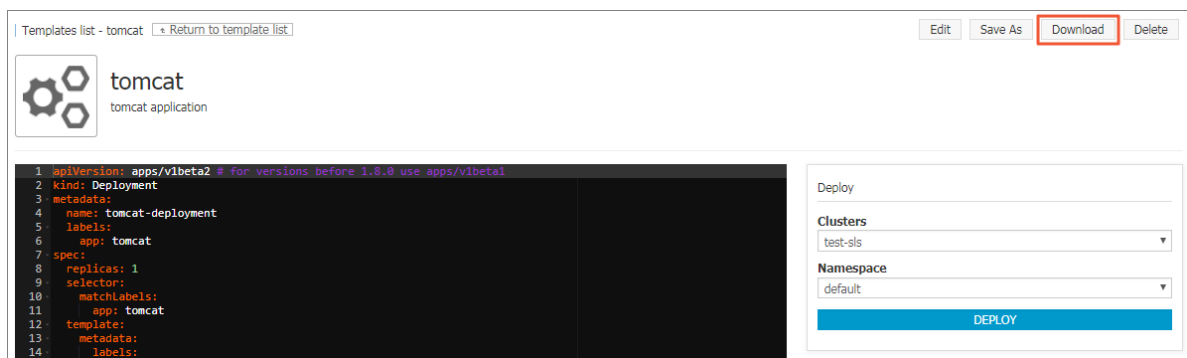
You have created an orchestration template, see [Create an orchestration template](#).

Procedure

1. Log on to the [Container Service console](#).
2. Under Kubernetes, click **Store > Orchestration Templates**. Existing orchestration templates are displayed under **My Template**.
3. Select a template and click **Details**.



4. Click **Download** in the upper-right corner, a template file with yml suffix is downloaded immediately.



1.13.5 Delete an orchestration template

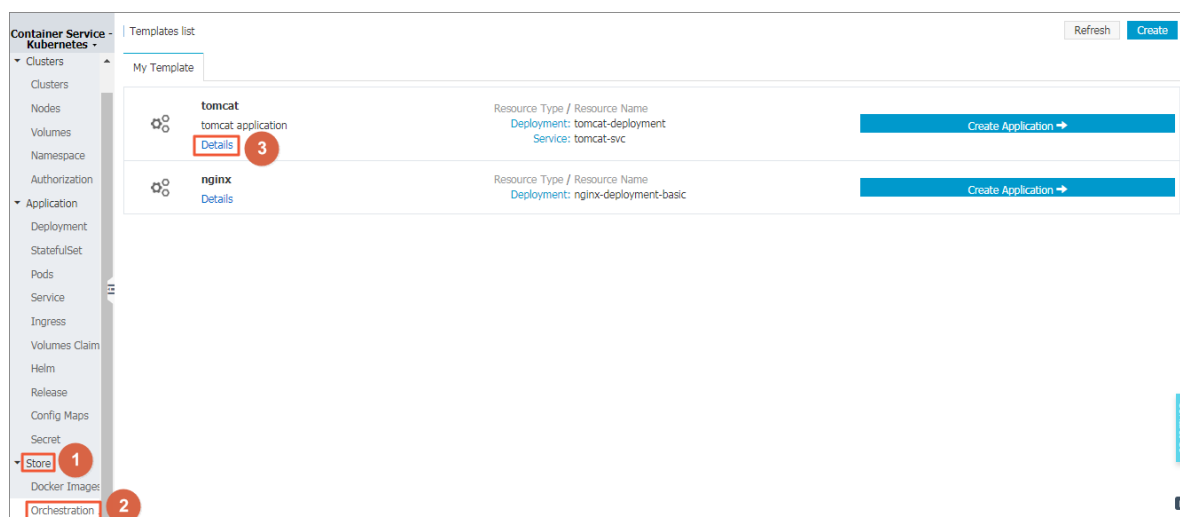
You can delete an orchestration template that is no longer needed.

Prerequisites

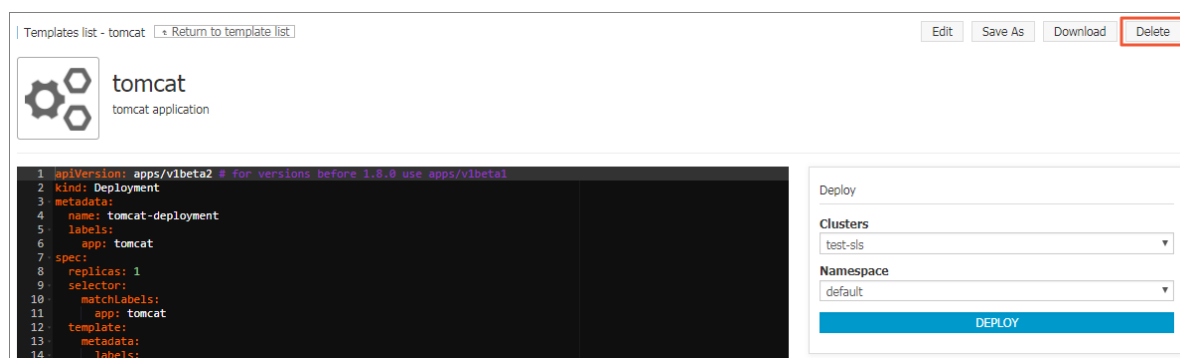
You have created an orchestration template, see [Create an orchestration template](#).

Procedure

1. Log on to the [Container Service console](#).
2. Under Kubernetes, click **Store > Orchestration Template**. Existing orchestration templates are displayed under **My Template** on the **Template list** page.
3. Select a template and click **Detail**.



4. On the detail page of the template, you can click **Delete** in the upper-right corner.



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

1.14 App catalog

1.14.1 App catalog overview

Microservice is the theme of container era. The application microservice brings great challenge to the deployment and management. By dividing a large single application into several microservices, the microservice can be independently deployed and extended so as to realize the agile development and fast iteration. Microservice brings great benefits to us. However, developers have to face the management issues of the microservices, such as the resource management, version management, and configuration management. The number of microservices is large because an application is divided into many components that correspond to many microservices.

For the microservice management issues under Kubernetes orchestration, Alibaba Cloud Container Service introduces and integrates with the Helm open-source project to help simplify the deployment and management of Kubernetes applications.

Helm is an open-source subproject in the Kubernetes service orchestration field and a package management tool for Kubernetes applications. Helm supports managing and controlling the published versions in the form of packaging softwares, which simplifies the complexity of deploying and managing Kubernetes applications.

Alibaba Cloud app catalog feature

Alibaba Cloud Container Service app catalog feature integrates with Helm, provides the Helm-related features, and extends the features, such as providing graphic interface and Alibaba Cloud official repository.

The chart list on the App Catalog page includes the following information:

- Chart name: A Helm package corresponding to an application, which contains the image, dependencies, and resource definition required to run an application.
- Version: The version of the chart.
- Repository: The repository used to publish and store charts, such as the official repository stable and incubator.

The information displayed on the details page of each chart may be different and include the following items:

- Chart introduction
- Chart details
- Prerequisites for installing chart to the cluster, such as pre-configuring the persistent storage volumes (pv)
- Chart installation commands
- Chart uninstallation commands
- Chart parameter configurations

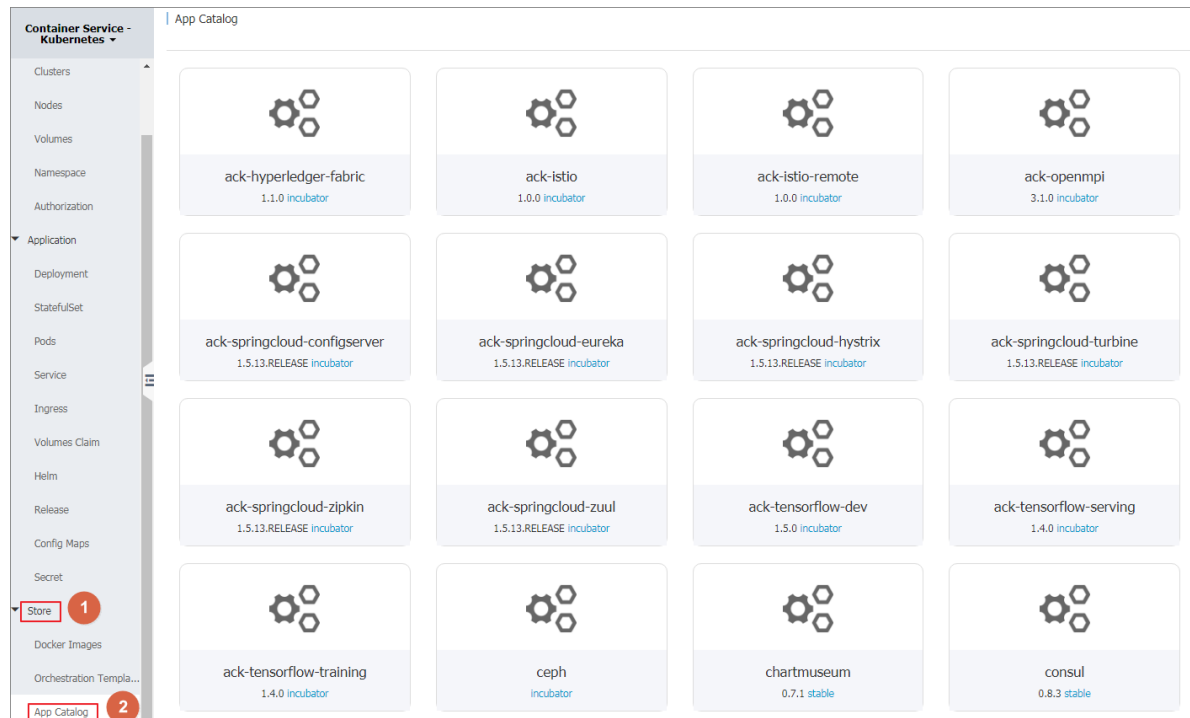
Currently, you can deploy and manage the charts in the app catalog by using the Helm tool. For more information, see [Simplify Kubernetes application deployment by using Helm](#).

1.14.2 View app catalog list

Procedure

1. Log on to the [Container Service console](#).
2. Under Kubernetes, click **Store > App Catalog** in the left-side navigation pane.

View the charts on the App Catalog page, each of which corresponds to an application, containing some basic information such as the application name, version, and source repository.



What's next

You can click to enter a chart and get to know the detailed chart information. Deploy the application according to the corresponding information by using the Helm tool. For more information, see [Simplify Kubernetes application deployment by using Helm](#).

1.15 Service catalog

1.15.1 Overview

Applications running on the cloud platform need some basic services such as databases, application servers, and other generic basic softwares. For example, a WordPress application, as a Web application, needs a database service (such as MariaDB) in the backend. Traditionally, you can create the MariaDB service on which the application depends in the WordPress application orchestration, and integrate the MariaDB service with the Web application. To develop applications on the cloud in this way, developers must spend time and energy deploying and configuring the dependent infrastructure softwares, which increases the costs of hosting and migrating applications.

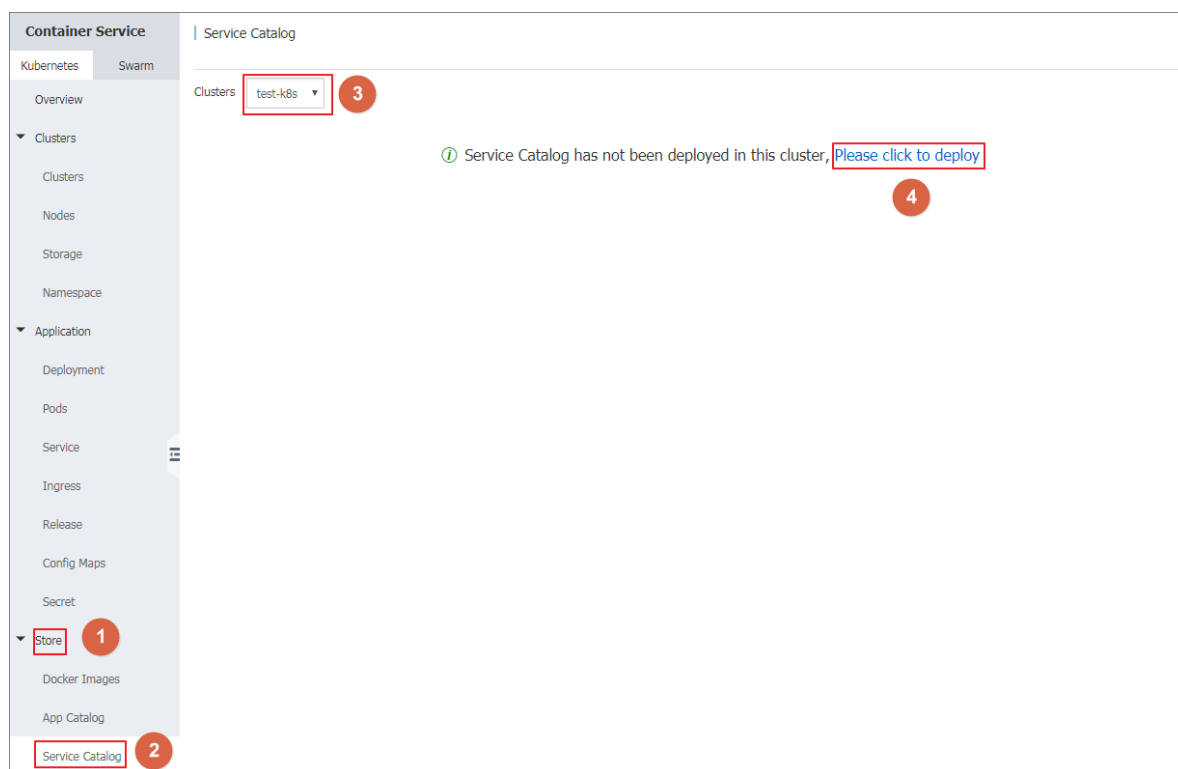
Alibaba Cloud Container Service supports and integrates with the service catalog function. The service catalog function aims to access and manage the service brokers, which allows applications running in Kubernetes clusters to use the managed services offered by service brokers. A series of infrastructure softwares are supported by the service catalog function, which allows the developers to use these softwares as services and focus on the applications, the core of the development, without concerning about the availability and scalability of the softwares or managing the softwares.

The service catalog uses the Open service broker API of Kubernetes to communicate with service brokers, acting as an intermediary for the Kubernetes API server to negotiate the initial provisioning and obtain the credentials necessary for the applications to use the managed services. For more information about the implementation principle of the service catalog, see [Service catalog](#).

1.15.2 Enable service catalog function

Procedure

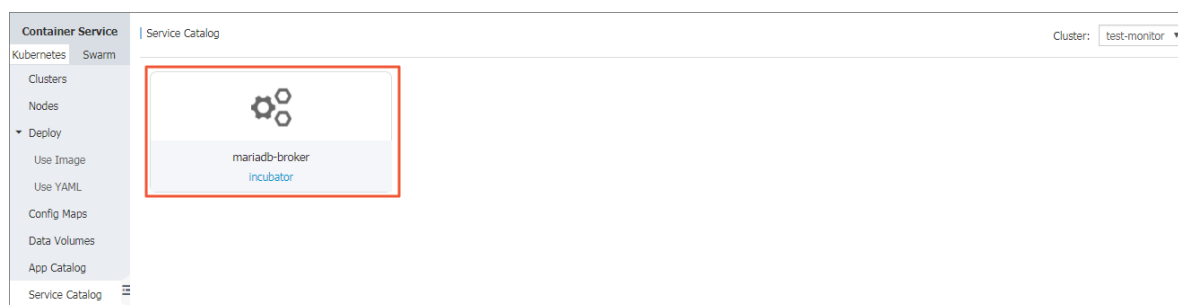
1. Log on to the [Container Service console](#).
2. Click Kubernetes > **Store** > > **Service Catalog in the left-side navigation pane**. Select the cluster from the Cluster drop-down list in the upper-right corner.
3. If you have not deployed the service catalog, click to install the service catalog as instructed on the page.



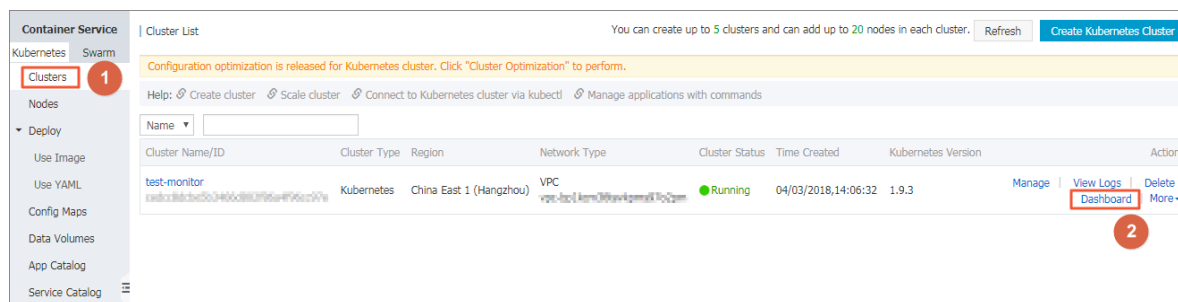
4. After the installation, the service broker, which is installed by default, is displayed on the Service Catalog page. You can click the mariadb-broker to view the details.

**Note:**

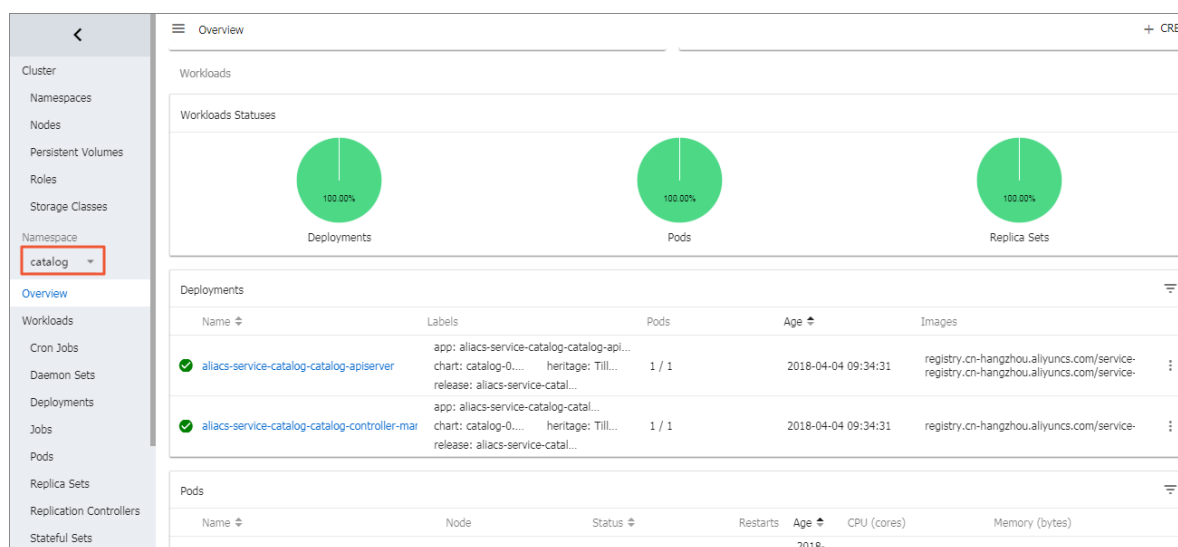
The service catalog is implemented as an extension API server and a controller. After Alibaba Cloud Container Service installs the service catalog function, the namespace catalog is created.



5. Click **Clusters** in the left-side navigation pane. Click **Dashboard** at the right of a cluster.



6. In the Kubernetes dashboard, select `catalog` as the Namespace in the left-side navigation pane. You can see the resource objects related to catalog apiserver and controller are installed under this namespace.



What's next

Then, you have successfully enabled the service catalog function. You can create a managed service by using the service broker in the service catalog, and apply the managed service to your applications.