ALIBABA CLOUD

Alibaba Cloud

Container Service for Kubernetes

Quick Start

Document Version: 20211202

Alibaba Cloud

Legal disclaimer

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

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

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

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

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

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

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

Document conventions

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

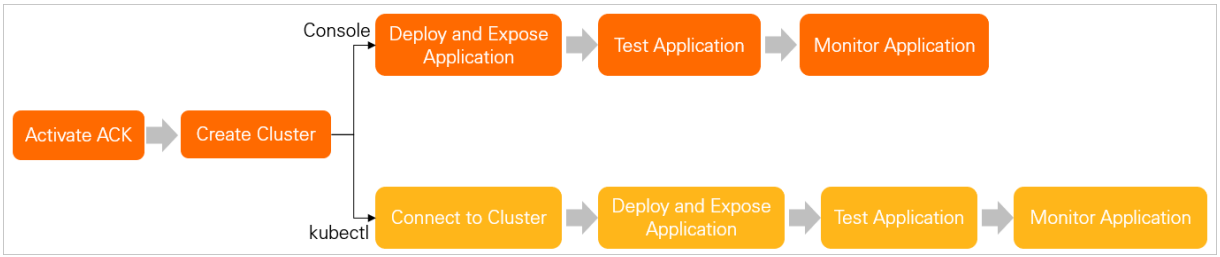
Table of Contents

1.Quick start overview

Container Service for Kubernetes (ACK) provides high-performance management services for containerized applications. You can use ACK to manage containerized applications that run on the cloud in a convenient and efficient manner. This topic describes how to use ACK and the ACK documentation, and provides answers to some frequently asked questions about ACK. This helps you quickly get started with ACK.

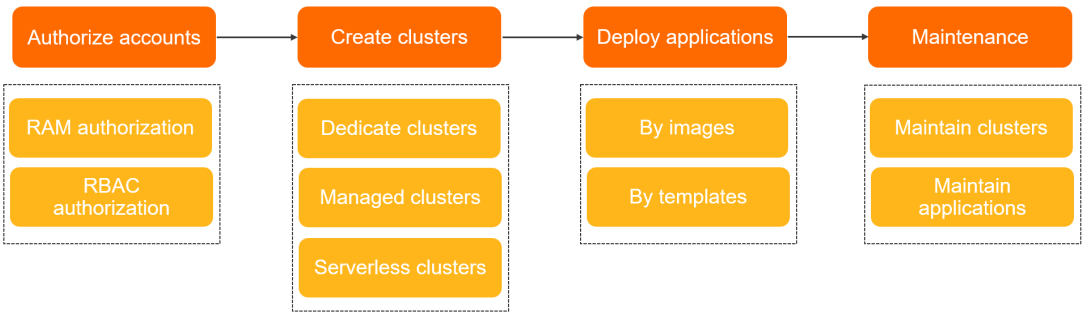
How to use ACK

The following figure shows how to use ACK.



How to use the ACK documentation

The following flowchart shows how to get started with ACK.



1. Assign roles to your Alibaba Cloud account. For more information, see ACK default roles.

   For more information about how to create a Resource Access Management (RAM) policy and assign a role-based access control (RBAC) role to a RAM user, see Create a custom RAM policy and Assign RBAC roles to RAM users.

2. Create a standard managed Kubernetes cluster. For more information, see Create a managed Kubernetes cluster.

   To create a cluster of other types, see the following topics:

   - Create an ASK cluster
   - Create a dedicated Kubernetes cluster
   - Create a professional managed Kubernetes cluster
   - Create a managed edge Kubernetes cluster
   - Create a managed Kubernetes cluster with GPU-accelerated nodes and Create a dedicated Kubernetes cluster with GPU-accelerated nodes
   - Create a managed Kubernetes cluster that runs sandboxed containers and Create a dedicated Kubernetes cluster that runs sandboxed containers

- Create a managed Kubernetes cluster for confidential computing

3. Deploy an application by using an image or an orchestration template.

   For more information, see Create a stateless application by using a Deployment.

   > **Note** If your application consists of containers that are created from different images, we recommend that you use a YAML file to deploy the application.

4. Perform O&M operations on the cluster and the application.

Cluster O&M	Application O&M
Cluster management ○ Upgrade the Kubernetes version of an ACK cluster ○ Scale out the number of worker nodes in an ACK cluster ○ Manage system components	Application deployment ○ Use a StatefulSet to create a stateful application ○ Create a Job ○ Create an application by using a private image repository
Node maintenance ○ Add existing ECS instances to an ACK cluster ○ Set node schedulability ○ Manage nodes in batches ○ Mount a data disk to a node	Application and image updates ○ Use an application trigger to redeploy an application ○ Use the aliyun-acr-credential-helper component to pull images without a password ○ Use kritis-validation-hook to automatically verify the signatures of container images
Node pool management ○ 管理节点池 ○ Schedule an application to a specific node pool ○ Set the ratio of preemptible instances to pay-as-you-go instances	Application scaling ○ Scale the number of pods for an application ○ HPA ○ Vertical pod autoscaling
○ Monitoring management Monitor application performance, Event monitoring, Enable ARMS Prometheus, and Use Prometheus to monitor a Kubernetes cluster ○ Log management Cluster auditing, 通过日志服务采集Kubernetes容器日志, Configure Log4jAppender for Kubernetes and Log Service, Collect the logs of control plane components in a managed Kubernetes cluster, and Monitor CoreDNS and analyze the CoreDNS log. ○ Cost analysis and alert management Cluster cost analysis and Alert management	

FAQ

- **How do I create a Docker image that can be used to deploy applications in ACK clusters?**

Container Registry allows you to create a container image in a convenient manner. For more information about how to create a Docker image, see Build an image for a Java application by using a Dockerfile with multi-stage builds. You can also use the open source tool Derrick to simplify the steps to containerize your application.

- **How do I deploy applications in ACK clusters if I cannot create images?**

  ACK allows you to deploy applications by using your images hosted by Container Registry. You can also use Docker official images, your favorite images added from Container Registry, and public images provided by Container Registry. For more information, see Deploy a stateless application from an image.

- **How do I plan the network when I create a cluster?**

  Before you create a cluster, make sure that the CIDR block of the virtual private cloud (VPC) where you want to deploy the cluster, the Service CIDR block, and the pod CIDR block do not overlap with one another. You can select an existing VPC to deploy the cluster and use the default CIDR blocks. In some complex scenarios, you must plan the IP addresses of Elastic Compute Service (ECS) instances, pods, and Services. For more information, see Plan CIDR blocks for an ACK cluster.

- **How do I choose between Terway and Flannel when I create a cluster?**

  Flannel is a simple and stable Container Network Interface (CNI) plug-in provided by the Kubernetes community. However, Flannel provides only basic features and does not support standard Kubernetes network policies. Terway is a network plug-in developed by Alibaba Cloud. Terway supports standard Kubernetes network policies and bandwidth throttling on containers. Terway outperforms Flannel in terms of network performance. For more information, see Use the Terway plug-in.

- **What do I do if I fail to create a cluster?**

  You can check the cluster log to locate issues and fix the issues based on the instructions provided by ACK. For more information, see Failed to create a Kubernetes cluster.

- **How do I access cluster workloads over the Internet?**

  ACK allows you to use the following methods to access cluster workloads over the Internet:

  - Use NodePort Services.
  - Use a Server Load Balancer (SLB) instance.
  - Nginx Ingress
  - Use a Domain Name System (DNS) server.
  - Use a NAT gateway that has DNAT rules configured.

- **How do I enable the communication among workloads in an ACK cluster?**

  You can use internal domain names or ClusterIP Services to enable the intercommunication of workloads in an ACK cluster.

  For example, Workload A and Workload B are deployed in a cluster. To allow Workload A to access Workload B, you can create a ClusterIP Service for Workload B. For more information, see Manage Services. After the ClusterIP Service is created, Workload A can access Workload B by using the following connection strings:

  - `<The name of the ClusterIP Service>.<The namespace to which Workload B belongs>.svc.cluster.local:<Port number>`
  - `ClusterIP:<Port number>`

- **What considerations do I take note of when I expose Services through SLB instances?**

  When you create a LoadBalancer Service, the cloud controller manager (CCM) automatically creates and configures an SLB instance for the Service. We recommend that you do not modify the SLB instance in the SLB console. Otherwise, access to the Service may be interrupted. For more information, see Considerations for configuring a LoadBalancer type Service.

- **How do I pull private images from Container Registry?**

  We recommend that you use the aliyun-acr-credential-helper component. By default, aliyun-acr-credential-helper is installed in each ACK cluster. You can use this component to pull private images from Container Registry without a password. For more information, see Use the aliyun-acr-credential-helper component to pull images without a password.

2.Work with the ACK console

Container Service for Kubernetes (ACK) allows you to manage containerized applications that run on the cloud in a convenient and efficient manner. This topic describes how to use the ACK console to deploy, expose, and monitor a containerized application in an ACK cluster.

Context

- This topic demonstrates how to deploy an ack-cube application in a professional Kubernetes cluster by using a container image. This application is an online magic cube game. After you perform the steps in this topic, a professional Kubernetes cluster is created and the magic cube application is deployed in the cluster.

- The container image used to deploy the sample application is built based on an open source project. The image address is `registry.cn-hangzhou.aliyuncs.com/acr-toolkit/ack-cube:1.0`.
- Standard Kubernetes clusters and professional Kubernetes clusters are both managed Kubernetes clusters. Compared with standard Kubernetes clusters, professional Kubernetes clusters provide higher stability and enhanced security, and are covered by the service level agreement (SLA) that includes compensation clauses. For more information about the billing of ACK clusters and the cloud resources used by ACK clusters, see Billing.

Prerequisites

You are familiar with the basic concepts of Kubernetes. For more information, see Basic concepts.

Procedure

Step 1: Activate and grant permissions to ACK

If this is the first time you use ACK, you must activate ACK and grant ACK the permissions to access cloud resources.

1. Go to the Container Service for Kubernetes page.

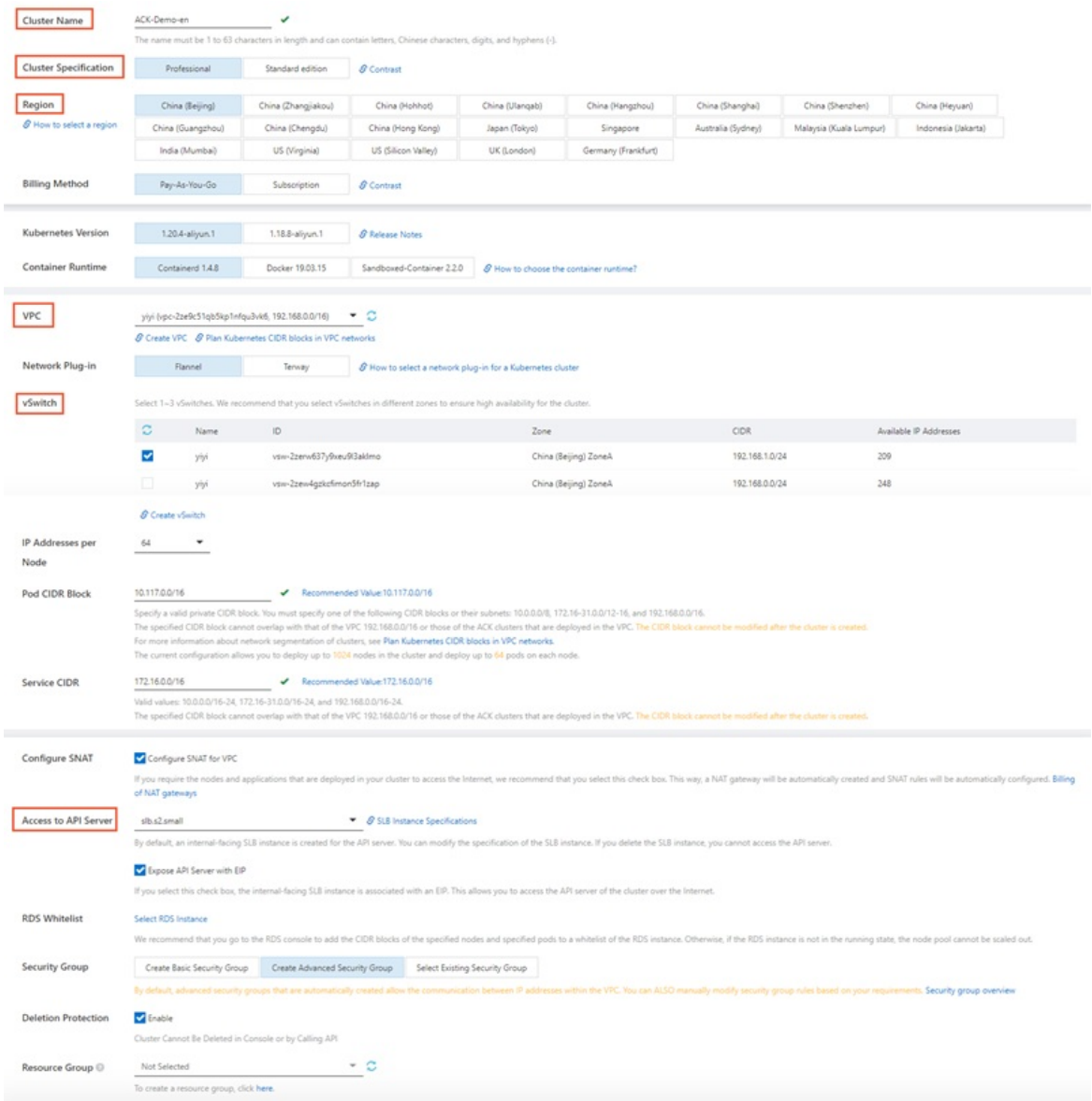
2. Read and select **Container Service for Kubernetes Terms of Service**.
3. Click **Activate Now**.
4. Log on to the ACK console.
5. On the **Container service needs to create default roles** page, click **Go to RAM console**. On the Cloud Resource Access Authorization page, click **Confirm Authorization Policy**.

   After you assign the Resource Access Management (RAM) roles to ACK, log on to the ACK console again to get started with ACK.

Step 2: Create a professional Kubernetes cluster

This step shows how to create a professional Kubernetes cluster. Default settings are used for most cluster parameters. For more information about cluster parameters, see Create a professional managed Kubernetes cluster.

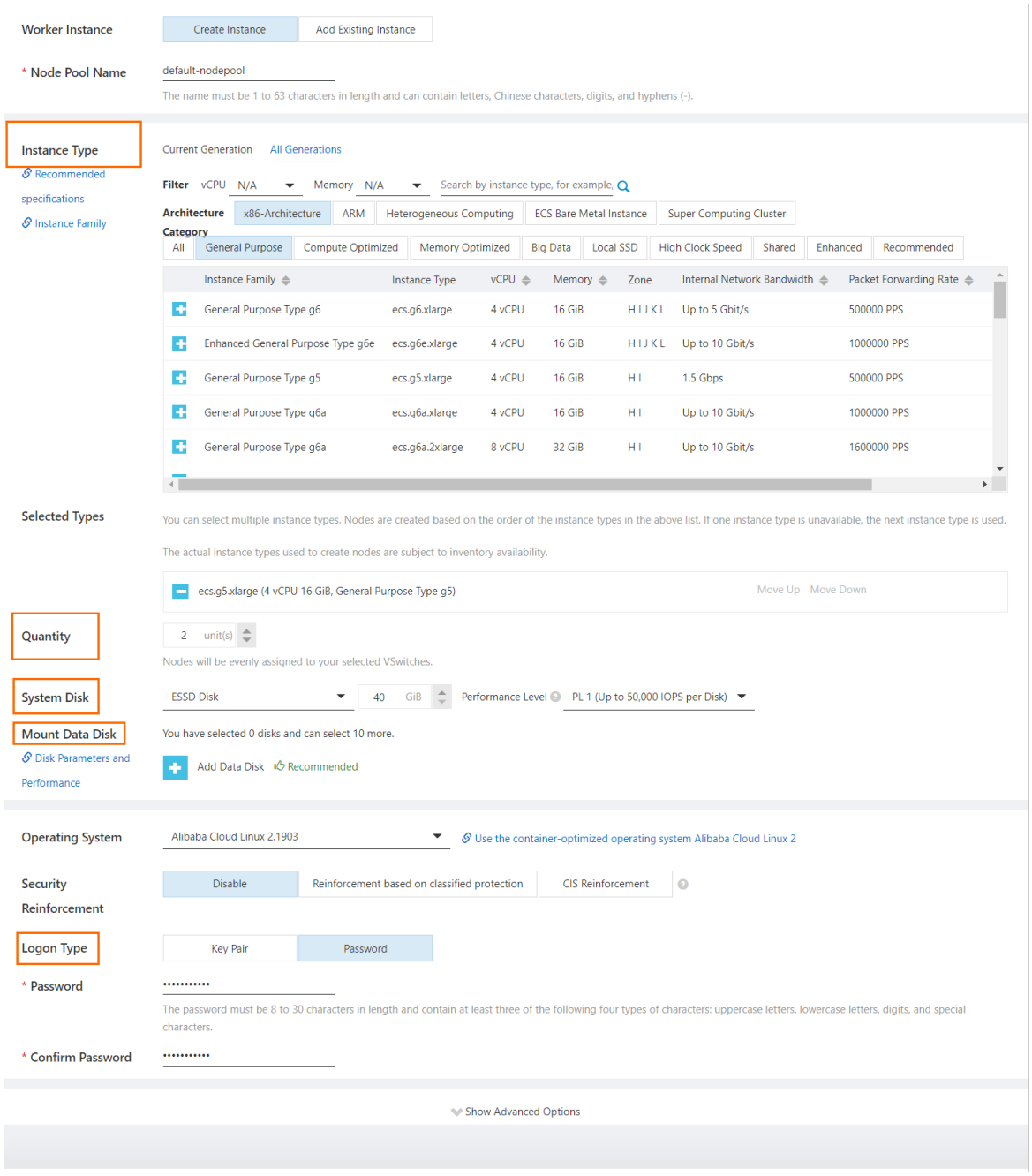
1. Log on to the ACK console.
2. In the left-side navigation pane of the ACK console, click **Clusters**.
3. In the upper-right corner of the **Clusters** page, click **Create Kubernetes Cluster**.
4. On the **Managed Kubernetes** tab, set cluster parameters as described in the following table. Use default settings for the parameters that are not included in the table.



Parameter	Description
Cluster Name	Enter a name for the cluster. In this example, the name is set to ACK-Demo.
Cluster Specification	Select the cluster type. In this example, Professional is selected. For more information about professional Kubernetes clusters, see Introduction to professional managed Kubernetes clusters.
Region	Select a region to deploy the cluster. In this example, the China (Beijing) region is selected.
VPC	ACK clusters can be deployed only in virtual private clouds (VPCs). You must specify a VPC in the same region as the cluster. In this example, a VPC named vpc-ack-demo is created in the China (Beijing) region. To create a VPC, click **Create VPC**. For more information, see Create and manage a VPC.

Parameter	Description
vSwitch	Select vSwitches for nodes in the cluster to communicate with each other. In this example, a vSwitch named vswitch-ack-demo is created in the vpc-ack-demo VPC. Select vswitch-ack-demo in the vSwitch list. To create a vSwitch, click **Create vSwitch**. For more information, see Work with vSwitches.
Access to API Server	Specify whether to expose the Kubernetes API server of the cluster to the Internet. If you want to manage the cluster over the Internet, you must expose the Kubernetes API server with an elastic IP address (EIP). In this example, **Expose API Server with EIP** is selected.

5. Click **Next: Node Pool Configurations**. Configure the following parameters as described. Use default parameters for the remaining parameters.



Parameter	Description
Instance Type	Select instance types that are used to deploy nodes. To ensure the stability of the cluster, we recommend that you select instance types with at least 4 vCPUs and 8 GiB of memory. For more information about Elastic Compute Service (ECS) instance types and how to select instance types, see Instance families and Select ECS instances to create the master and worker nodes of an ACK cluster. In this example, the ecs.g5.xlarge instance type is selected to deploy worker nodes. You can enter ecs.g5.xlarge in the search box and click the search icon.
Quantity	Specify the number of worker nodes. In this example, the number is set to 2 to avoid service interruptions caused by single points of failure (SPOFs).

Parameter	Description
System Disk	Set the system disk for nodes. In this example, the enhanced SSD is selected and the disk size is set to 40 GiB, which is the smallest size available.
Logon Type	Select the logon type for nodes. In this example, password logon is selected as the logon type and a password is specified.

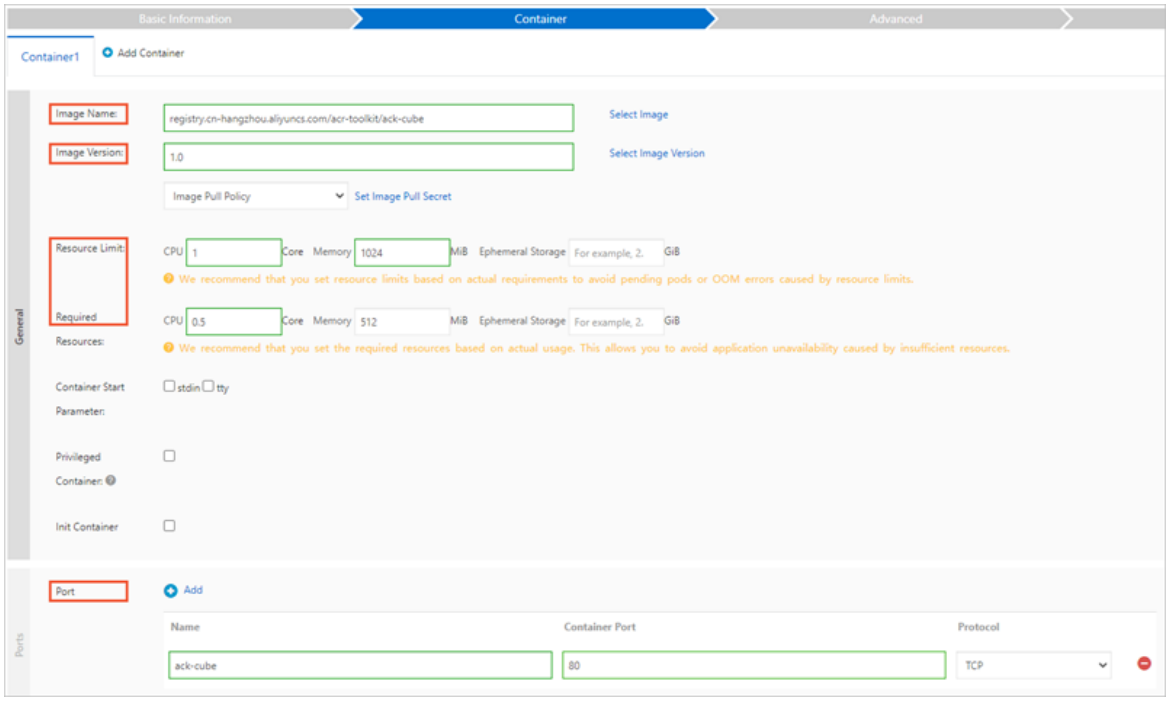
6. Click **Next: Component Configurations**. Use default settings for all component parameters.
7. Click **Next: Confirm Order**, read and select **Terms of Service**, and then click **Create Cluster**.

> **Note** It requires approximately 10 minutes to create a cluster. After the cluster is created, you can view the cluster on the Clusters page.

Step 3: Create and expose an application

This step shows how to deploy a stateless application by using a Deployment and how to expose the application to the Internet. This application is a magic cube game. For more information about the parameters used to create a Deployment, see Create a stateless application by using a Deployment.

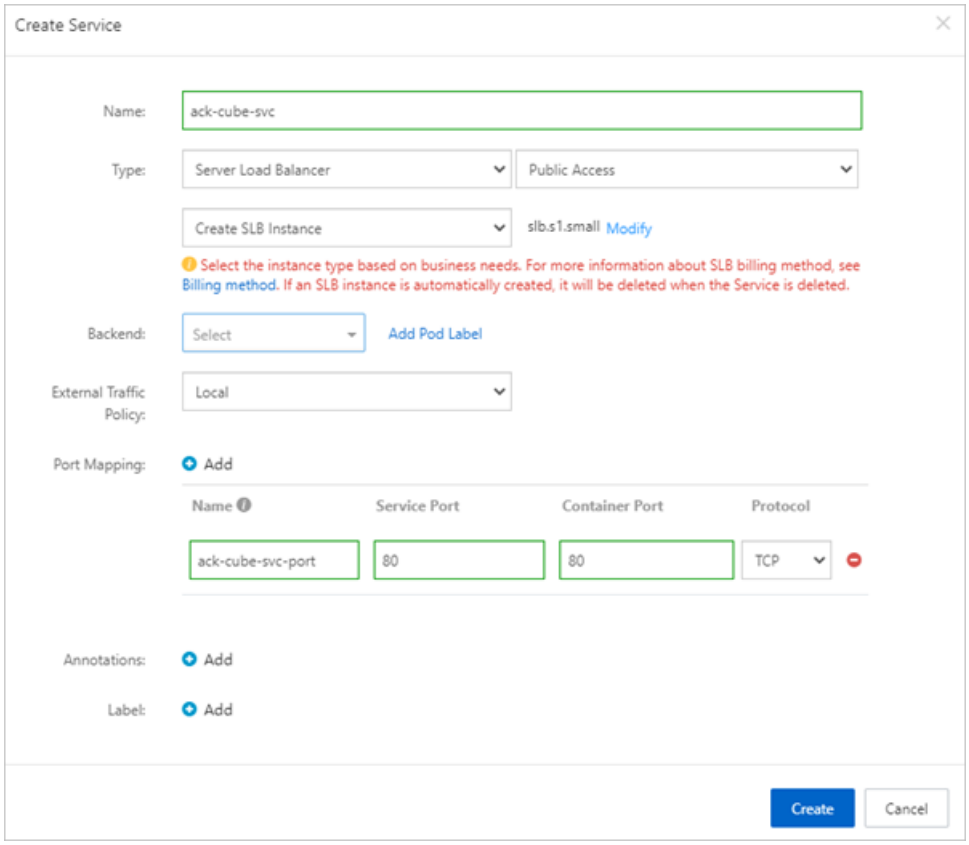
1. On the Clusters page, click the name of the ACK-Demo cluster.
2. In the left-side navigation pane of the details page, choose **Workloads > Deployments**.
3. In the upper-right corner of the **Deployments** page, click **Create from Image**.
4. On the **Basic Information** wizard page, set the application name to ack-cube.
5. Click **Next**. On the **Container** wizard page, set container parameters.



Parameter	Description

Parameter	Description
Image Name	You can enter an untagged image address or click Select Image to select the image that you want to use. In this example, `registry.cn-hangzhou.aliyuncs.com/acr-toolkit/ack-cube` is specified.
Image Version	Click **Select Image Version** and select an image version. If you do not specify an image version, the latest version is used. In this example, 1.0 is specified.
Resource Limit	Specify the resource limits of the application. This prevents the application from occupying excessive amounts of resources. In this example, 1 CPU core and 1,024 MiB of memory are specified. Ephemeral Storage is left empty.
Required Resources	Specify the amount of resources that are reserved for the application. This prevents application unavailability caused by insufficient resources. In this example, 0.5 CPU core and 512 MiB of memory are specified. Ephemeral Storage is left empty.
Port	Configure container ports. In this example, TCP port 80 is configured and named ack-cube.

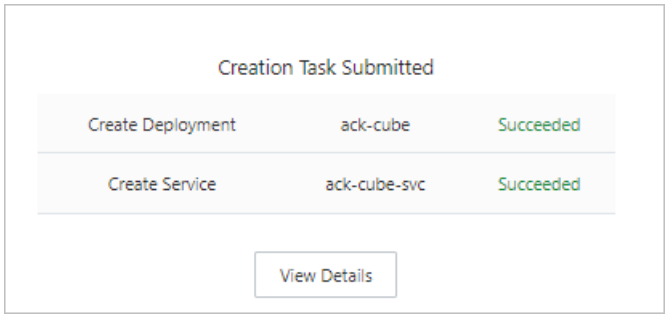
6. Click **Next**. On the **Advanced** wizard page, click **Create** to the right side of **Services**.
7. In the **Create Service** dialog box, set Service parameters and click **Create**. This creates a Service to expose the ack-cube application.



Parameter	Description
Name	Enter a name for the Service. In this example, the name is set to ack-cube-svc.
Type	The type of Service. This parameter determines how the Service is accessed. Select **Server Load Balancer**. Select **Public Access** and **Create SLB Instance**. You can click **Modify** to change the SLB instance specification based on your business requirements. In this example, the default specification Small I (slb.s1.small) is used.
Port Mapping	Specify a Service port and a container port. The container port must be the same as the one that is exposed in the backend pod. In this example, the Service port and container port are both set to 80.

8. In the lower-right corner of the **Advanced** wizard page, click **Create**.

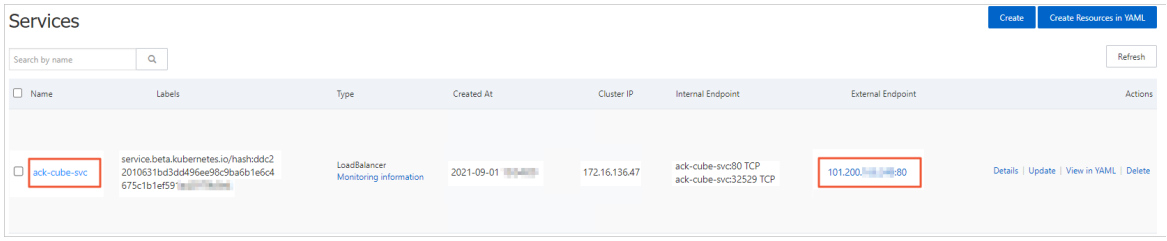
   After the application is created, you are directed to the Complete wizard page. You can view the resource objects that are included in the application and click View Details to view the application details.



Step 4: Test access to the application

This step shows how to access the application by using the Service.

1. On the Clusters page, click the name of the ACK-Demo cluster.
2. In the left-side navigation pane of the details page, choose **Network > Services**
3. On the Services page, find the ack-cube-svc Service and click the IP address in the **External Endpoint** column to start the magic cube game.

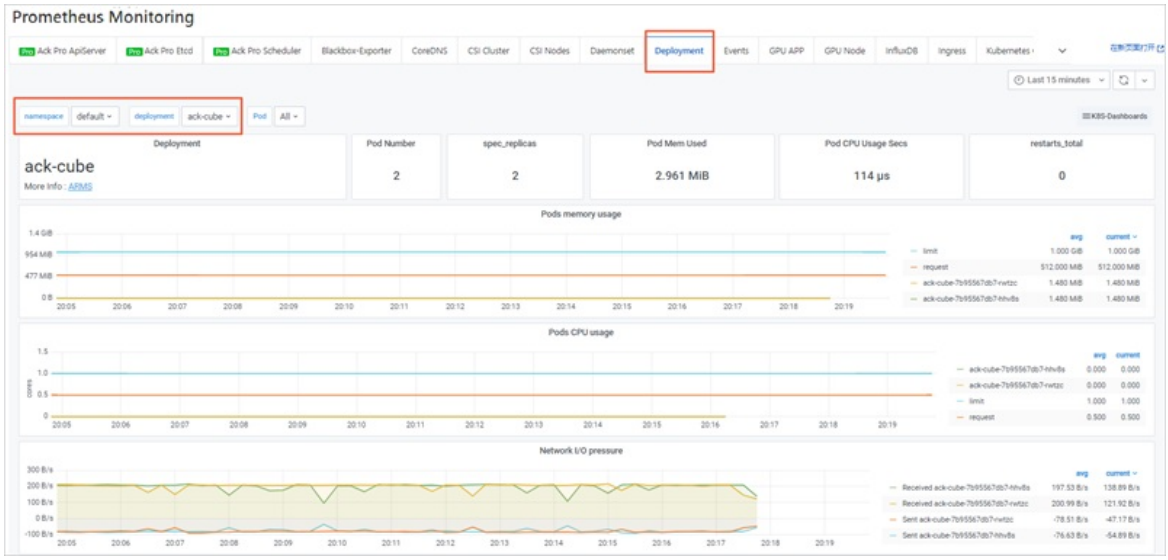


Step 5: Monitor the application

This step shows how to monitor the status of the application based on metrics such as CPU usage, memory usage, and network I/O.

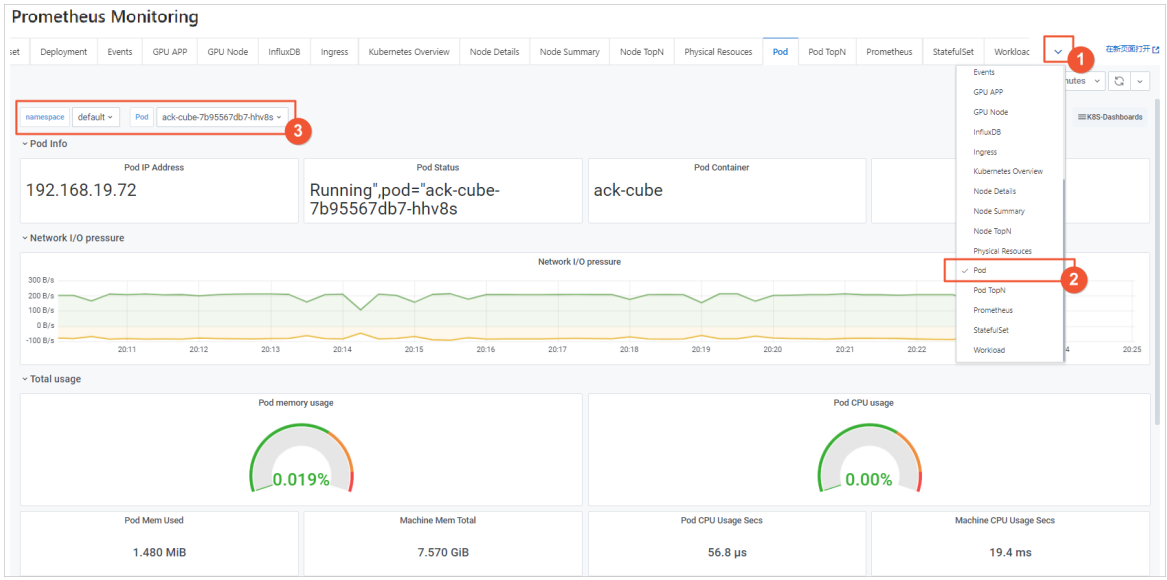
1. On the Clusters page, click the name of the ACK-Demo cluster.
2. In the left-side navigation pane of the details page, choose **Operations > Prometheus Monitoring**.
3. On the **Prometheus Monitoring** page, click the **Deployment** tab. On the Deployment tab, set **namespace** to default and **deployment** to ack-cube.

   Then, you can view the resource usage of the selected application, including the requested resources and the resource limits, as shown in the following figure.



4. On the **Prometheus Monitoring** page, click the **Pod** tab. On the Pod tab, set **namespace** to default and **Pod** to the pod that you want to check.

   The page displays the resource usage of the selected pod.



References

- To enable auto scaling for the application pods, you can configure the Horizontal Pod Autoscaler (HPA), Cron Horizontal Pod Autoscaler (CronHPA), and Vertical Pod Autoscaler (VPA). For more information, see Auto scaling overview.
- In addition to exposing applications through Services, you can use Ingresses to enable application traffic routing at Layer 7. For more information, see Create an Ingress.
- In addition to monitoring container performance, you can monitor the cluster infrastructure, application performance, and operations on your workloads. For more information, see Observability overview.
- To avoid unnecessary costs, we recommend that you delete clusters no longer in use. For more information, see Delete an ACK cluster.

3.Use ACK with kubectl

Container Service for Kubernetes (ACK) provides high-performance management services for containerized applications. You can use ACK to manage containerized applications that run on the cloud in a convenient and efficient manner. This topic describes how to use kubectl to deploy, expose, and monitor a containerized application in an ACK cluster.

Context

- This topic demonstrates how to deploy an ack-cube application in a professional Kubernetes cluster by using a container image. This application provides an online magic cube game. After you perform the steps in this topic, a professional Kubernetes cluster is created and deployed with an application that provides a magic cube game.

- The container image used to deploy the sample application is built based on an open source project. The image address is `registry.cn-hangzhou.aliyuncs.com/acr-toolkit/ack-cube:1.0`.
- kubectl is a command-line tool that Kubernetes provides for you to connect to and manage Kubernetes clusters. For more information about kubectl, see kubectl.
- Cloud Shell is a web-based command-line tool provided by Alibaba Cloud. You can use kubectl in Cloud Shell in the ACK console to manage ACK clusters. Installation and configuration are not required.

Prerequisites

You are familiar with the basic concepts of Kubernetes. For more information, see Basic concepts.

Procedure

Step 1: Activate and grant permissions to ACK

If this is the first time you use ACK, you must activate ACK and grant ACK the permissions to access cloud resources.

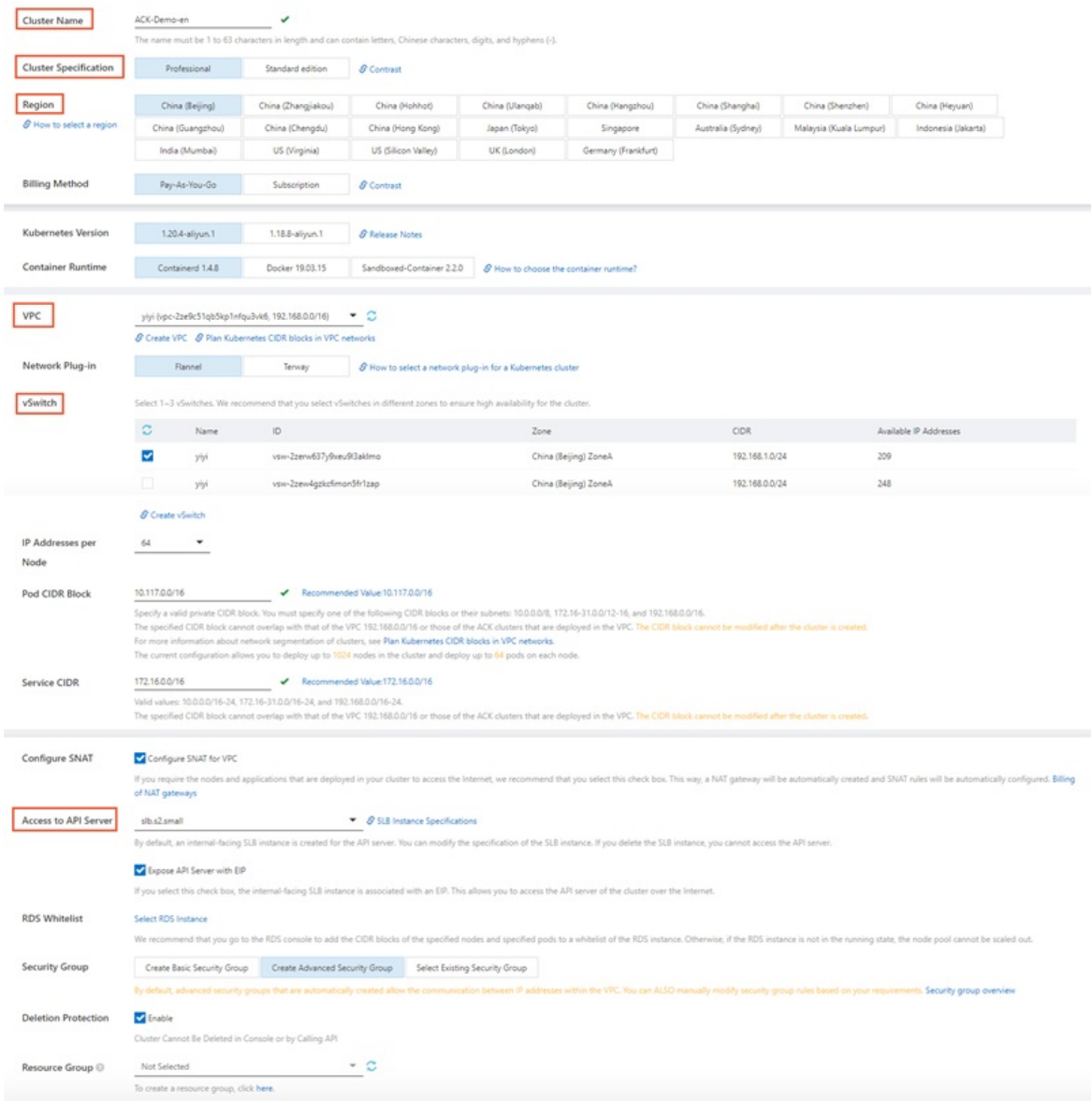
1. Go to the Container Service for Kubernetes page.
2. Read and select **Container Service for Kubernetes Terms of Service**.
3. Click **Activate Now**.
4. Log on to the ACK console.
5. On the **Container service needs to create default roles** page, click **Go to RAM console**. On the Cloud Resource Access Authorization page, click **Confirm Authorization Policy**.

   After you assign the Resource Access Management (RAM) roles to ACK, log on to the ACK console again to get started with ACK.

Step 2: Create a professional Kubernetes cluster

This step shows how to create a professional Kubernetes cluster. Default settings are used for most cluster parameters. For more information about cluster parameters, see Create a professional managed Kubernetes cluster.

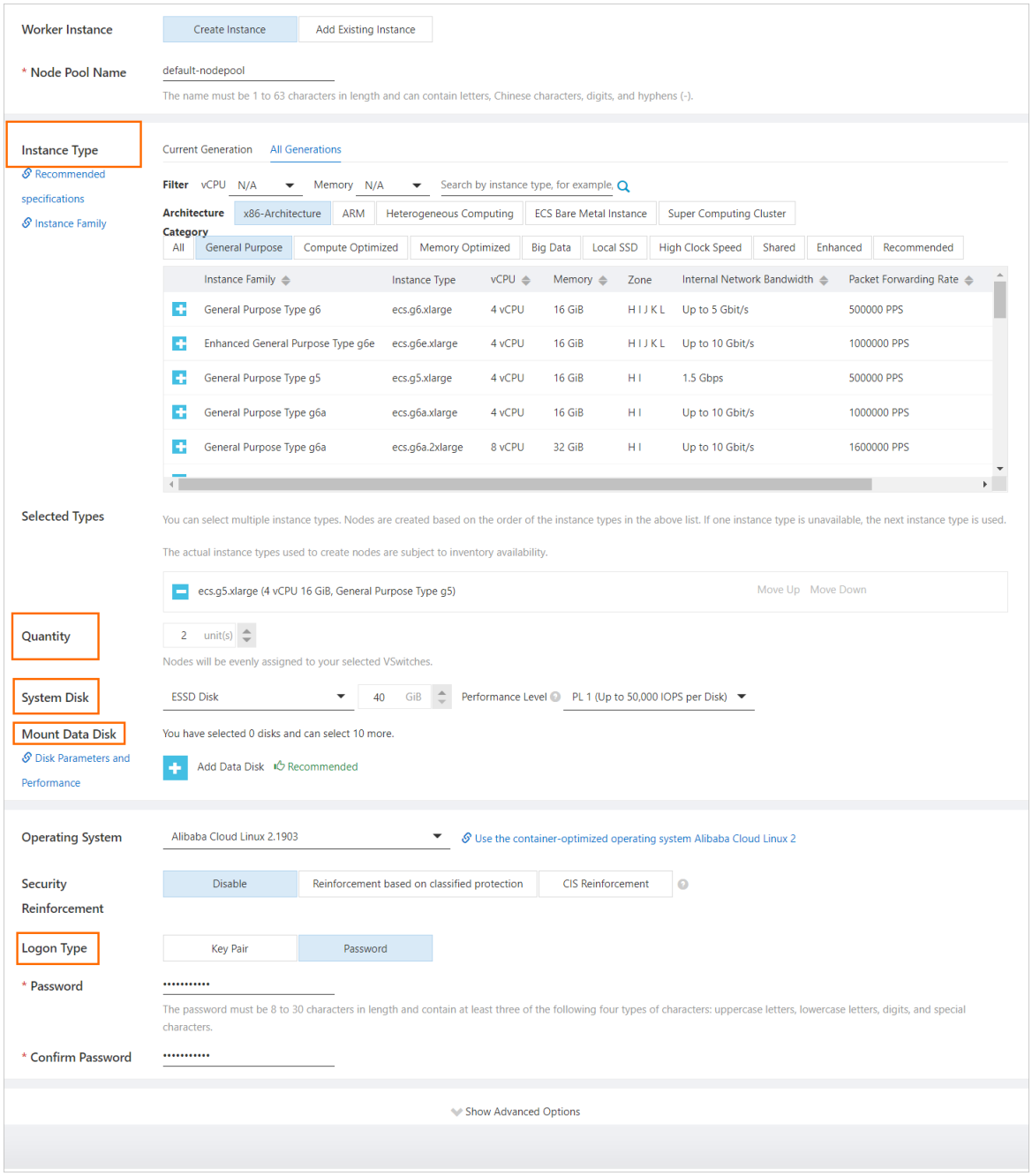
1. Log on to the ACK console.
2. In the left-side navigation pane of the ACK console, click **Clusters**.
3. In the upper-right corner of the **Clusters** page, click **Create Kubernetes Cluster**.
4. On the **Managed Kubernetes** tab, set cluster parameters as described in the following table. Use default settings for the parameters that are not included in the table.



Parameter	Description
Cluster Name	Enter a name for the cluster. In this example, the name is set to ACK-Demo.
Cluster Specification	Select the cluster type. In this example, Professional is selected. For more information about professional Kubernetes clusters, see Introduction to professional managed Kubernetes clusters.
Region	Select a region to deploy the cluster. In this example, the China (Beijing) region is selected.
VPC	ACK clusters can be deployed only in virtual private clouds (VPCs). You must specify a VPC in the same region as the cluster. In this example, a VPC named vpc-ack-demo is created in the China (Beijing) region. To create a VPC, click **Create VPC**. For more information, see Create and manage a VPC.

Parameter	Description
vSwitch	Select vSwitches for nodes in the cluster to communicate with each other. In this example, a vSwitch named vswitch-ack-demo is created in the vpc-ack-demo VPC. Select vswitch-ack-demo in the vSwitch list. To create a vSwitch, click **Create vSwitch**. For more information, see Work with vSwitches.
Access to API Server	Specify whether to expose the Kubernetes API server of the cluster to the Internet. If you want to manage the cluster over the Internet, you must expose the Kubernetes API server with an elastic IP address (EIP). In this example, **Expose API Server with EIP** is selected.

5. Click **Next: Node Pool Configurations**. Configure the following parameters as described. Use default parameters for the remaining parameters.



Parameter	Description
Instance Type	Select instance types that are used to deploy nodes. To ensure the stability of the cluster, we recommend that you select instance types with at least 4 vCPUs and 8 GiB of memory. For more information about Elastic Compute Service (ECS) instance types and how to select instance types, see Instance families and Select ECS instances to create the master and worker nodes of an ACK cluster. In this example, the ecs.g5.xlarge instance type is selected to deploy worker nodes. You can enter ecs.g5.xlarge in the search box and click the search icon.
Quantity	Specify the number of worker nodes. In this example, the number is set to 2 to avoid service interruptions caused by single points of failure (SPOFs).

Parameter	Description
System Disk	Set the system disk for nodes. In this example, the enhanced SSD is selected and the disk size is set to 40 GiB, which is the smallest size available.
Logon Type	Select the logon type for nodes. In this example, password logon is selected as the logon type and a password is specified.

6. Click **Next: Component Configurations**. Use default settings for all component parameters.
7. Click **Next: Confirm Order**, read and select **Terms of Service**, and then click **Create Cluster**.

> **Note** It requires approximately 10 minutes to create a cluster. After the cluster is created, you can view the cluster on the Clusters page.

Step 3: Connect to the cluster

This step shows how to connect to the ACK cluster by using a kubectl client or Cloud Shell. For more information, see Connect to ACK clusters by using kubectl and Use kubectl on Cloud Shell to manage ACK clusters.

Method 1: Connect to the cluster by using a kubectl client

1. Log on to the ACK console.
2. In the left-side navigation pane of the ACK console, click **Clusters**.
3.
4. On the **Cluster Information** page, click the **Connection Information** tab. Click Copy on the **Public Access** tab. This way, the credential used to access the cluster over the Internet is copied.
5. Paste the credential to the *config* file in the *$HOME/.kube* directory, save the file, and then exit.

> **Note** If the *.kube* folder and the *config* file do not exist in the *$HOME/* directory, you must manually create the folder and file.

6. Run a kubectl command to connect to the cluster.

   Run the following command to query the namespaces of the cluster:

   ```
   kubectl get namespace
   ```

   Expected output:

   ```
   NAME              STATUS   AGE
   arms-prom         Active   4h39m
   default           Active   4h39m
   kube-node-lease   Active   4h39m
   kube-public       Active   4h39m
   kube-system       Active   4h39m
   ```

Method 2: Connect to the cluster by using Cloud Shell

1. Log on to the ACK console.
2. In the left-side navigation pane of the ACK console, click **Clusters**.

3. On the Clusters page, find the cluster that you want to manage and choose **More > Open Cloud Shell** in the **Actions** column.

   It requires a few seconds to start Cloud Shell. After Cloud Shell is started, you can run kubectl commands on the Cloud Shell interface to manage the cluster and applications deployed in the cluster.

Step 4: Deploy and expose an application

This step shows how to use kubectl to deploy a stateless application by creating a Deployment and use a LoadBalancer Service to expose the application. For more information about how to expose an application, see Use an automatically created SLB instance to expose an application.

1. Use the following YAML template to create an *ack-cube.yaml* file:

```
apiVersion: apps/v1 # for versions before 1.8.0 use apps/v1beta1
kind: Deployment
metadata:
 name: ack-cube # The name of the application.
 labels:
  app: ack-cube
spec:
 replicas: 2 # The number of replicated pods.
 selector:
  matchLabels:
   app: ack-cube  # You must specify the same value for the selector of the Service that is used to expos
e the application.
 template:
  metadata:
   labels:
    app: ack-cube
  spec:
   containers:
   - name: ack-cube
    image: registry.cn-hangzhou.aliyuncs.com/acr-toolkit/ack-cube:1.0 # Replace with the address of th
e image that you want to use in the format of <image_name:tags>.
    ports:
    - containerPort: 80 # The container port that you want to open.
    resources:
     limits: # The resource limits of the application.
      cpu: '1'
      memory: 1Gi
     requests: # The resource requests of the application.
      cpu: 500m
      memory: 512Mi
```

2. Run the following command to deploy the ack-cube application:

```
kubectl apply -f ack-cube.yaml
```

3. Run the following command to query the status of the application:

```
kubectl get deployment ack-cube
```

   Expected output:

```
NAME    READY  UP-TO-DATE  AVAILABLE  AGE
ack-cube  2/2   2       2       96s
```

4. Use the following YAML template to create an *ack-cube-svc.yaml* file. Set `selector` to the value of `matchLabels` in the *ack-cube.yaml* file. In this example, the value is `app: ack-cube`. This adds the application to the backend of the Service.

```
apiVersion: v1
kind: Service
metadata:
 labels:
  app: ack-cube
 name: ack-cube-svc
 namespace: default
spec:
 ports:
 - port: 80
  protocol: TCP
  targetPort: 80
 selector:
  app: ack-cube # You must specify the value of the matchLabels parameter in the YAML file that is used to create the Deployment.
 type: LoadBalancer
```

5. Run the following command to create a Service named ack-cube-svc and use the Service to expose the application.

   ACK automatically creates an Internet-facing Server Load Balancer (SLB) instance and associates the instance with the Service.

```
kubectl apply -f ack-cube-svc.yaml
```

6. Run the following command to verify that the LoadBalancer Service is created.

   The application that you created is exposed by using the IP address in the EXTERNAL-IP column in the output.

```
kubectl get svc ack-cube-svc
```

   Expected output:

```
NAME      TYPE      CLUSTER-IP    EXTERNAL-IP   PORT(S)    AGE
ack-cube-svc  LoadBalancer  172.16.72.161  47.94.xx.xx  80:31547/TCP  32s
```

Step 5: Test the application

Enter the IP address (EXTERNAL-IP) in the address bar of your browser and press Enter to start the magic cube game.

Step 6: Monitor the application

For more information, see Step 5: Monitor the application.

References

- To enable the auto scaling of application pods, you can configure the Horizontal Pod Autoscaler

(HPA), Cron Horizontal Pod Autoscaler (CronHPA), and Vertical Pod Autoscaler (VPA). For more information, see Auto scaling overview.

- In addition to exposing applications through Services, you can use Ingresses to enable application traffic routing at Layer 7. For more information, see Create an Ingress.
- In addition to monitoring container performance, you can also monitor the cluster infrastructure, application performance, and your business operations. For more information, see Observability overview.
- To avoid unnecessary costs, we recommend that you delete clusters no longer in use. For more information, see Delete an ACK cluster.

4.Basic operations

4.1. Create a managed Kubernetes cluster

This topic describes how to create a managed Kubernetes cluster in the Container Service for Kubernetes (ACK) console.

Prerequisites

- ACK must be authorized to access other cloud services before you start. For more information, see Quick start for first-time users.
- Resource Access Management (RAM) is activated in the RAM console.
- Auto Scaling (ESS) is activated.

Context

The following example shows how to create a managed Kubernetes cluster. Default values and minimal configurations are used in specific settings.

For more information about the limits of ACK clusters, see Create a managed Kubernetes cluster.

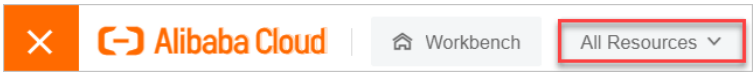
Procedure

1. Log on to the ACK console.
2. In the left-side navigation pane of the ACK console, click **Clusters**.
3. In the upper-right corner of the **Clusters** page, click **Create Kubernetes Cluster**.
4. Set the parameters.

   Default values are used for most parameters in this example. The following table describes the parameters.

   i. Set the parameters on the **Cluster Configurations** wizard page.

Parameter	Description
Cluster Name	Enter a name for the ACK cluster. **Note** The name must be 1 to 63 characters in length, and can contain digits, letters, and hyphens (-).
Cluster Specification	Select a cluster type. You can select **Standard edition** or **Professional**.
Region	Select a region to deploy the cluster.

Parameter	Description
All Resources	Move the pointer over **All Resources** at the top of the page and select the resource group that you want to use. After you select a resource group, virtual private clouds (VPCs) and vSwitches are filtered based on the selected resource group. When you create a cluster, only the VPCs and vSwitches that belong to the selected resource group are displayed in the console.  **All Resources** is selected in this example.
Kubernetes Version	Use the default setting.
Container Runtime	The Docker runtime is selected in this example. For more information about the differences among Docker, containerd, and Sandboxed-Container, see Comparison of Docker, containerd, and Sandboxed-Container.
VPC	Set the virtual private cloud (VPC) where you want to deploy the cluster. Select a VPC to deploy the cluster. Standard VPCs and shared VPCs are supported. ■ Shared VPC: The owner of a VPC (resource owner) can share the vSwitches in the VPC with other accounts in the same organization. ■ Standard VPC: The owner of a VPC (resource owner) cannot share the vSwitches in the VPC with other accounts. **Note** ACK clusters support only VPCs. You can select a VPC from the drop-down list. If no VPC is available, click **Create VPC** to create one. For more information, see Create and manage a VPC.
vSwitch	Select vSwitches. You can select up to three vSwitches that are deployed in different **zones**. If no vSwitch is available, click **Create vSwitch** to create one. For more information, see Work with vSwitches.
Network Plug-in	By default, Flannel is selected. For more information about Flannel and Terway, see Use the Terway plug-in.

Parameter	Description
IP Addresses per Node	The default value is 64. If you select **Flannel** as the network plug-in, you must set **IP Addresses per Node**. **Note** ■ **IP Addresses per Node** specifies the maximum number of IP addresses that can be assigned to each node. We recommend that you use the default value. ■ After you select the VPC and specify the number of IP addresses per node, recommended values are automatically generated for Pod CIDR Block and Service CIDR. The system also provides the maximum number of nodes that can be deployed in the cluster and the maximum number of pods that can be deployed on each node. You can modify the values based on your business requirements.
Pod CIDR Block	If you select **Flannel** as the network plug-in, you must set **Pod CIDR Block**. The CIDR block specified by **Pod CIDR Block** cannot overlap with that of the VPC or those of the existing clusters in the VPC. The CIDR block cannot be modified after the cluster is created. The Service CIDR block cannot overlap with the pod CIDR block. For more information about how to plan CIDR blocks for an ACK cluster, see Plan CIDR blocks for an ACK cluster.
Service CIDR	Set **Service CIDR**. The CIDR block specified by **Service CIDR** cannot overlap with that of the VPC or those of the existing clusters in the VPC. The CIDR block cannot be modified after the cluster is created. The Service CIDR block cannot overlap with the pod CIDR block. For more information about how to plan CIDR blocks for an ACK cluster, see Plan CIDR blocks for an ACK cluster.
Configure SNAT	By default, **Configure SNAT for VPC** is selected. By default, an ACK cluster cannot access the Internet. If the VPC that you select for the cluster cannot access the Internet, you can select **Configure SNAT for VPC**. This way, ACK will create a NAT gateway and configure SNAT rules to enable Internet access for the VPC.

Parameter	Description
Access to API Server	By default, an internal-facing SLB instance is created for the Kubernetes API server of the cluster. You can modify the specification of the SLB instance. For more information, see Instance types and specifications. **Notice** If you delete the SLB instance, you cannot access the Kubernetes API server of the cluster. By default, **Expose API Server with EIP** is not selected. The default setting is used in this example. Select or clear **Expose API Server with EIP**. The ACK API server provides multiple HTTP-based RESTful APIs, which can be used to create, delete, modify, query, and monitor resources, such as pods and Services. ■ If you select this check box, an elastic IP address (EIP) is created and associated with an SLB instance. Port 6443 used by the API server is opened on master nodes. You can connect to and manage the cluster by using kubeconfig files over the Internet. ■ If you clear this check box, no EIP is created. You can connect to and manage the cluster by using kubeconfig files only from within the VPC.
RDS Whitelist	Configure the whitelist of the ApsaraDB RDS instance. Add the IP addresses of nodes in the cluster to the ApsaraDB RDS whitelist. **Note** To enable an ApsaraDB RDS instance to access the cluster, you must make sure that the instance is deployed in the VPC where the cluster is deployed. By default, RDS whitelist is not configured.
Security Group	You can select **Create Basic Security Group**, **Create Advanced Security Group**, or **Select Existing Security Group**. For more information about security groups, see Overview. **Note** To select **Select Existing Security Group**, Submit a ticket to apply to be added to a whitelist. By default, **Create Advanced Security Group** is selected. The default setting is used in this example.
Show Advanced Options	The advanced settings of the cluster are not configured in this example. For more information about the advanced settings of an ACK cluster, see Configure advanced settings for a cluster.

ii. Click **Next: Node Pool Configurations** to configure the node pool.

Parameter	Description
Worker Instance	Specify whether to use existing Elastic Compute Service (ECS) instances or create ECS instances. By default, **Create Instance** is selected.
Node Pool Name	The name of the node pool. **Note** The name must be 1 to 63 characters in length, and can contain digits, letters, and hyphens (-).
Billing Method	By default, **Pay-As-You-Go** is selected. The **pay-as-you-go** and **subscription** billing methods are supported.
Instance Type	You can select multiple instance types. For more information, see Instance families.
Selected Types	The selected instance types are displayed.
Quantity	Specify the number of worker nodes (ECS instances) to be created.
System Disk	By default, **Ultra Disk** is selected. The default storage capacity is **120 GiB**.
Mount Data Disk	By default, this option is not selected. **Enhanced SSDs**, **standard SSDs**, and **ultra disks** are supported. You can enable **disk encryption** and **disk backup** when you mount a data disk. If you enable disk encryption, only the default customer master key (CMK) can be used.
Operating System	By default, Alibaba Cloud Linux 2.1903 is selected.

Parameter	Description
Logon Type	■ Key pair logon ■ **Key Pair**: Select an SSH key pair from the drop-down list. ■ **create a key pair**: Create an SSH key pair if none is available. For more information about how to create an SSH key pair, see Create an SSH key pair. After the key pair is created, set it as the credential that is used to log on to the cluster. ■ Password logon ■ **Password**: Enter the password that is used to log on to the nodes. ■ **Confirm Password**: Enter the password again. **Note** The password must be 8 to 30 characters in length, and must contain at least three of the following character types: uppercase letters, lowercase letters, digits, and special characters. The password cannot contain underscores (_).
Advanced settings of worker nodes	The advanced settings of worker nodes are not configured in this example. For more information about the advanced settings of an ACK cluster, see Configure advanced settings for a cluster.

iii. Click **Next: Component Configurations** to configure components on the **Component Configurations** wizard page.

Parameter	Description
Ingress	By default, **Install Ingress Controllers** is selected. Select **Public Network** for **SLB Network Type**.
Volume Plug-in	By default, **CSI** is selected. For more information about the Flexvolume and CSI plug-ins, see CSI overview.
Monitoring Agents	Specify whether to install the CloudMonitor agent. By default, **Install CloudMonitor Agent on ECS Instance** and **Enable Prometheus Monitoring** are selected. After the CloudMonitor agent is installed on ECS nodes, you can view monitoring data about the nodes in the CloudMonitor console. The CloudMonitor agent is not installed in this example.
Log Service	Specify whether to enable Log Service. You can select an existing Log Service project or create one. By default, **Enable Log Service** is selected. When you create an application, you can enable Log Service with a few steps. For more information, see 通过日志服务采集Kubernetes容器日志. By default, **Install node-problem-detector and Create Event Center** is selected. You can specify whether to enable the Kubernetes event center in the Log Service console. For more information, see Create and use a Kubernetes event center. Log Service is enabled in this example.
Log Collection for Control Plane Components	If you select **Enable**, log of the control plane components is collected to the specified Log Service project that belongs to the current account. For more information, see Collect the logs of control plane components in a managed Kubernetes cluster.

iv. Click **Next: Confirm Order**.

v. Read **Terms of Service**, select the check box, and then click **Create Cluster**.

> **Note** It requires about 10 minutes to create a managed Kubernetes cluster that contains multiple nodes.

4.2. Deploy a stateless application from an image

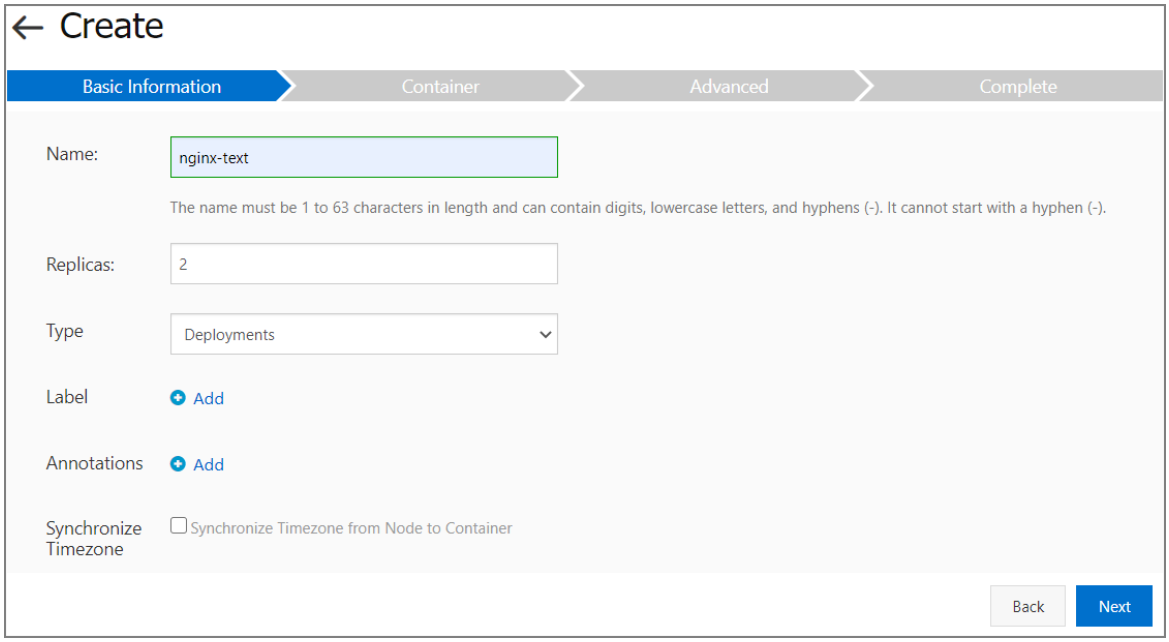
This topic describes how to use an image to deploy an NGINX application that is accessible over the Internet.

Prerequisites

A Container Service for Kubernetes (ACK) cluster is created. For more information, see Create a managed Kubernetes cluster.

Procedure

1. Log on to the ACK console.
2. In the left-side navigation pane of the ACK console, click **Clusters**.
3. On the **Clusters** page, find the cluster that you want to manage and click the name of the cluster or click **Details** in the **Actions** column. The details page of the cluster appears.
4. In the left-side navigation pane of the details page, choose **Workloads > Deployments**.
5. In the upper-right corner of the **Deployments** page, click **Create from Image**.
6. On the **Basic Information** wizard page, configure the basic settings.



Parameter	Description
Name	The name of the application.
Namespace	The namespace where you want to deploy the application. The default namespace is automatically selected. You can select another namespace.
Replicas	The number of pods that are provisioned for the application.
Type	The type of workload. You can select **Deployments**, **StatefulSets**, **Jobs**, **CronJobs**, or **DaemonSets**.
Label	Add a label to the application. The label is used to identify the application.
Annotations	Add an annotation to the application.

Parameter	Description
Synchronize Timezone	Specify whether to synchronize the time zone between nodes and containers.

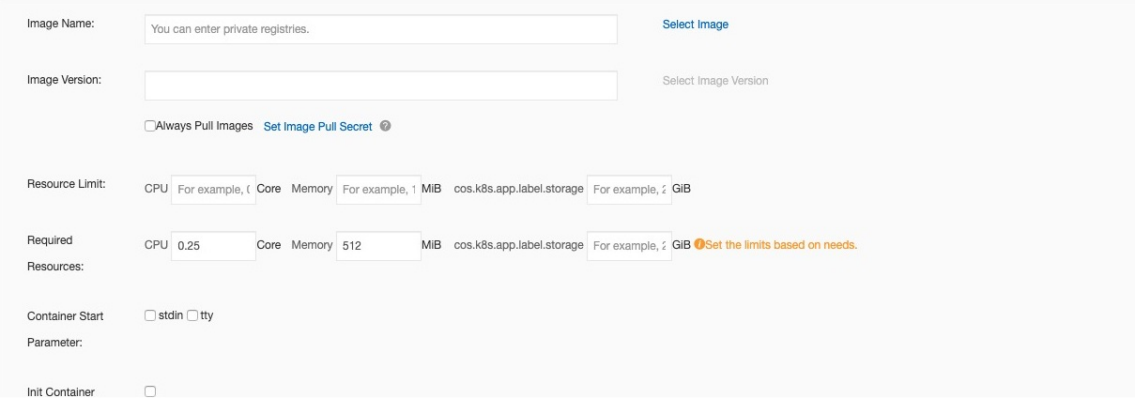
> **Note** In this example, **Deployments** is selected. The default namespace is selected. You can select another **namespace**. The number of replicas equals the number of pods that are provisioned for the application.

7. Click **Next** to proceed to the **Container** wizard page.
8. Configure containers.

> **Note** In the upper part of the **Container** wizard page, click **Add Container** to add more containers for the application.

The following parameters are required to configure the containers.

- General settings



Parameter	Description
Image Name	Click **Select Image**. In the dialog box that appears, select an image and click **OK**. In this example, an NGINX image is selected. You can also enter the path of an image stored in a private registry. The image path must be in the `domainname/namespace/imagename:tag` format.

Parameter	Description
Image Version	■ Click **Select Image Version** and select an image version. If you do not specify an image version, the latest image version is used. ■ You can select the following image pull policies: ■ **ifNotPresent**: If the image that you want to pull is found on your on-premises machine, the image on your on-premises machine is used. Otherwise, ACK pulls the image from the corresponding repository. ■ **Always**: ACK pulls the image from Container Registry each time the application is deployed or scaled out. ■ **Never**: ACK uses only images on your on-premises machine. **Note** If you select **Image Pull Policy**, no image pull policy is applied. ■ To pull the image without a password, click **Set Image Pull Secret** to set a Secret that is used to pull the image. For more information, see Use the aliyun-acr-credential-helper component to pull images without a password.
Resource Limit	You can specify an upper limit for the CPU, memory, and ephemeral storage space that the container can consume. This prevents the container from occupying an excessive amount of resources. The CPU resource is measured in milicores (one thousandth of one core). The memory resource is measured in MiB. The ephemeral storage resource is measured in GiB.
Required Resources	The amount of CPU and memory resources that are reserved for this application. These resources are exclusive to the container. This prevents the application from becoming unavailable if other services or processes compete for computing resources.
Container Start Parameter	■ stdin: Pass stdin to the container. ■ tty: Stdin is a TeleTYpewriter (TTY).
Privileged Container	■ If you select Privileged Container, privileged=true is set for the container and the privilege mode is enabled. ■ If you do not select Privileged Container, privileged=false is set for the container and the privilege mode is disabled.
Init Container	If you select Init Container, an init container is created. An init container provides tools to manage pods. For more information, see Init Containers.

- (Optional)Ports

  Configure container ports.

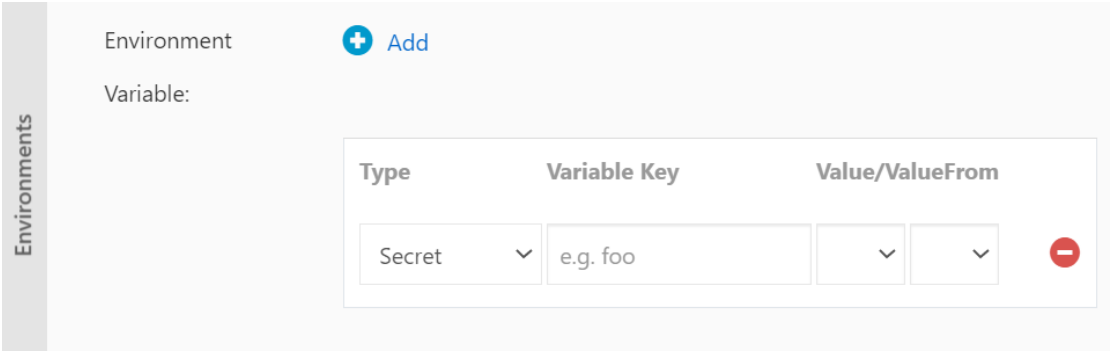
  - Name: Enter a name for the port.
  - Container Port: Enter the container port that you want to open. Enter a port number from 1 to 65535.
  - Protocol: TCP or UDP.

- (Optional)Environments

  You can configure environment variables in key-value pairs for pods. Environment variables are used to apply pod configurations to containers. For more information, see Pod variables.

  - Type: Select the type of the environment variable. You can select **Custom**, **ConfigMaps**, **Secret**, **Value/ValueFrom**, or **ResourceFieldRef**. If you select ConfigMaps or Secret as the type of the environment variable, all values in the selected ConfigMap or Secret are passed to the container environment variables. In this example, Secret is selected.

    Select **Secret** from the Type drop-down list and select a Secret from the **Value/ValueFrom** drop-down list. All values in the selected Secret are passed to the environment variable.

    In this case, the YAML file that is used to deploy the application contains the settings that reference all values in the specified Secret.

    ```
    envFrom:
      - secretRef:
          name: test
    ```

  - Variable Key: Specify the name of the environment variable.
  - Value/ValueFrom: Specify the value that is referenced by the environment variable.

- (Optional)Health Check

  Health check settings include liveness, readiness, and startup probes. Liveness probes determine when to restart the container. Readiness probes determine whether the container is ready to accept network traffic. Startup probes detect whether the application in the container is started. For more information about health checks, see Configure Liveness, Readiness, and Startup Probes.

Request type	Description

Request type	Description
HTTP	Sends an HTTP GET request to the container. You can configure the following parameters: ■ Protocol: HTTP or HTTPS. ■ Path: the requested path on the server. ■ Port: Enter the container port that you want to open. Enter a port number from 1 to 65535. ■ HTTP Header: Enter the custom headers in the HTTP request. Duplicate headers are allowed. Key-value pairs are supported. ■ Initial Delay (s): the initialDelaySeconds field in the YAML file. This field specifies the time (in seconds) that the system must wait before it can send a probe to the container after the container is started. Default value: 3. ■ Period (s): the periodSeconds field in the YAML file. This field specifies the interval (in seconds) at which probes are performed. Default value: 10. Minimum value: 1. ■ Timeout (s): the timeoutSeconds field in the YAML file. This field specifies the time (in seconds) after which a probe times out. Default value: 1. Minimum value: 1. ■ Healthy Threshold: the minimum number of times that an unhealthy container must consecutively pass health checks before it is considered healthy. Default value: 1. Minimum value: 1. For liveness probes, this parameter must be set to 1. ■ Unhealthy Threshold: the minimum number of times that a healthy container must consecutively fail health checks before it is considered unhealthy. Default value: 3. Minimum value: 1.

Request type	Description
TCP	Sends a TCP socket to the container. kubelet attempts to open the socket on the specified port. If the connection can be established, the container is considered healthy. Otherwise, the container is considered unhealthy. You can set the following parameters: ■ Port: Enter the container port that you want to open. Enter a port number from 1 to 65535. ■ Initial Delay (s): the initialDelaySeconds field in the YAML file. This field specifies the time (in seconds) that the system must wait before it can send a probe to the container after the container is started. Default value: 15. ■ Period (s): the periodSeconds field in the YAML file. This field specifies the interval (in seconds) at which probes are performed. Default value: 10. Minimum value: 1. ■ Timeout (s): the timeoutSeconds field in the YAML file. This field specifies the time (in seconds) after which a probe times out. Default value: 1. Minimum value: 1. ■ Healthy Threshold: the minimum number of times that an unhealthy container must consecutively pass health checks before it is considered healthy. Default value: 1. Minimum value: 1. For liveness probes, this parameter must be set to 1. ■ Unhealthy Threshold: the minimum number of times that a healthy container must consecutively fail health checks before it is considered unhealthy. Default value: 3. Minimum value: 1.
Command	Runs a probe command in the container to check the health status of the container. You can set the following parameters: ■ Command: the probe command that is run to check the health status of the container. ■ Initial Delay (s): the initialDelaySeconds field in the YAML file. This field specifies the time (in seconds) that the system must wait before it can send a probe to the container after the container is started. Default value: 5. ■ Period (s): the periodSeconds field in the YAML file. This field specifies the interval (in seconds) at which probes are performed. Default value: 10. Minimum value: 1. ■ Timeout (s): the timeoutSeconds field in the YAML file. This field specifies the time (in seconds) after which a probe times out. Default value: 1. Minimum value: 1. ■ Healthy Threshold: the minimum number of times that an unhealthy container must consecutively pass health checks before it is considered healthy. Default value: 1. Minimum value: 1. For liveness probes, this parameter must be set to 1. ■ Unhealthy Threshold: the minimum number of times that a healthy container must consecutively fail health checks before it is considered unhealthy. Default value: 3. Minimum value: 1.

- Lifecycle

You can configure the lifecycle of the container by using the following parameters: Start, Post Start, and Pre Stop. For more information, see Configure the lifecycle of a container.

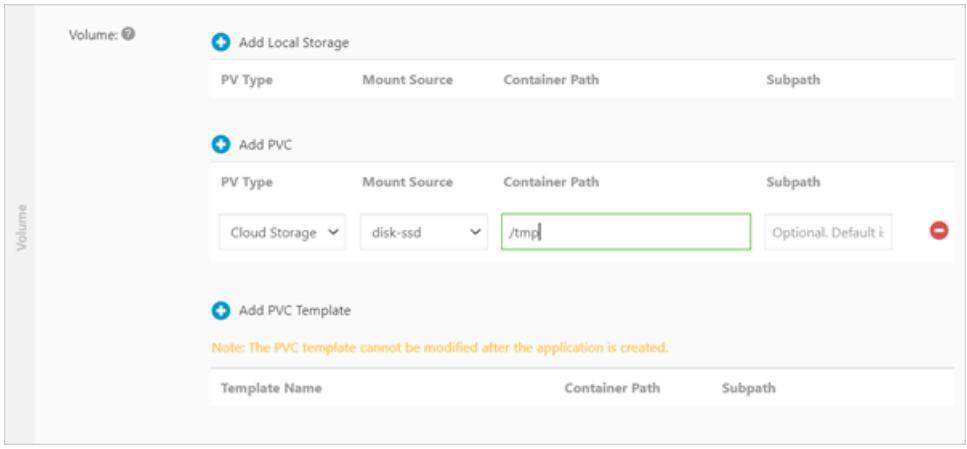
- **Start**: Set the command and parameter that take effect before the container starts.
- **Post Start**: Set the command that takes effect after the container starts.
- **Pre Stop**: Set the command that takes effect before the container stops.

- (Optional)Volume

  You can mount local volumes and persistent volume claims (PVCs) to the container.

  - **Add Local Storage**: You can select HostPath, ConfigMap, Secret, and EmptyDir. The specified volume is mounted to a path in the container. For more information, see Volumes.
  - **Add PVC**: You can select Cloud Storage.

  In this example, a PVC named disk-ssd is mounted to the */tmp* path of the container.



- (Optional)Log

  Configure **Log Service**. You can specify collection configurations and add custom tags.

  Notice Make sure that the Log Service agent is installed in the cluster.

Parameter	Description
	Logstore: creates a Logstore in Log Service to store collected log data.

Collection Configuration Parameter	Description
	Log Path in Container: specifies stdout or a path to collect log data. ■ stdout: specifies that the stdout files are collected. ■ Text Logs: specifies that log data in the specified path of the container is collected. In this example, */var/log/nginx* is specified as the path. Wildcard characters can be used to specify the path.
Custom Tag	You can also add custom tags. Custom tags are added to the log data of the container when the log data is collected. Log data with tags is easier to aggregate and filter.

9. Configure the parameters based on your business requirements and click **Next**.
10. (Optional)Configure advanced settings.
    - Access Control

> **Note**
>
> You can configure the following access control settings based on your business requirements:
>
> - Internal applications: For applications that run inside the cluster, you can create a ClusterIP or NodePort Service to enable internal communication.
> - External applications: For applications that are open to the Internet, you can configure access control by using one of the following methods:
>   - Create a LoadBalancer Service and enable access to your application over the Internet by using a Server Load Balancer (SLB) instance.
>   - Create an Ingress and use the Ingress to expose your application to the Internet. For more information, see Ingress.

You can also specify how the backend pods are exposed to the Internet. In this example, a ClusterIP Service and an Ingress are created to expose the NGINX application to the Internet.

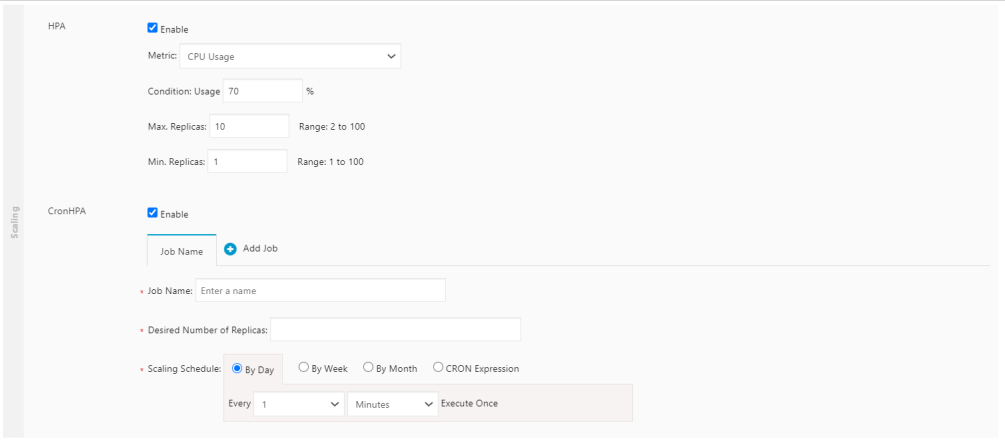
Parameter	Description
Services	Click **Create** on the right side of **Services**. In the Create Service dialog box, set the parameters. For more information about the parameters that are required to create a Service, see Manage Services. **Cluster IP** is selected in this example.

Parameter	Description
Ingresses	Click **Create** on the right side of **Ingresses**. In the Create dialog box, set the parameters. For more information, see Create an Ingress. **Note** When you deploy an application from an image, you can create an Ingress only for one Service. In this example, a virtual hostname is used as the test domain name. You must add the following entry to the hosts file to map the domain name to the IP address of the Ingress. In actual scenarios, use a domain name that has obtained an Internet Content Provider (ICP) number. `101.37.224.146 foo.bar.com #The IP address of the Ingress.`

You can find the created Service and Ingress in the **Access Control** section. You can click **Update** or **Delete** to change the configurations.

- Scaling

  In the **Scaling** section, specify whether to enable **HPA** and **CronHPA**. This allows you to meet the resource requirements of the application at different load levels.



- HPA can automatically scale the number of pods in an ACK cluster based on the CPU and memory usage.

  > **Note** To enable HPA, you must configure required resources for the container. Otherwise, HPA does not take effect.

Parameter	Description
Metric	Select CPU Usage or Memory Usage. The selected resource type must be the same as that specified in the Required Resources field.
Condition	Specify the resource usage threshold. HPA triggers scaling activities when the threshold is exceeded.
Max. Replicas	Specify the maximum number of pod replicas to which the application can be scaled.
Min. Replicas	Specify the minimum number of pod replicas that must run.

- CronHPA can scale an ACK cluster at a scheduled time. For more information about CronHPA, see Create CronHPA jobs.

- Scheduling

  You can set the following parameters: Update Method, Node Affinity, Pod Affinity, Pod Anti Affinity, and Toleration. For more information, see Affinity and anti-affinity.

  > **Note** Node affinity and pod affinity affect pod scheduling based on node labels and pod labels. You can add node labels and pod labels that are provided by Kubernetes to configure node affinity and pod affinity. You can also add custom labels to nodes and pods, and then configure node affinity and pod affinity based on these custom labels.

Parameter	Description
Update Method	Select Rolling Update or OnDelete. For more information, see Deployments.

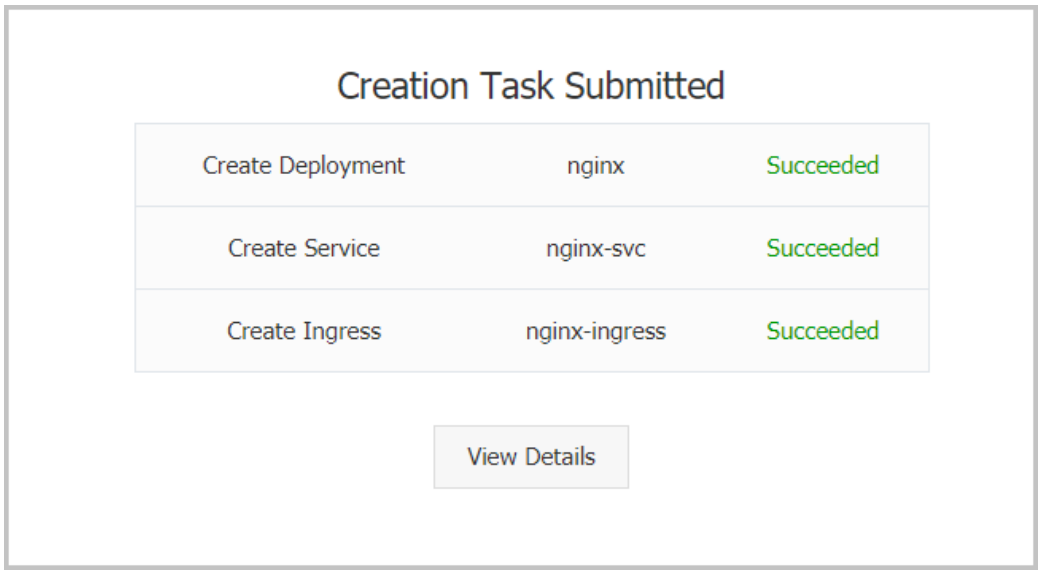
Parameter	Description
Node Affinity	Set **Node Affinity** by adding labels to worker nodes. Node affinity supports required and preferred rules, and various operators, such as In, NotIn, Exists, DoesNotExist, Gt, and Lt. ■ **Required**: Specify the rules that must be matched for pod scheduling. In the YAML file, these rules are defined by the requiredDuringSchedulingIgnoredDuringExecution field of the nodeAffinity parameter. These rules have the same effect as the `NodeSelector` parameter. In this example, pods can be scheduled only to nodes with the specified labels. You can create multiple required rules. However, only one of them must be met. ■ **Preferred**: Specify the rules that are not required to be matched for pod scheduling. Pods are scheduled to a node that matches the preferred rules when multiple nodes match the required rules. In the YAML file, these rules are defined by the preferredDuringSchedulingIgnoredDuringExecution field of the nodeAffinity parameter. In this example, the scheduler attempts to schedule a pod to a node that matches the preferred rules. You can also set weights for preferred rules. If multiple nodes match the rule, the node with the highest weight is preferred. You can create multiple preferred rules. However, all of them must be met before the pod can be scheduled.

Parameter	Description
Pod Affinity	Pod affinity rules specify how pods are deployed relative to other pods in the same topology domain. For example, you can use pod affinity to deploy services that communicate with each other to the same topological domain, such as a host. This reduces the network latency between these services. Pod affinity enables you to select nodes to which pods can be scheduled based on the labels of other running pods. Pod affinity supports required and preferred rules, and the following operators: `In, NotIn, Exists, and DoesNotExist`. ■ **Required**: Specify rules that must be matched for pod scheduling. In the YAML file, these rules are defined by the requiredDuringSchedulingIgnoredDuringExecution field of the podAffinity parameter. A node must match the required rules before pods can be scheduled to the node. ■ **Namespace**: Specify the namespace to apply the required rule. Pod affinity rules are defined based on the labels that are added to pods and therefore must be scoped to a namespace. ■ **Topological Domain**: Set the topologyKey. This specifies the key for the node label that the system uses to denote the topological domain. For example, if you set the parameter to `kubernetes.io/hostname`, topologies are determined by nodes. If you set the parameter to `beta.kubernetes.io/os`, topologies are determined by the operating systems of nodes. ■ **Selector**: Click Add to add pod labels. ■ **View Applications**: Click **View Applications** and set the namespace and application in the dialog box that appears. You can view the pod labels on the selected application and select the labels as selectors. ■ Required Rules: Specify labels on existing applications, the operator, and the label value. In this example, the required rule specifies that the application to be created is scheduled to a host that runs applications with the `app:nginx` label. ■ **Preferred**: Specify rules that are not required to be matched for pod scheduling. In the YAML file, preferred rules are defined by the preferredDuringSchedulingIgnoredDuringExecution field of the podAffinity parameter. The scheduler attempts to schedule the pod to a node that matches the preferred rules. You can set weights for preferred rules. The other parameters are the same as those of required rules. **Note** **Weight**: Set the weight of a preferred rule to a value from 1 to 100. The scheduler calculates the weight of each node that meets the preferred rule based on an algorithm, and then schedules the pod to the node with the highest weight.

Parameter	Description
Pod Anti Affinity	Pod anti-affinity rules specify that pods are not scheduled to topological domains where pods with matching labels are deployed. Pod anti-affinity rules apply to the following scenarios: ■ Schedule the pods of an application to different topological domains, such as multiple hosts. This allows you to enhance the stability of the service. ■ Grant a pod exclusive access to a node. This enables resource isolation and ensures that no other pod can share the resources of the specified node. ■ Schedule pods of an application to different hosts if the pods may interfere with each other. **Note** The parameters of pod anti-affinity rules are the same as those of pod affinity rules. You can create the rules for different scenarios.
Toleration	Configure toleration rules to allow pods to be scheduled to nodes with matching taints.
Schedule to Virtual Nodes	Specify whether to schedule pods to virtual nodes. This option is unavailable if the cluster does not contain a virtual node.

- Labels and Annotations
  - Pod Labels: Add a label to the pod. The label is used to identify the application.
  - Pod Annotations: Add an annotation to the pod.

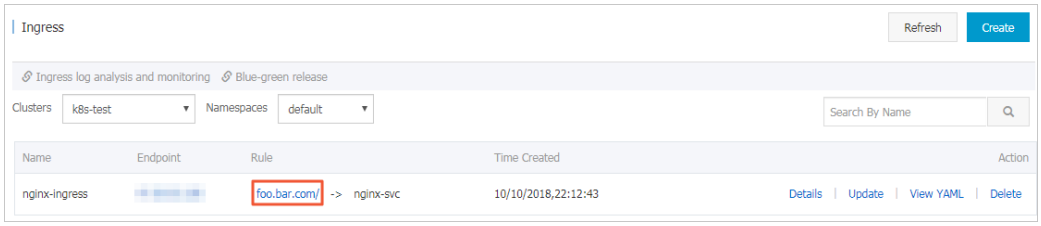
11. Click **Create**.
12. After the application is created, you are redirected to the Complete wizard page. You can find the resource objects under the application and click **View Details** to view application details.



The nginx-deployment details page appears.

> **Note** You can also perform the following steps to create an Ingress and Service: In the **Access Control** section:
>
> - Click **Create** on the right side of **Services**. For more information, see Manage Services.
> - Click **Create** on the right side of **Ingresses**. For more information, see Manage Ingresses in the ACK console.

13. Return to the details page of the cluster. In the left-side navigation pane, click **Ingresses**. You can find the created Ingress on the Ingresses page.



14. Enter the test domain name in the address bar of your browser and press Enter. The NGINX welcome page appears.

4.3. Use a StatefulSet to create a stateful application

Container Service for Kubernetes (ACK) allows you to create stateful applications by using the ACK console. This topic provides an example on how to create a stateful NGINX application and demonstrates the features of StatefulSets.

Prerequisites

Before you deploy a stateful application from an image, make sure that you have performed the following steps:

- Create a managed Kubernetes cluster
- Create a PVC
- Connect to ACK clusters by using kubectl

Background information

StatefulSets provide the following features:

Feature	Description
Pod consistency	Pod consistency ensures that pods are started and terminated in the specified order and ensures network consistency. Pod consistency is determined by pod configurations, regardless of the node to which a pod is scheduled.
Stable and persistent storage	VolumeClaimTemplate allows you to mount a persistent volume (PV) to each pod. The mounted PVs are not deleted after you delete or scale in the number of pod replicas.
Stable network identifiers	Each pod in a StatefulSet derives its hostname from the name of the StatefulSet and the ordinal of the pod. The pattern of the hostname is `StatefulSet name-pod ordinal`.
Stable orders	For a StatefulSet with N pod replicas, each pod is assigned an integer ordinal from 0 to N-1. The ordinals assigned to pods within the StatefulSet are unique.

Procedure

1. Log on to the ACK console.
2. In the left-side navigation pane of the ACK console, click **Clusters**.
3. On the **Clusters** page, find the cluster that you want to manage and click the name of the cluster or click **Details** in the **Actions** column. The details page of the cluster appears.
4. In the left-side navigation pane of the details page, choose **Workloads > StatefulSets**.
5. In the upper-right corner of the **StatefulSets** page, click **Create from Image**.
6. On the **Basic Information** wizard page, configure the basic settings.

   In this example, the Type parameter is set to **StatefulSet** to deploy a stateful application.

Parameter	Description
Name	The name of the application.
Namespace	The namespace where you want to deploy the application. The default namespace is automatically selected. You can select another namespace.
Replicas	The number of pods that are provisioned for the application.
Type	The type of the application. You can select **Deployment**, **StatefulSet**, **Job**, **CronJob**, or **DaemonSet**.
Label	Add a label to the application. The label is used to identify the application.
Annotations	Add an annotation to the application.
Synchronize Timezone	Specify whether to synchronize the time zone between nodes and containers.

7. Click **Next** to proceed to the **Container** wizard page.
8. Configure containers.

   > **Note** In the upper part of the **Container** wizard page, click **Add Container** to add more containers for the application.

   The following table describes the parameters that are required to configure the containers.

   - General settings

Parameter	Description
Image Name	■ You can click **Select Image**. In the dialog box that appears, select an image and click **OK**. In this example, an NGINX image is selected. On the **Search** tab, select **Docker Images** from the drop-down list, enter *NGINX* into the search box, and then click Search. ■ Images from Container Registry: On the **Alibaba Cloud Container Registry** tab, you can select an image from Container Registry. You must select the region and the Container Registry instance to which the image belongs. For more information about Container Registry, see What is Container Registry?. **Note** On the **Alibaba Cloud Container Registry** tab, you can search for images by name. ■ Docker Official Images: On the **Docker Official Images** tab, you can select a Docker image. ■ Favorite Images: On the **Favorite Images** tab, you can select a Docker image that you have added to your favorite list. ■ Search: On the **Search** tab, you can select Alibaba Cloud Image from the drop-down list and specify a region to search for an image in Container Registry. You can also select Docker Images from the drop-down list and search for a Docker image. ■ You can also enter the address of a private registry. The registry address must be in the `domainname/namespace/imagename:tag` format.

Parameter	Description
Image Version	■ Click **Select Image Version** and select an image version. If you do not specify an image version, the latest image version is used. ■ You can select the following image pull policies: ■ **ifNotPresent**: If the image that you want to pull is found on your on-premises machine, the image on your on-premises machine is used. Otherwise, ACK pulls the image from the corresponding repository. ■ **Always**: ACK pulls the image from Container Registry each time the application is deployed or scaled out. ■ **Never**: ACK uses only images on your on-premises machine. **Note** If you select **Image Pull Policy**, no image pull policy is applied. ■ To pull the image without a password, click **Set Image Pull Secret** to set a Secret that is used to pull the image. For more information, see Use the aliyun-acr-credential-helper component to pull images without a password.
Resource Limit	You can specify an upper limit for the CPU, memory, and ephemeral storage space that the container can consume. This prevents the container from occupying an excess amount of resources. The CPU resource is measured in millicores (one thousandth of one core). The memory resource is measured in MiB. The ephemeral storage resource is measured in GiB.
Required Resources	The amount of CPU and memory resources that are reserved for this application. These resources are exclusive to the container. This prevents the application from becoming unavailable if other services or processes compete for computing resources.
Container Start Parameter	■ stdin: Pass stdin to the container. ■ tty: Stdin is a TeleTYpewriter (TTY).
Privileged Container	■ If you select Privileged Container, privileged=true is set for the container and the privilege mode is enabled. ■ If you do not select Privileged Container, privileged=false is set for the container and the privilege mode is disabled.
Init Container	If you select Init Container, an init container is created. An init container contains useful utilities. For more information, see Init Containers.

- (Optional)Ports

  Configure container ports.

  ■ Name: Enter a name for the port.

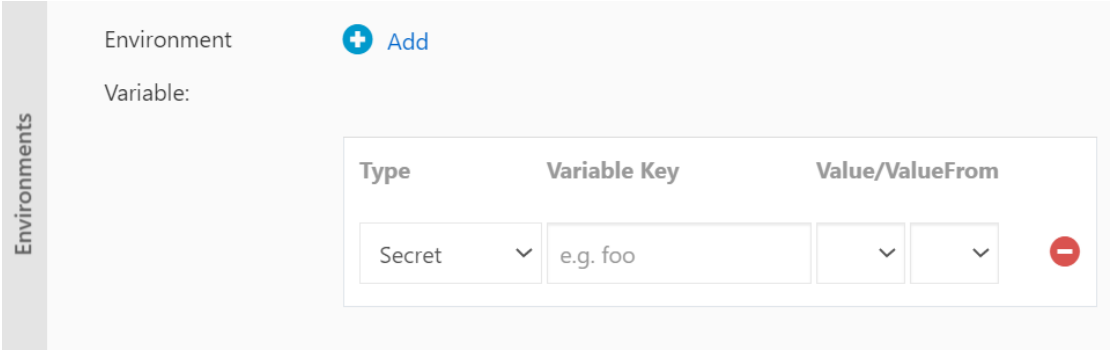
- Container Port: Enter the container port that you want to open. Enter a port number from 1 to 65535.
- Protocol: Select TCP or UDP.

- (Optional)Environments

  You can configure environment variables for pods in key-value pairs. Environment variables are used to apply pod configurations to containers. For more information, see Pod variables.

  - Type: Select the type of environment variable. You can select **Custom**, **ConfigMaps**, **Secrets**, or **Value/ValueFrom**. If you select ConfigMaps or Secrets as the type of environment variable, all values in the selected ConfigMap or Secret are passed to the container environment variables. In this example, Secrets is selected.

    Select **Secrets** from the Type drop-down list and select a Secret from the **Value/ValueFrom** drop-down list. All values in the selected Secret are passed to the environment variable.

    In this case, the *YAML* file used to deploy the application contains the settings that reference all values in the specified Secret.

    ```
    envFrom:
      - secretRef:
          name: test
    ```

  - Variable Key: Specify the key of the environment variable.
  - Value/ValueFrom: Specify the value that is referenced by the environment variable.

- (Optional)Health Check

  Health check settings include liveness and readiness probes. Liveness probes determine when to restart the container. Readiness probes determine whether the container is ready to accept network traffic. For more information about health checks, see Configure Liveness, Readiness, and Startup Probes.

Request type	Description

Request type	Description
HTTP	Sends an HTTP GET request to the container. You can configure the following parameters: ■ Protocol: HTTP or HTTPS. ■ Path: Enter the requested path on the server. ■ Port: Enter the container port that you want to open. Enter a port number from 1 to 65535. ■ HTTP Header: Enter the custom headers in the HTTP request. Duplicate headers are allowed. Key-value pairs are supported. ■ Initial Delay (s): the initialDelaySeconds field in the YAML file. This field specifies the wait time (in seconds) before the first probe is performed after the container is started. Default value: 3. ■ Period (s): the periodSeconds field in the YAML file. This field specifies the time interval (in seconds) at which probes are performed. Default value: 10. Minimum value: 1. ■ Timeout (s): the timeoutSeconds field in the YAML file. This field specifies the time (in seconds) after which the probe times out. Default value: 1. Minimum value: 1. ■ Healthy Threshold: the minimum number of consecutive successes that must occur before a container is considered healthy after a failed probe. Default value: 1. Minimum value: 1. For liveness probes, this parameter must be set to 1. ■ Unhealthy Threshold: the minimum number of consecutive failures that must occur before a container is considered unhealthy after a success. Default value: 3. Minimum value: 1.

Request type	Description
TCP	Sends a TCP socket to the container. kubelet attempts to open the socket on the specified port. If the connection can be established, the container is considered healthy. Otherwise, the container is considered unhealthy. You can configure the following parameters: ■ Port: Enter the container port that you want to open. Enter a port number that ranges from 1 to 65535. ■ Initial Delay (s): the initialDelaySeconds field in the YAML file. This field specifies the wait time (in seconds) before the first probe is performed after the container is started. Default value: 15. ■ Period (s): the periodSeconds field in the YAML file. This field specifies the time interval (in seconds) at which probes are performed. Default value: 10. Minimum value: 1. ■ Timeout (s): the timeoutSeconds field in the YAML file. This field specifies the time (in seconds) after which the probe times out. Default value: 1. Minimum value: 1. ■ Healthy Threshold: the minimum number of consecutive successes that must occur before a container is considered healthy after a failed probe. Default value: 1. Minimum value: 1. For liveness probes, this parameter must be set to 1. ■ Unhealthy Threshold: the minimum number of consecutive failures that must occur before a container is considered unhealthy after a success. Default value: 3. Minimum value: 1.
Command	Runs a probe command in the container to check the health status of the container. You can configure the following parameters: ■ Command: Enter the probe command that is run to check the health status of the container. ■ Initial Delay (s): the initialDelaySeconds field in the YAML file. This field specifies the wait time (in seconds) before the first probe is performed after the container is started. Default value: 5. ■ Period (s): the periodSeconds field in the YAML file. This field specifies the interval (in seconds) at which probes are performed. Default value: 10. Minimum value: 1. ■ Timeout (s): the timeoutSeconds field in the YAML file. This field specifies the time (in seconds) after which the probe times out. Default value: 1. Minimum value: 1. ■ Healthy Threshold: the minimum number of consecutive successes that must occur before a container is considered healthy after a failed probe. Default value: 1. Minimum value: 1. For liveness probes, this parameter must be set to 1. ■ Unhealthy Threshold: the minimum number of consecutive failures that must occur before a container is considered unhealthy after a success. Default value: 3. Minimum value: 1.

- Lifecycle

  You can set the following parameters to configure the lifecycle of the container: Start, Post Start, and Pre Stop. For more information, see Attach Handlers to Container Lifecycle Events.

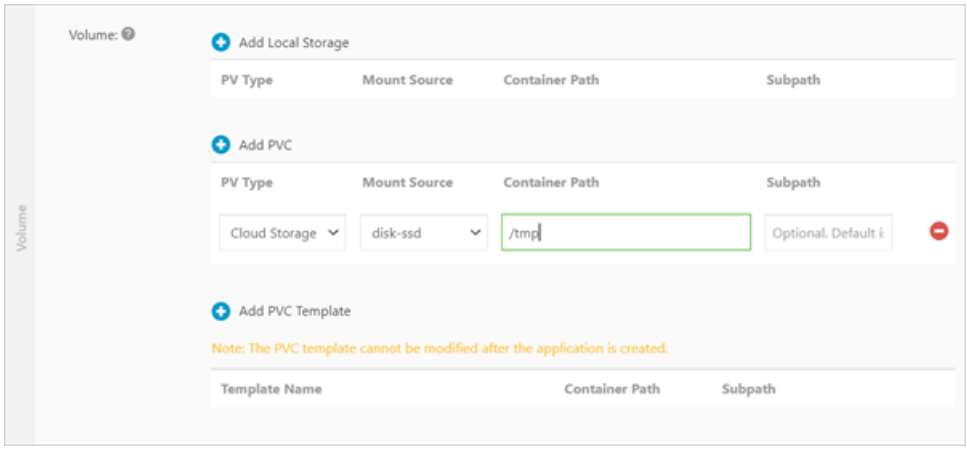
  - **Start**: Set the command and parameter that take effect before the container starts.
  - **Post Start**: Set the command that takes effect after the container starts.
  - **Pre Stop**: Set the command that takes effect before the container stops.

- (Optional)Volume

  You can mount local volumes and persistent volume claims (PVCs) to the container.

  - **Add Local Storage**: You can select HostPath, ConfigMap, Secret, and EmptyDir. The specified volume is mounted to a path in the container. For more information, see Volumes.
  - **Add PVC**: You can select Cloud Storage.

  In this example, a PVC named disk-ssd is mounted to the */tmp* path of the container.



- (Optional)Log

  Configure **Log Service**. You can specify collection configurations and add custom tags.

> **Notice** Make sure that the Log Service agent is installed in the cluster.

Parameter	Description
	Logstore: creates a Logstore in Log Service to store collected logs.

Parameter Collection Configuration	Description
	Log Path in Container: specifies stdout or a path to collect logs. ■ stdout: specifies that the stdout files are collected. ■ Text Logs: specifies that logs in the specified path of the container are collected. In this example, */var/log/nginx* is specified as the path. Wildcard characters can be used in the path.
Custom Tag	You can also add custom tags. Custom tags are added to the log of the container when the log is collected. Custom tags provide an easy method to filter collected logs and perform statistical analytics.

9. Set the parameters based on your business requirements and click **Next**.
10. (Optional)Configure advanced settings.
    - Access Control

> **Note**
>
> You can configure the following access control settings based on your business requirements:
>
> - Internal applications: For applications that run inside the cluster, you can create a ClusterIP or NodePort Service to enable internal communication.
> - External applications: For applications that are exposed to the Internet, you can configure access control by using one of the following methods:
>   - Create a LoadBalancer Service and enable access to your application over the Internet by using a Server Load Balancer (SLB) instance.
>   - Create an Ingress and use the Ingress to expose your application to the Internet. For more information, see Ingress.

You can also specify how the backend pods are exposed to the Internet. In this example, a ClusterIP Service and an Ingress are created to expose the NGINX application to the Internet.

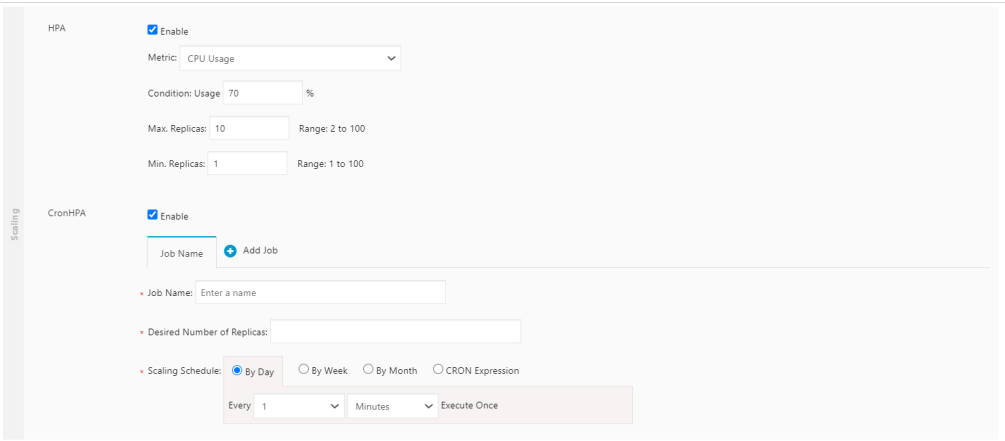
Parameter	Description
Services	Click **Create** on the right side of **Service**. In the **Create Service** dialog box, set the parameters. For more information about the parameters, see Manage Services. **Cluster IP** is selected in this example.

Parameter	Description
Ingresses	Click **Create** on the right side of **Ingresses**. In the Create dialog box, set the parameters. For more information about how to configure an Ingress, see Create an Ingress. **Note** When you deploy an application from an image, you can create an Ingress only for one Service. In this example, a virtual hostname is used as the test domain name. You must add the following entry to the hosts file to map the domain name to the IP address of the Ingress. In actual scenarios, use a domain name that has obtained an ICP number. `101.37.224.146 foo.bar.com #The IP address of the Ingress.`

You can find the created Service and Ingress in the **Access Control** section. You can click **Update** or **Delete** to change the configurations.

- Scaling

  In the **Scaling** section, specify whether to enable **HPA** and **CronHPA**. Horizontal Pad Autoscaler (HPA) allows you to meet the resource requirements of the application at different load levels.



- HPA can automatically scale the number of pods in an ACK cluster based on the CPU and memory usage.

  > **Note** To enable HPA, you must configure required resources for the container. Otherwise, HPA does not take effect.

Parameter	Description
Metric	Select CPU Usage or Memory Usage. The selected resource type must be the same as that specified in the Required Resources field.
Condition	Specify the resource usage threshold. HPA triggers scale-out activities when the threshold is exceeded.
Max. Replica	Specify the maximum number of pod replicas to which the application can be scaled.
Min. Replica	Specify the minimum number of pod replicas that must run.

- CronHPA can scale an ACK cluster at a scheduled time. For more information about CronHPA, see Create CronHPA jobs.

- Scheduling

  You can set the following parameters: Update Method, Node Affinity, Pod Affinity, Pod Anti Affinity, and Toleration. For more information, see Affinity and anti-affinity.

  > **Note** Node affinity and pod affinity affect pod scheduling based on node labels and pod labels. You can add node labels and pod labels that are provided by Kubernetes to configure node affinity and pod affinity. You can also add custom labels to nodes and pods, and then configure node affinity and pod affinity based on these custom labels.

Parameter	Description
Update Method	Select Rolling Update or OnDelete. For more information, see Deployments.

Parameter	Description
Node Affinity	Add labels to worker nodes to set **Node Affinity**. Node Affinity supports required and preferred rules, and various operators, such as In, NotIn, Exists, DoesNotExist, Gt, and Lt. ■ **Required**: Specify the rules that must be matched for pod scheduling. In the YAML file, these rules are defined by the requiredDuringSchedulingIgnoredDuringExecution field of the nodeAffinity parameter. These rules have the same effect as the `NodeSelector` parameter. In this example, pods can be scheduled only to nodes with the specified labels. You can create multiple required rules. However, only one of them must be met. ■ **Preferred**: Specify the rules that are not required to be matched for pod scheduling. Pods are scheduled to a node that matches the preferred rules when multiple nodes match the required rules. In the YAML file, these rules are defined by the preferredDuringSchedulingIgnoredDuringExecution field of the nodeAffinity parameter. In this example, the scheduler attempts to not schedule a pod to a node that matches the preferred rules. You can also set weights for preferred rules. If multiple nodes match the rule, the node with the highest weight is preferred. You can create multiple preferred rules. However, all of them must be met before the pod can be scheduled.

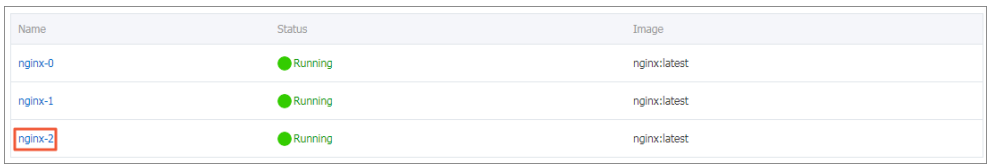
Parameter	Description
Pod Affinity	Pod affinity rules specify how pods are deployed relative to other pods in the same topology domain. For example, you can use pod affinity to deploy services that communicate with each other to the same topological domain, such as a host. This reduces the network latency between these services. Pod affinity enables you to specify to which node pods can be scheduled based on the labels on other running pods. Pod affinity supports required and preferred rules, and the following operators: `In, NotIn, Exists, and DoesNotExist`. ■ **Required**: Specify rules that must be matched for pod scheduling. In the YAML file, these rules are defined by the requiredDuringSchedulingIgnoredDuringExecution field of the podAffinity parameter. A node must match the required rules before pods can be scheduled to the node. ■ **Namespace**: Specify the namespace to apply the required rule. Pod affinity rules are defined based on the labels that are added to pods and therefore must be scoped to a namespace. ■ **Topological Domain**: Set the topologyKey. This specifies the key for the node label that the system uses to denote the topological domain. For example, if you set the parameter to `kubernetes.io/hostname`, topologies are determined by nodes. If you set the parameter to `beta.kubernetes.io/os`, topologies are determined by the operating systems of nodes. ■ **Selector**: Click Add to add pod labels. ■ **View Applications**: Click **View Applications** and set the namespace and application in the dialog box that appears. You can view the pod labels on the selected application and add the labels as selectors. ■ Required Rules: Specify labels on existing applications, the operator, and the label value. In this example, the required rule specifies that the application to be created is scheduled to a host that runs applications with the `app:nginx` label. ■ **Preferred**: Specify rules that are not required to be matched for pod scheduling. In the YAML file, preferred rules are defined by the preferredDuringSchedulingIgnoredDuringExecution field of the podAffinity parameter. The scheduler attempts to schedule the pod to a node that matches the preferred rules. You can set weights for preferred rules. The other parameters are the same as those of required rules. **Note** **Weight**: Set the weight of a preferred rule to a value from 1 to 100. The scheduler calculates the weight of each node that meets the preferred rule based on an algorithm, and then schedules the pod to the node with the highest weight.

Parameter	Description
Pod Anti Affinity	Pod anti-affinity rules specify that pods are not scheduled to topological domains where pods with matching labels are deployed. Pod anti-affinity rules apply to the following scenarios: ■ Schedule the pods of an application to different topological domains, such as multiple hosts. This allows you to enhance the stability of the service. ■ Grant a pod exclusive access to a node. This enables resource isolation and ensures that no other pods can share the resources of the specified node. ■ Schedule pods of an application to different hosts if the pods may interfere with each other. **Note** The parameters of pod anti-affinity rules are the same as those of pod affinity rules. You can create the rules for different scenarios.
Toleration	Configure toleration rules to allow pods to be scheduled to nodes with matching taints.
Schedule to Virtual Nodes	Specify whether to schedule pods to virtual nodes. This parameter is unavailable if the cluster does not contain a virtual node.

- Labels and Annotations
  - Pod Labels: Add a label to the pod. The label is used to identify the application.
  - Pod Annotations: Add an annotation to the pod.

11. Click **Create**.
12. After the application is created, you are redirected to the Complete wizard page. You can find the resource objects under the application and click **View Details** to view application details.

    The details page of the created stateful application appears.
13. In the upper-left corner of the page, click the **Back** icon to go to the StatefulSets page. On the StatefulSets page, you can view the created application.
14. (Optional)Click **Scale** in the Actions column to scale the application.
    i. In the Scale dialog box, set Desired Number of Pods to 3 and click OK. After you scale out the application, all pods in the application are listed in ascending order of ordinal indexes. If you scale in the application, pods are deleted in descending order of ordinal indexes. This ensures that all pods follow a specific order.



ii. In the left-side navigation pane, choose **Volumes > Persistent Volume Claims**. Verify that after you scale out the application, new PVs and PVCs are created for the newly added pods. However, if the application is scaled in, existing PVs and PVCs are not deleted.

Related operations

In the left-side navigation pane, click **Clusters**. On the Clusters page, click the name of the cluster where the application is deployed or click **Applications** in the **Actions** column. In the left-side navigation pane, choose **Workloads** > **StatefulSets**. On the **StatefulSets** page, click the name of the application that you want to manage or click **Details** in the **Actions** column. On the details page of the application, you can **edit**, **scale**, **redeploy**, and **refresh** the application. You can also **view the YAML file** of the application.

- Edit: On the details page of the application, click **Edit** in the upper-right corner of the page to modify the configurations of the application.
- Scale: On the details page of the application, click **Scale** in the upper-right corner of the page to scale the application to a required number of pods.
- View in YAML: On the details page of the application, click **View in YAML** in the upper-right corner of the page. You can **Update** and **Download** the YAML file. You can also click **Save As** to save the *YAML* file as a different name.
- Redeploy: On the details page of the application, click **Redeploy** in the upper-right corner of the page to redeploy the application.
- Refresh: On the details page of the application, click **Refresh** in the upper-right corner of the page to refresh the application details page.

What's next

Log on to a master node and run the following commands to test persistent storage.

1. Run the following commands to create a temporary file in the disk that is mounted to pod nginx-1:

```
kubectl exec nginx-1 ls /tmp        #Query files in the /tmp directory.
kubectl exec nginx-1 touch /tmp/statefulset      #Create a file named statefulset.
kubectl exec nginx-1 ls /tmp
lost+found
statefulset
```

2. Run the command to delete pod nginx-1 and verify data persistence:

```
kubectl delete pod nginx-1
pod"nginx-1" deleted
```

3. After the system recreates and starts a new pod, query the files in the /tmp directory. The following result shows that the statefulset file still exists. This shows the high availability of the stateful application.

```
kubectl exec nginx-1 ls /tmp              #Query files in the /tmp directory.
statefulset
```

4.4. Deploy WordPress from App Catalog

This topic describes how to deploy a WordPress application in a Container Service for Kubernetes (ACK) cluster by using App Catalog.

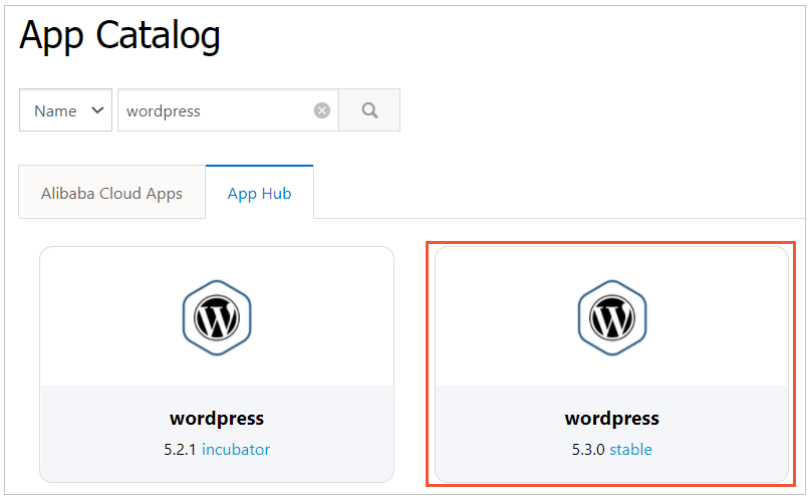
Prerequisites

An ACK cluster is created. For more information, see Create a managed Kubernetes cluster.

> **Note** You can deploy a WordPress application by using App Catalog only in ACK clusters. Serverless Kubernetes (ASK) clusters are not supported.

Procedure

1. Log on to the ACK console.
2. In the left-side navigation pane, choose **Marketplace > App Catalog**.
3. On the **App Catalog** page, click the **App Hub** tab. Find **wordpress** and click **wordpress 5.3.0**.



4. On the **wordpress** page, set **Cluster**, **Namespace**, and **Release Name** in the **Deploy** section.
5. Click the **Parameters** tab, set persistence to false, and then click **Create**.
   - Set persistence to false, as shown in the following figure.

```
196  master:
197    persistence:
198      enabled: false
```

   - Set persistence to false, as shown in the following figure.

```
323 persistence:
324   enabled: false
```

6. In the left-side navigation pane of the ACK console, click **Clusters**.
7. On the **Clusters** page, find the cluster that you want to manage and click the name of the cluster or click **Details** in the **Actions** column. The details page of the cluster appears.
8. In the left-side navigation pane of the details page, choose **Network > Services**
9. On the **Services** page, find the Service that is created for the WordPress application and click the hyperlink in the External Endpoint column. The WordPress homepage appears.

Note To modify the parameters after the WordPress application is deployed, delete and redeploy the application. In the left-side navigation pane of the cluster details page, choose **Applications > Helm**. On the Helm tab, find and delete the WordPress release. Then, perform the preceding steps to redeploy the WordPress application.

4.5. Deploy the WordPress application from Helm CLI

This topic describes how to deploy the WordPress application in a cluster of Container Service for Kubernetes (ACK) from Helm command-line interface (CLI).

Prerequisites

- Create a managed Kubernetes cluster
- Install and set up kubectl.

Install and set up Alibaba Cloud CLI and kubectl

- Install Helm. You can install Helm in the following ways:
  - Use the source code or a binary package to install Helm. For more information, see Installing Helm.
  - Run the following script to install Helm:

    ```
    curl -sSL https://raw.githubusercontent.com/helm/helm/master/scripts/get-helm-3 | bash
    ```

  - Use a package manager to install Helm.
    - Install Helm in macOS. For more information, see Homebrew. After the packet manager is installed, run the following command to install Helm:

      ```
      brew install helm
      ```

    - Install Helm in Windows. For more information, see Chocolatey. After the packet manager is installed, run the following command to install Helm:

      ```
      choco install kubernetes-helm
      ```

- Create a Helm repository. Alibaba Cloud provides Helm repository images for developers in China. This provides an easy way to use Helm.
  - Run the following command to create a stable repository from an image that is provided by Alibaba Cloud:

    ```
    helm repo add stable https://apphub.aliyuncs.com/stable/
    ```

    Note Users outside China can use official Helm repository images. Run the following command to create a stable repository from an official image:

    ```
    helm repo add stable https://kubernetes-charts.storage.googleapis.com/
    ```

  - Run the following command to search for the stable repository.

    ```
    helm search repo stable
    ```

- Install and set up the kubectl client. For more information, see Install and Set Up kubectl.

Deploy WordPress

1. Run the following command to add an official Helm repository:

   ```
   helm repo add bitnami https://charts.bitnami.com/bitnami
   ```

2. Run the following command to deploy WordPress:

   ```
   helm install wordpress bitnami/wordpress \
     --set mariadb.master.persistence.enabled=true \
     --set mariadb.master.persistence.storageClass=alicloud-disk-ssd \
     --set mariadb.master.persistence.size=20Gi \
     --set persistence.enabled=false
   ```

   Note
   - MariaDB persists data to persistent volumes (PVs).
   - You can provision an SSD of 20 GiB as the PV. The PV can be claimed by setting StorageClassName.
   - If you do not want WordPress to persist data, you can set persistence.enabled to false.

   If information similar to the following output is returned, it indicates that the WordPress application is deployed:

```
NAME: wordpress
LAST DEPLOYED: Tue Sep  8 10:37:05 2020
NAMESPACE: default
STATUS: deployed
REVISION: 1
NOTES:
** Please be patient while the chart is being deployed **
Your WordPress site can be accessed through the following DNS name from within your cluster:
   wordpress.default.svc.cluster.local (port 80)
To access your WordPress site from outside the cluster follow the steps below:
1. Get the WordPress URL by running these commands:
 NOTE: It may take a few minutes for the LoadBalancer IP to be available.
      Watch the status with: 'kubectl get svc --namespace default -w wordpress'
  export SERVICE_IP=$(kubectl get svc --namespace default wordpress --template "{{ range (index .statu
s.loadBalancer.ingress 0) }}{{.}}{{ end }}")
  echo "WordPress URL: http://$SERVICE_IP/"
  echo "WordPress Admin URL: http://$SERVICE_IP/admin"
2. Open a browser and access WordPress using the obtained URL.
3. Login with the following credentials below to see your blog:
 echo Username: user
 echo Password: $(kubectl get secret --namespace default wordpress -o jsonpath="{.data.wordpress-p
assword}" | base64 --decode)
```

Access the WordPress application

1. After Helm is installed, follow the instructions and run the following command to query the IP address that is used to access the application:

   ```
   kubectl get svc --namespace default wordpress-acs-sample --template "{{ range (index .status.loadBala
   ncer.ingress 0) }}{{.}}{{ end }}"
   ```

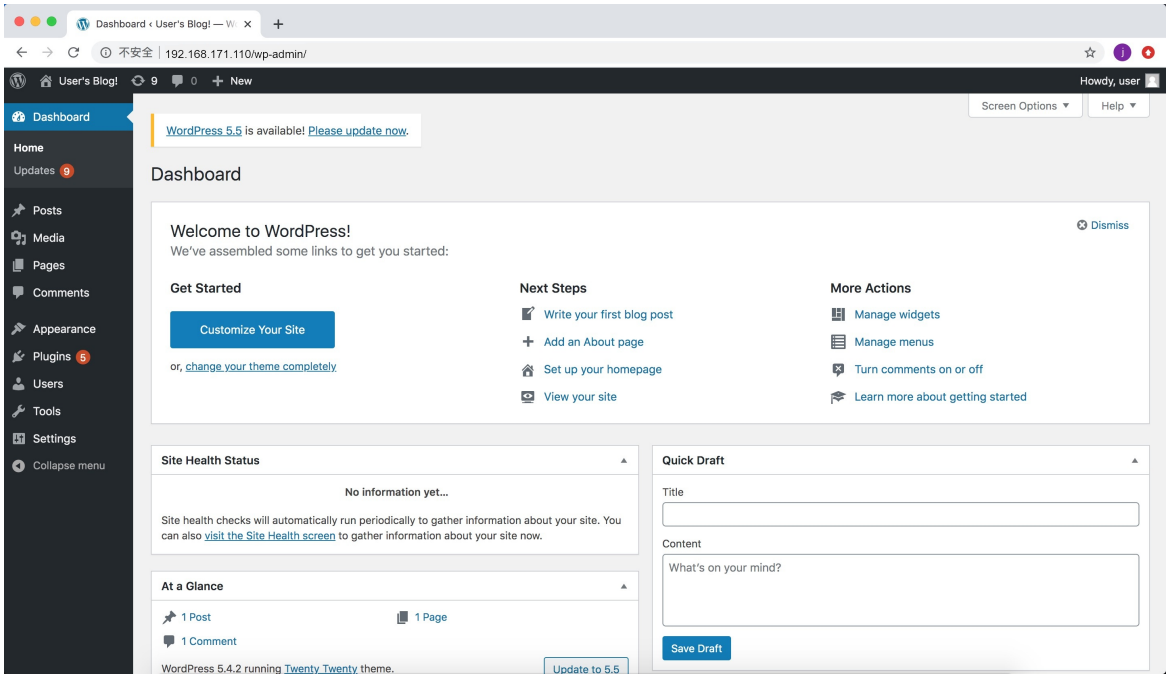
   Sample output:

   ```
   192.168.171.110
   ```

2. Run the following command to query the password of the admin account:

   ```
   kubectl get secret --namespace default wordpress-ack-wordpress-sample -o jsonpath="{.data.wordpre
   ss-password}" | base64 --decode
   K*****X**7
   ```

3. Enter the IP address into the address bar of your browser and press Enter to access the WordPress application. On the logon page, enter admin as the username and enter the obtained password to log on to the application.



Delete the WordPress application

Run the following command to delete the WordPress application:

```
helm delete wordpress
release "wordpress" uninstalled
```

5.Advanced operations

5.1. Create an application by using a private image repository

In many scenarios, you use an image in a private image repository to deploy an application. This topic describes how to create a private image repository in the Container Registry console and use an image in this repository to create an application.

Create a private image repository

If this is the first time you use the Container Registry console, the **Tips** message appears, prompting you to set a password for logging on to the console. Click **Activate Now** and set a password.

1. Log on to the Container Registry console.
2. In the top navigation bar, select a region.
3. In the left-side navigation pane, click **Instances**.
4. On the **Instances** page, click the default instance.
5. In the left-side navigation pane of the management page of the Container Registry Personal Edition instance, choose **Repository > Repositories**.
6. In the upper-left corner of the **Repositories** page, click **Create Repositories**.
7. In the **Repository Info** step, set **Namespace**, **Repository Name**, **Summary**, and **Repository Type**. In this example, the private type is selected. Click **Next**.
8. In the **Code Source** step, select **Local Repository** for Code Source and click **Create Repositories**.

   > **Note** In the repository list, click the name of the created repository. On the **Guide** tab of the **Details** page, you can view information about how to use the private image repository.

9. Run the following command to log on to the image repository:

   > **Note**
   > - If you use an Alibaba Cloud account, the name of the Alibaba Cloud account is the username for logging on to the repository.
   > - If you use a Resource Access Management (RAM) user, the string before .onaliyun.com is the username for logging on to the repository. For example, if the name of your RAM user is 123@1880770869021234.onaliyun.com, the username for logging on to the repository is 123@1880770869021234.

   ```
   sudo docker login --username=<Repository username> registry.cn-<The region where the instance of Container Registry Personal Edition is deployed>.aliyuncs.com
   ```

   In the output, enter the password. If `login succeeded` is displayed, the logon is successful.

10. Run the following command to query the IDs of images in the repository:

```
docker images
```

11. Run the following command to add a tag to an image:

```
sudo docker tag <Image ID> registry.cn-hangzhou.aliyuncs.com/<Namespace>/<Repository name>:[Image version]
```

12. Run the following command to push the image to the repository:

```
sudo docker push registry.cn-hangzhou.aliyuncs.com/<Namespace>/<Repository name>:[Image version]
```

Expected output:

```
The push refers to a repository [registry.cn-hangzhou.aliyuncs.com/XXX/tomcat-private]
9072c7b03a1b: Pushed
f9701cf47c58: Pushed
365c8156ff79: Pushed
2de08d97c2ed: Pushed
6b09c39b2b33: Pushed
4172ffa172a6: Pushed
1dccf0da88f3: Pushed
d2070b14033b: Pushed
63dcf81c7ca7: Pushed
ce6466f43b11: Pushed
719d45669b35: Pushed
3b10514a95be: Pushed
V1: digest: sha256:cded14cf64697961078aedfdf870e704a52270188c8194b6f70c778a8289**** size: 2836
```

Go to the repository details page. In the left-side navigation pane, click **Tags**. Verify that the image is uploaded to the repository. You can also view the image version.

Create a private repository logon Secret

To pull private images, you must use a **private repository logon Secret**.

1. Log on to the ACK console.
2. In the left-side navigation pane of the ACK console, click **Clusters**.
3. On the **Clusters** page, find the cluster that you want to manage and click the name of the cluster or click **Details** in the **Actions** column. The details page of the cluster appears.
4. In the left-side navigation pane of the details page, choose **Configurations > Secrets**.
5. In the upper-right corner of the **Secrets** page, click **Create**.
6. In the **Create** panel, set the parameters and click **OK**.

Parameter	Description
Name	The name of the Secret.

Parameter	Description
Type	The following types of Secret are supported: ○ Opaque: a regular Secret. Enter a key and a value. The value must be encoded in Base64. ○ Private Repository Logon Secret: This type of Secret stores the credentials that are required to pull images from a private image repository. Enter the address, username, and password of the image repository. Note The username is the full name of your Alibaba Cloud account. The password is the one specified when you activated Container Registry. You can go to the **Access Credential** page to change the password. ○ TLS Certificate: Use a Transport Layer Security (TLS) certificate to verify user identities. ■ Cert: Enter the content of the TLS certificate. ■ Key: Enter the private key of the TLS certificate.

After the Secret is created, you are redirected to the Secrets page. You can find the newly created Secret in the list.

> **Note**
>
> You can also create a **private repository logon Secret** by using a CLI. For more information, see Connect to ACK clusters by using kubectl.

Create an application by using a private image repository

1. Log on to the ACK console.
2. In the left-side navigation pane of the ACK console, click **Clusters**.
3. On the **Clusters** page, find the cluster that you want to manage and click the name of the cluster or click **Details** in the **Actions** column. The details page of the cluster appears.
4. In the left-side navigation pane of the details page, choose **Workloads > Deployments**.
5. On the **Deployments** page, click **Create from YAML** in the upper-right corner.

   > **Note** You can also click **Create from Image** to create an application. For more information, see Create an application by using an image pull Secret.

6. Set **Sample Template** to **Custom** and copy the following content to the **Template** section.

```
apiVersion: apps/v1
kind: Deployment
metadata:
 name: private-image
 nameSpace: default
 labels:
  app: private-image
spec:
 replicas: 1
 selector:
  matchLabels:
   app: private-image
 template:
  metadata:
   labels:
    app: private-image
  spec:
   containers:
   - name: private-image
    image: registry.cn-hangzhou.aliyuncs.com/The name of the namespace/tomcat-private:latest
    ports:
    - containerPort: 8080
   imagePullSecrets:
   - name: regsecret
```

7. Click **Create**.
   Go to the Deployments page. You can view the newly created application.

For more information, see Use a private image repository.