ALIBABA CLOUD

Alibaba Cloud

Container Service for Kubernetes

Quick Start

Document Version: 20201111

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.Use ACK for the first time

To use Container Service for Kubernetes (ACK) for the first time, you do not need to activate the service. However, you must assign default roles to the service account. After you perform this operation, the service account can be used to call services such as Elastic Compute Service (ECS), Object Storage Service (OSS), Apsara File Storage NAS, and Server Load Balancer (SLB), create clusters, and store logs. This topic describes how to assign roles to the service account.

Context

The default roles in ACK include:

- AliyunCSManagedLogRole
- AliyunCSManagedCmsRole
- AliyunCSManagedCsiRole
- AliyunCSManagedVKRole
- AliyunCSClusterRole
-
-
- AliyunCSManagedNetworkRole
-
-
- AliyunCSManagedArmsRole

Considerations

- If you used ACK before January 15, 2018, the system automatically assigns the default roles to the service account. For more information about role permissions, see Default roles. If you use a Resource Access Management (RAM) user to access ACK, you must upgrade the RAM policy that is attached to the RAM user. For more information, see Customize RAM policies.
- As of January 15, 2018, new users must assign their Alibaba Cloud accounts the default roles to use ACK. To authorize RAM users to use ACK, a new user must log on to the RAM console. For more information, see Overview.

Procedure

1. Log on to the ACK console.
2. If you have not assigned your Alibaba Cloud account the default roles, click **Go to RAM console**. On the Cloud Resource Access Authorization page, click **Confirm Authorization Policy**.

> **Notice**
>
> - If you use a managed cluster, you must assign KubernetesAuditRole to your account to access your cloud resources.
> - The default policy WorkerRolePolicy that is attached to worker roles in a managed cluster has high permissions. To ensure data security and resource isolation in multi-tenancy scenarios, ACK reduces the permissions of RAM roles in a managed cluster. For more information, see Container Service for Kubernetes reduces the permissions of worker RAM roles in managed clusters.
> - To modify the permission settings of default roles, log on to the RAM console and go to the RAM Roles page. Make sure that ACK is granted the required permissions when you modify the permission settings.

3. After you assign the default roles to your account, refresh the console to use ACK.

Related information

- Default roles
- 快速创建Kubernetes托管版集群
- Deploy stateless applications from images
- Use an image to create a stateful application

2.Flowchart

This topic provides the flowchart and some commonly asked questions (FAQ) when you use Container Service for Kubernetes (ACK).

Procedure

The flowchart consists of the following steps:

1. Assign roles to your Alibaba Cloud account. For more information, see Default roles.For more information about RAM policies and role-based access control (RBAC), see Create a custom RAM policy and Assign RBAC roles to a RAM user.
2. Create a standard managed cluster. For more information, see Create a managed Kubernetes cluster.To create a cluster of another type, see the following topics:
   - Create an ASK cluster.
   - Create a dedicated Kubernetes cluster.
   - Create an ACK Pro cluster.
   - Create a managed edge cluster.
   - Create a managed GPU cluster and Create a dedicated GPU cluster for heterogeneous computing.
   - Create a managed Kubernetes cluster that runs sandboxed containers and Create a dedicated Kubernetes cluster that runs sandboxed containers.
   - Create a managed Kubernetes cluster that supports confidential computing.
3. Deploy an application by using an image or orchestration template.For more information, see Use an image to create a stateless application and Use an orchestration template to create a Linux application.

   > **Note** If your application consists of multiple services 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 a cluster ◦ Expand a cluster ◦ Manage system components	Application deployment ◦ Use an image to create a stateful application ◦ Create a Job application by using an image ◦ Create an application from a private image repository

Cluster O&M	Application O&M
Node maintenance ○ Add existing ECS instances to an ACK cluster ○ Mark a node as unschedulable ○ Manage nodes in batches ○ Attach a data disk to a node	Application and image updates ○ Use an application trigger to redeploy an application ○ 使用免密组件拉取容器镜像 ○ Use kritis-validation-hook to automatically verify the signatures of container images
Node pool management ○ Create a node pool ○ Scale out a node pool ○ Schedule an application pod to a specified node pool	Application scaling ○ Scale an application ○ HPA ○ Vertical pod autoscaling
○ Monitoring Monitor application performance, Monitor application architecture, Event monitoring, Enable ARMS Prometheus, and Deploy Prometheus to a Kubernetes cluster ○ Log management Audit logs, Use Log Service to collect container logs, Use log-pilot to collect logs of a cluster, and Configure Log4jAppender for Kubernetes and Log Service	

FAQ

- **How do I build a Docker image for an application that runs on an ACK cluster?**

  Container Registry allows you to build a container image within a few clicks. For more information about how to build a Docker image for an application, 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 dockerize an application in a simplified manner.

- **If I do not want to build an image, how do I deploy an application to an ACK cluster?**

  ACK allows you to create applications by using images of the following types: images stored in Container Registry, official images, favorite images, and public images. For more information, see Deploy stateless applications from images.

- **How do I plan CIDR blocks before I create a cluster?**

  Before you create a cluster, make sure that the CIDR blocks of virtual private clouds (VPCs), services, and pods do not overlap. You can select to create a VPC automatically. In this case, use the default network address when you create a cluster. However, in some complex scenarios, you must plan CIDR blocks for Elastic Compute Service (ECS) instances, pods, and services. For more information, see Assign CIDR blocks to resources in a Kubernetes cluster under a VPC.

- **How do I select the Terway or Flannel plug-in when I create a cluster?**

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

- **How do I handle a cluster creation failure?**

  You can view the cluster events for troubleshooting. For more information, see Failed to create a Kubernetes cluster.

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

  You can use the following methods to access workloads over the Internet:

  - Use a NodePort
  - Use Server Load Balancer (SLB) instances
  - Nginx Ingress
  - Use an external DNS
  - Use a Destination Network Address Translation (DNAT) gateway

- **If multiple workloads exist in a cluster, how can a workload be accessed by other workloads in the cluster?**

  To access a workload from other workloads in the same cluster, use the internal DNS or ClusterIP service.

  Assume that Workload A and Workload B exist in a cluster. To allow Workload A to access Workload B, create a service of the ClusterIP type for Workload B. For more information, see Create a service. After you create a ClusterIP service, Workload A can use one of the following methods to access Workload B:

  - `<ClusterIP service name>.<Namespace to which Workload B belongs>.svc.cluster.local:<Port number>`
  - `ClusterIP:<Port number>`

- **What are the considerations when I access services through SLB instances?**

  If you create a service of the LoadBalancer type, Cloud Controller Manager (CCM) automatically creates and configures an SLB instance for the service. We recommend that you do not configure the SLB instance in the SLB console. This may cause the unavailability of the service. For more information, see Considerations for configuring a LoadBalancer service.

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

  We recommend that you use the aliyun-acr-credential-helper component. By default, each cluster has aliyun-acr-credential-helper installed. You can use this component to pull images without a password from Container Registry. For more information, see 使用免密组件拉取容器镜像.

3.Basic operations

3.1. Deploy stateless applications from images

This topic describes how to use an image to deploy an NGINX application that is accessible over the Internet. You can deploy stateless applications that are accessible over the Internet with images.

Prerequisites

Create a managed Kubernetes cluster

Procedure

1. Log on to the ACK console.
2. In the left-side navigation pane, click **Clusters**.
3. On the **Clusters** page, click the name of a cluster or click **Details** in the **Actions** column. The details page of the cluster appears.
4. In the left-side navigation pane, click **Workload**.
5. On the **Deployments** tab, 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 in which the application is deployed. The default namespace is automatically selected. You can select another namespace.
Replicas	The number of replicated pods that are provisioned for the application.
Type	The type of the application. You can select **Deployments**, **StatefulSets**, **Jobs**, **Cron Jobs**, and **DaemonSets**.
Labels	Add labels to the application.
Annotation	Add annotations to the application.
Synchronize Timezone	Specify whether to synchronize the timezone from nodes to 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 replicated pods that are provisioned for the application.

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

> **Note** At the top of the **Container** wizard page, click **Add Container** to add more containers for the application.

The following content describes how to set the parameters that are required to create a container.

- Basic Settings

Parameter	Description
Image Name	To use a Docker image or an image from Container Registry (ACR), click **Select Image**. In the dialog box that appears, select an image and click **OK**. In this example, the NGINX image is selected. To use a private image, enter the address of a private image registry. The registry address must be in the `domainname/namespace/imagename:tag` format.
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.
	Always Pull Images: If you do not select this check box, Container Service for Kubernetes (ACK) caches the pulled image. This improves the efficiency of pulling and deploying images. If the specified image version is the same as the cached image version, ACK deploys the application from the cached image. Therefore, when you update the application code, if you do not change the image version for reasons such as to support the upper-layer workloads, the previously cached image is used. If you select this check box, ACK pulls the image from the repository each time the application is deployed. This ensures that the latest image and code are used.
	Set Image Pull Secret: Click **Set Image Pull Secret** to set a secret for pulling images. You must set the secret if you want to pull the image from a private repository. For more information, see Use an image secret.
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 too many 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 when other services or processes compete for resources.

Parameter	Description
Container Start Parameter	■ stdin: specifies that start parameters defined in the console are imported to the Linux system. ■ tty: specifies that start parameters defined in a virtual terminal are imported to the console.
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 for managing pods. For more information, see init Containers.

- (Optional)Port Settings

  Specify the container ports.

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

- (Optional)Environment Variables

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

  - Type: the type of the environment variable. You can select **Custom**, **ConfigMaps**, **Secret**, **Value/ValueFrom**, and **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 Name drop-down list and select the target 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 for deploying the application contains the settings that reference all values in the specified Secret.

  - Variable Key: Specify the key of the environment variable.
  - Value/ValueFrom: Specify a resource as the reference to define 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 traffic. For more information about health checks, see Configure Liveness, Readiness and Startup Probes.

Request type	Description
HTTP request	Sends an HTTP GET request to the container. You can set the following parameters: ■ Protocol: HTTP or HTTPS. ■ Path: the requested path on the server. ■ Port: the container port that you want to open. Enter a port number from 1 to 65535. ■ HTTP Header: 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 field. This field specifies the time (in seconds) to wait 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 how often (in seconds) the probe is 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 the probe is considered failed after a failure. 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 the probe is considered failed after a success. Default value: 3. Minimum value: 1.

Request type	Description
TCP connection	Sends a TCP socket to the container. The kubelet attempts to open the socket on the specified port. If the connection can be established, the container is considered healthy. Otherwise, it is considered unhealthy. You can set the following parameters: ■ Port: 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) to wait 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 how often (in seconds) the probe is 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 the probe is considered failed after a failure. 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 the probe is considered failed after a success. Default value: 3. Minimum value: 1.
Command	Runs a probe command in the container to check the health status. You can set the following parameters: ■ Command: the probe command that is used 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) to wait 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 how often (in seconds) the probe is 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 the probe is considered failed after a failure. 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 the probe is considered failed 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 Configure the lifecycle of a container.

- **Start**: Specify the pre-start command and parameter.
- **Post Start**: Specify the post-start command.
- **Pre Stop**: Specify the pre-stop command.

- (Optional)Volume

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

  - **Local Storage**: You can select hostPath, ConfigMap, Secret, and EmptyDir. The source directory or file is mounted to a path in the container. For more information, see Volumes.
  - **PVC**: Select Cloud Storage.

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

- (Optional)Log configuration

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

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

Parameter	Description
Collection configuration	Logstore: Create a Logstore to store log data in Log Service.
	Log Path: Specify 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 tags	You can also add custom tags to the collected logs. After the tags are added, they are collected and printed along with logs. Custom tags provide an easy method to filter collected logs and perform statistical analytics.

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

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

Configure the access control settings to expose pods that run the application. In this example, a ClusterIP Service and an Ingress are created to enable Internet access to the NGINX application.

Parameter	Description
Service	Click **Create** on the right side of **Service**. In the Create Service dialog box, set the parameters. For more information about the parameters that are required to create a Service, see Create a service. Select **ClusterIP**.
Ingress	Click **Create** on the right side of **Ingress**. In the Create dialog box, set the parameters. For more information about the parameters that are required to create an Ingress, see Ingress configurations. **Note** When you deploy an application from an image, you can create an Ingress for only one Service. In this example, a virtual hostname is specified as the test domain. You need to add a mapping rule for this domain to the hosts file, as shown in the following code block: In actual scenarios, use a domain that has obtained an ICP number. `101.37.224.146 foo.bar.com # The IP address of the Ingress.`

You can find the newly created Service and Ingress in the **Access Control** section. Click **Update** or **Delete** to modify the settings.

- Scaling configurations

  Specify whether to enable **HPA** to automatically scale the number of pods based on the CPU and memory usage. This enables the application to run smoothly at different load levels.

  > **Note** To enable HPA, you must configure resource objects that can be scaled for the container. Otherwise, HPA cannot work.

  - **Metric**: Select CPU Usage or Memory Usage. The selected resource type must be the same as the one that you have specified in the Required Resources field.
  - **Condition**: Specify the resource usage threshold. HPA triggers scaling events when the threshold is exceeded.
  - **Max. Replicas**: Specify the maximum number of replicated pods to which the application can be scaled.
  - **Min. Replicas**: Specify the maximum number of replicated pods to which the application can be scaled.

- 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 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
Upgrade Method	Select Rolling Update or OnDelete. For more information, see Deployments.

Parameter	Description
Node Affinity	Add labels on worker nodes to the **node affinity**. Node Affinity supports required and preferred rules, and various operators such as In, NotIn, Exists, DoesNotExist, Gt, and Lt: ■ **Required**: Specify node labels that must be matched for pod scheduling. In the YAML file, these rules are specified by the requiredDuringSchedulingIgnoredDuringExecution field of nodeAffinity parameter. These rules have the same effect as the `NodeSelector` parameter. In this example, pods can be scheduled to only nodes with specific labels. You can create multiple required rules. However, only one of them must be met. ■ **Preferred**: Specify the node weight and node labels that may not 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 specified by the preferredDuringSchedulingIgnoredDuringExecution field of the nodeAffinity parameter. In this example, the scheduler attempts to schedule the pod to a node that matches the preferred rules. You can set node weights in preferred rules. If multiple nodes match the required and preferred rules, the node with the highest weight is preferred for pod scheduling. You can create multiple preferred rules. However, all of them must be met before the pods are scheduled.
	Pod affinity specifies that pods can be scheduled to nodes or topological domains where pods with matching labels are deployed. For example, you can use pod affinity to deploy services that communicate with each other to the same topology domain, such as a host. This reduces the network latency between these services. Pod affinity enables you to specify 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, DoesNotExist`.

Parameter	Description
Pod Affinity	■ **Required**: Specify rules that must be matched for pod scheduling. In the YAML file, these rules are specified 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 on pods and therefore must be scoped to a namespace. ■ **Topological Domain**: Specify the topologyKey. This specifies the key for the node label that the system uses to denote the topology domain. For example, if you set the parameter to `kubernetes.io/hostname`, nodes are used to determine topologies. If you set it to `beta.kubernetes.io/os`, the operating systems of nodes are used to determine topologies. ■ **Selector**: Click Add to add pod labels. ■ **View Applications**: Click **View Applications** and specify the namespace and application in the dialog box that appears. You can view the pod labels on the selected application and add themas 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 may not be matched for pod scheduling. In the YAML file, preferred rules are specified 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 node weights in preferred rules. Set the other parameters as described in the preceding settings. **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 schedules the pod to the node with the highest weight.

Parameter	Description
Pod Anti Affinity	Pod anti-affinity specifies that pods are not scheduled to topology domains where pods with matching labels are deployed. Pod anti-affinity is applicable in the following scenarios: ■ Schedule the pods of an application to different topology domains, such as multiple hosts. This allows you to enhance the stability of the application. ■ Grant a pod exclusive access to a node. This ensures resource isolation and guarantees that no other pod can share the specified node. ■ Schedule the pods of an application to different hosts if these pods may interfere each other. **Note** The parameters of pod anti-affinity rules are the same as those of pod affinity rules. We recommend that you create the rules based on different scenarios.
Toleration	Specify 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 labels to the pod.
  - Pod Annotations: Add annotations to the pod.

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

    

    After you submit the request, you are redirected to the nginx-deployment details page.

    Note You can also perform the following steps to create Ingresses and Services: In the **Access Control** section:
    - Click **Create** on the right side of **Service**. For more information, see Create a service.
    - Click **Create** on the right side of **Ingress**. For more information, see Ingress configurations.

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

    

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

3.2. Use an image to create a stateful application

You can use an image to create a stateful application. This topic describes how to use an image to create a stateful NGINX application in the Container Service for Kubernetes (ACK) console. This topic also describes the features of StatefulSets.

Prerequisites

Before you deploy the NGINX application, perform the following steps:

- Create a managed Kubernetes cluster
- Create a PVC
- Use kubectl to connect to an ACK cluster

Context

StatefulSets provide the following features:

Scenario	Description
Pod consistency	Pod consistency is used. This allows you to make sure that pods are launched and terminated in the defined order. This also allows you to make sure that the consistency of networks is high. Pod consistency is based on pod configurations, regardless of the node to which a pod is scheduled.
Stable and persistent storage	VolumeClaimTemplate allows you to mount persistent volumes (PVs) to pods. Mounted PVs are not deleted if pod replicas are deleted or scaled in.
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 for the hostname is `StatefulSet name-pod ordinal`.
Ordinal pod indexes	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, click **Clusters**.
3. On the **Clusters** page, click the name of a cluster or click **Details** in the **Actions** column. The details page of the cluster appears.
4. In the left-side navigation pane, click **Workload**.

5. Click the **StatefulSets** tab. On the **StatefulSets** tab, click **Create from Image**.
6. On the **Basic Information** wizard page, configure the basic settings.

   In this example, an application of the **StatefulSet** type is created.

Parameter	Description
Name	The name of the application.
Namespace	The namespace in which the application is deployed. The default namespace is automatically selected. You can select another namespace.
Replicas	The number of replicated pods that are provisioned for the application.
Type	The type of the application. You can select **Deployments**, **StatefulSets**, **Jobs**, **Cron Jobs**, or **DaemonSets**.
Labels	Add labels to the application.
Annotations	Add annotations to the application.
Synchronize Timezone	Specify whether to synchronize the timezone from nodes to containers.

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

   > **Note** At the top of the **Container** wizard page, click **Add Container** to add more containers for the application.

   The following parameters are required to configure the containers.

   - Basic settings

Parameter	Description

Parameter	Description
Image Name	■ To use a Docker image or an image from Container Registry (ACR), click **Select Image**. In the dialog box that appears, select an image and click **OK**. In this example, the NGINX image is selected. In the Select Image dialog box, click the **Search** tab, select **Docker Images** from the drop-down list, enter *NGINX* in the search bar, and then click Search. ■ Select an image from Alibaba Cloud Container Registry (ACR): In the Select Image dialog box, click the **Alibaba Cloud Container Registry** tab and select an image. You must first select a region and an ACR instance. For more information about ACR, see What is Container Registry Enterprise Edition?. **Note** On the **Alibaba Cloud Container Registry** tab, you can search for an image from ACR by name. ■ Select a Docker image: In the Select Image dialog box, click the **Docker Official Images** tab and select a Docker image. ■ Select a favorite image: In the Select Image dialog box, click the **Favorite Images** tab and select a favorite image. ■ Search for a Docker image or an image from ACR: In the Select Image dialog box, click the **Search** tab and search for a Docker image or an image in a specified region from ACR by name. ■ To use a private image, enter the address of a private image 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 you want to pull is found in the region where the cluster is deployed, the local image is used. Otherwise, ACK pulls the image from the corresponding repository. ■ **Always**: ACK pulls the image from the repository each time the application is deployed or expanded. ■ **Never**: ACK uses only local images. **Note** If you select **Image Pull Policy**, no image pull policy is applied for the deployment of the application. ■ To pull the image without a secret, click **Set Image Pull Secret** to set a secret for pulling images. For more information, see 使用免密组件拉取容器镜像.
Resource Limit	You can specify an upper limit for the CPU, memory, and ephemeral storage space that the container can consume. This allows you to make sure that the container does not occupy unnecessary 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 allows you to make sure that the application is still available when other services or processes occupy these resources.
Container Start Parameter	■ stdin: specifies that start parameters defined in the ACK console are imported to the Linux system. ■ tty: specifies that start parameters defined in a virtual terminal are imported to the ACK console.
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

  Specify the container ports.

- Name: Enter a name for a port.
- Port: 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 use key-value pairs to set environment variables for pods. Environment variables are used to apply pod configurations to containers. For more information, see Expose Pod Information to Containers Through Environment Variables.

  - Type: the type of the environment variable. You can select **Custom**, **ConfigMaps**, **Secret**, or **Value/ValueFrom**. 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 Name 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 required settings. These settings are used to reference all values in the specified Secret.
  - Variable Key: Specify the key of the environment variable.
  - Value/ValueFrom: This is the reference that is used to define the environment variable.

- (Optional)Health check

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

Request type	Description

Request type	Description
HTTP request	Sends an HTTP GET request to the container. You can set the following parameters: ■ Protocol: HTTP or HTTPS. ■ Path: the requested path on the server. ■ Port: the container port that you want to open. Enter a port number from 1 to 65535. ■ HTTP Header: 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: 15. ■ Period (s): the periodSeconds field in the YAML file. This field specifies the frequency (in seconds) that the probe is 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 successful probes that must occur for a container to be 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 failed probes that must occur for a container to be considered unhealthy after a successful probe. Default value: 3. Minimum value: 1.

Request type	Description
TCP connection	Opens a TCP socket to the container. The kubelet attempts to open the socket on the specified port. If the connection can be established, the container is considered healthy. Otherwise, it is considered unhealthy. You can set the following parameters: ■ Port: 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 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 frequency (in seconds) that the probe is 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 successful probes that must occur for a container to be 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 failed probes that must occur for a container to be considered unhealthy after a successful probe. Default value: 3. Minimum value: 1.

Request type	Description
Command	Runs a probe command in the container to check the health status. 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 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 frequency (in seconds) that the probe is 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 successful probes that must occur for a container to be 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 failed probes that must occur for a container to be considered unhealthy after a successful probe. 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 starts.

- (Optional)Volume

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

  - **Local Storage**: You can select hostPath, ConfigMap, Secret, and EmptyDir. The source directory or file is mounted to a path in the container. For more information, see Volumes.
  - **PVC**: Select Cloud Storage.

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

  

- (Optional)Log configuration

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

> **Notice** Make sure that the Log Service agent has been installed for the cluster.

Parameter	Description
Collection configuration	Logstore: Create a Logstore in Log Service to store log data.
	Log Path: Specify 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 tags	You can also set custom tags. Custom tags are added to the logs of the container when the logs are collected. Custom tags provide an easy method to filter collected logs and perform statistical analytics.

9. Set the preceding 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 service of the Cluster IP or Node Port type to enable internal communication.
> - External applications: For applications that are open to the Internet, you can configure the access control by using one of the following methods:
>   - Create a service of the LoadBalancer type and enable access to your application over the Internet by using a Server Load Balancer (SLB) instance.
>   - Create an Ingress to route external access to a service inside a cluster. For more information, see Ingress.

Configure the access control settings to enable access to pods that run the application. In this example, a Cluster IP service and an Ingress are created to enable access to the NGINX application over the Internet.

Parameter	Description
Service	Click **Create** on the right side of **Service**. In the **Create Service** dialog box, set the parameters. For more information about the parameters that are required to create a service, see Create a service. Select **Cluster IP**.

Parameter	Description
Ingress	Click **Create** on the right side of **Ingress**. In the Create dialog box, set the parameters. For more information about the parameters that are required to create an Ingress, see Ingress configurations. **Note** When you deploy an application from an image, you can create an Ingress for only one service. In this example, a virtual host name 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 newly created service and Ingress in the **Access Control** section. Click **Update** or **Delete** to modify the settings.

- Scaling settings

  Specify whether to enable **HPA** to automatically scale the number of pods based on the CPU and memory usage. This enables the application to run as expected at different load levels.

  Note To enable Horizontal Pod Autoscaler (HPA), you must configure resources that support scaling for the container. Otherwise, HPA does not take effect.

  - **Metric**: Select CPU Usage or Memory Usage. The selected resource type must be the same as the one you have specified in the Required Resources field.
  - **Condition**: Specify the resource usage threshold. HPA triggers scaling events when the threshold is exceeded.
  - **Max. Replicas**: Specify the maximum number of replicated pods to which the application can be scaled.
  - **Min. Replicas**: Specify the minimum number of replicated pods that must run.

- Scheduling settings

  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 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.
Node Affinity	Add labels to worker nodes to set the node affinity. Node Affinity supports required and preferred rules, and various operators, such as In, NotIn, Exists, DoesNotExist, Gt, and Lt. ■ **Required**: Set node labels 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 to only nodes with the specified labels. You can create multiple required rules. However, only one of the rules must be met. ■ **Preferred**: Specify the node weight and node labels 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 the pod to a node that matches the preferred rules. You can set node weights in preferred rules. If multiple nodes match the required and preferred rules, the node with the highest weight is preferred for pod scheduling. You can create multiple preferred rules. However, all of the rules must be met before the pods are scheduled.
	Pod affinity specifies that pods can be scheduled to nodes or topological domains where pods with matching labels are deployed. For example, you can use pod affinity to deploy services that communicate with each other to the same topology domain, such as a host. This reduces the network latency between these services. Pod affinity enables you to specify which node pods can be scheduled based on the labels on other running pods. Pod affinity supports required and preferred rules, and various operators, such as **In, NotIn, Exists, and DoesNotExist**.

Parameter	Description
Pod Affinity	■ **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. In this case, labels are limited by namespace constraints. ■ **Topological Domain**: Set the topologyKey. This specifies the key for the node label that the system uses to denote the topology domain. For example, if you set the parameter to `kubernetes.io/hostname`, nodes are used to define topologies. If you set the parameter to `beta.kubernetes.io/os`, the operating systems of nodes are used to determine topologies. ■ **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 node weights in preferred rules. Set the other parameters as described in the preceding settings. **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 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 topology 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 topology 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 specified node. ■ To ensure that these pods do not interfere with each other, we recommend that you schedule the pods of an application to different hosts. **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	Set 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 labels to the pod.
  - Pod Annotations: Add annotations to the pod.

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

    The details page of the created StatefulSet appears.
13. In the upper-left corner of the page, click **Back** to return to the StatefulSets tab. On the StatefulSets tab, you can find the created application.
14. (Optional)Click **Scale** in the Actions column to scale the application.
    i. In the dialog box that appears, set Desired Number of Pods to 3 and click OK. After you scale out the application, you can find that all pods 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, click Persistent Volumes. On the **Persistent Volume Claims** > **Persistent Volumes** tab, you can find that a PV and a PVC are created for the newly added pod. However, if the application is scaled in, existing PVs and PVCs are not deleted.

Related operations

In the left-side navigation pane of the ACK console, click **Clusters**. On the Clusters page, click the name of the cluster or click **Applications** in the **Actions** column. In the left-side navigation pane, click **Workload**. Click the **StatefulSets** tab. On the **StatefulSets** tab, click the name of the application or click **Details** in the **Actions** column. On the details page, you can edit, scale, redploy, 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 Actions column of the application to modify the configurations of the application.
- Scale: On the details page of the application, click **Scale** in the Actions column of the application to scale the application to a required number of pods.
- View the YAML file: On the details page of the application, click **View in YAML** in the Actions column of the application to update, and download the YAML file of the application. You can also save the YAML file as a template.
- Redeploy: On the details page of the application, click **Redeploy** in the Actions column of the application to redeploy the application.
- Refresh: On the details page of the application, click **Refresh** to refresh the application information.

What's next

Log on to a master node and perform the following steps to test the persistent storage.

1. Create a test file in the cloud 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. Delete pod nginx-1:

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

3. After the system recreates and starts pod nginx-1, query the files in the /tmp directory. The following result shows that the statefulset file still exists. This ensures the high availability of the application that runs on the StatefulSet.

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

3.3. Deploy WordPress with MySQL

This topic describes how to deploy a WordPress site with a MySQL database on Kubernetes.

Prerequisites

- A cluster of Container Service for Kubernetes (ACK) is created. For more information, see 快速创建

Kubernetes托管版集群.

- Persistent volumes (PVs) and persistent volume claims (PVCs) are created. For more information about how to create a PV, see Use Alibaba Cloud disks as volumes, Use NAS volumes, and Use an OSS volume. For more information about how to create a PVC, see Create a PVC. In this topic, an Alibaba Cloud disk is provisioned as a PV and mounted by using a PVC. A PVC named WordPress-pv-claim and a PVC named WordPress-MySQL-pv-claim are created. WordPress-pv-claim is used to mount a PV to the WordPress application. WordPress-MySQL-pv-claim is used to mount a PV to the WordPress-MySQL application.

Context

This example shows how to use custom orchestration templates to set up a WordPress site with a MySQL database.

The following tools are used:

- WordPress
- MySQL

The following Kubernetes resources are used:

- PV
- Kubernetes Secrete
- Kubernetes Service

Procedure

1. Log on to the ACK console.
2. In the left-side navigation pane, click **Clusters**.
3. On the **Clusters** page, click the name of a cluster or click **Details** in the **Actions** column. The details page of the cluster appears.
4. In the left-side navigation pane, click **Configurations**.
5. On the **Configurations** page, click the **Secrets** tab. In the upper-right corner of the **Secrets** tab, click **Create**.

   For more information, see Create a Secret.

   The created PVCs are used when you configure the YAML files of the applications. You can specify WordPress-pvc in the YAML file of the WordPress application to mount the corresponding PV. You can also specify WordPress-MySQL in the YAML file of the WordPress-MySQL application to mount the corresponding PV.

   A Secret is used to manage the usernames and passwords that are required to create and access the MySQL database.

   In this example, a Secret named mysql-pass is created to manage the MySQL root password. The Secret type is set to **Opaque**. You can specify the created Secret in the YAML files of the applications.
6. In the left-side navigation pane, click **Workload**.
7. Click the **Deployments** tab. In the upper-right corner of the **Deployments** tab, click **Create from**

Template.

Select the namespace to deploy the application. Use the following YAML template to create a Deployment on which the WordPress application runs.

```
apiVersion: apps/v1
kind: Deployment
metadata:
 name: wordpress
 labels:
  app: wordpress
spec:
 selector:
  matchLabels:
   app: wordpress
   tier: frontend
 strategy:
  type: Recreate
 template:
  metadata:
   labels:
    app: wordpress
    tier: frontend
  spec:
   containers:
   - image: wordpress:4
    name: wordpress
    env:
    - name: WORDPRESS_DB_HOST
     value: wordpress-mysql  # Specify the MySQL database that is accessed by the WordPress applicati
on. You must specify the value as the name of the Service that is created for the WordPress-MySQL appl
ication.
    - name: WORDPRESS_DB_PASSWORD
     valueFrom:
      secretKeyRef:
       name: mysql-pass
       key: password-mysql
    ports:
    - containerPort: 80
     name: wordpress
    volumeMounts:
```

```
    - name: wordpress-pvc
      mountPath: /var/www/html
  volumes:
  - name: wordpress-pvc
    persistentVolumeClaim:
      claimName: wordpress-pv-claim
```

Use the following YAML template to create a Deployment on which the MySQL application runs.

```
apiVersion: apps/v1
kind: Deployment
metadata:
 name: wordpress-mysql
 labels:
  app: wordpress
spec:
 selector:
  matchLabels:
   app: wordpress
   tier: mysql
 strategy:
  type: Recreate
 template:
  metadata:
   labels:
    app: wordpress
    tier: mysql
  spec:
   containers:
   - image: mysql:5.6
    name: mysql
    env:
    - name: MYSQL_ROOT_PASSWORD
     valueFrom:
      secretKeyRef:
       name: mysql-pass
       key: password-mysql
    ports:
    - containerPort: 3306
     name: mysql
    volumeMounts:
    - name: wordpress-mysql-pvc
     mountPath: /var/lib/mysql
   volumes:
   - name: wordpress-mysql-pvc
    persistentVolumeClaim:
     claimName: wordpress-mysql-pv-claim
```

8. Create Services.

To enable external access to the WordPress application, create a **LoadBalancer** type Service for the WordPress application. ACK will automatically create a Server Load Balancer (SLB) instance for the created LoadBalancer type Service. This allows external access to the WordPress application.

To enable WordPress to access the MySQL database internally, create a ClusterIP type Service for the WordPress-MySQL application. The MySQL database does not require external access. Therefore, you do not need to create a **LoadBalancer** type Service for the WordPress-MySQL application. For more information about how to create a Service, see Create a service.

Use the following YAML templates to create a Service for the WordPress application and a Service for the WordPress-MySQL application.

```
apiVersion: v1
kind: Service
metadata:
 name: wordpress
 labels:
  app: wordpress
spec:
 ports:
  - port: 80
 selector:
  app: wordpress
  tier: frontend
 type: LoadBalancer
---
apiVersion: v1
kind: Service
metadata:
 name: wordpress-mysql
 labels:
  app: wordpress
spec:
 ports:
  - port: 3306
 selector:
  app: wordpress
  tier: mysql
 clusterIP: None
```

9. After the Services are created, go to the details page of the cluster. In the left-side navigation pane, click **Services**.

On the Services page, you can find the Service that is created for the WordPress application and obtain the external endpoint of the Service.

10. Enter *<external endpoint IP address>/wp-admin/install.php* into the address bar of your browser and press Enter to access WordPress.

What's next

When you configure WordPress, you can use the created Secret as the credential for logging on to WordPress. Data generated in the container that runs WordPress is stored in the mounted PV.

3.4. Deploy WordPress applications in ACK clusters

This topic describes how to deploy WordPress applications in clusters of Alibaba Cloud Container Service for Kubernetes (ACK).

Prerequisites

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

> **Note** You can deploy WordPress applications in only ACK cluster. 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. On the App Hub tab, enter wordpress into the Name bar and click the search icon on the right side of the page. Find and click wordpress 5.3.0.

4. On the **App Catalog - wordpress** page, click the **Parameter** tab. Set the persistence parameter to false as shown in the following figures:
   - Set the persistence parameter in line 197 to false.

   - Set the persistence parameter in line 323 to false.

5. In the Deploy section, set the **Cluster**, **Namespace**, and **Release Name** parameters, and click **Create**.
6. In the left-side navigation pane, click **Clusters**.
7. On the **Clusters** page, click the name of a cluster or click **Details** in the **Actions** column. The details page of the cluster appears.
8. In the left-side navigation pane, click **Services**.

9. On the **Services** page, find the newly deployed application, and click the hyperlink in the External Endpoint column. The WordPress homepage appears.

> **Note** If you want to modify the parameters after the WordPress application is deployed, you can delete and redeploy it. In the left-side navigation pane, click **Releases**. On the Release page, click the **Helm** tab. On the Helm tab, find and delete the Helm chart of the target WordPress application. Then, follow the preceding steps to redeploy the WordPress application.

4.Advanced operations

4.1. Create an application from a private image repository

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 in the Container Service for Kubernetes (ACK) console.

Create a private image repository

If this is the first time you use the Container Registry console, you will be **prompted** to activate Container Registry. Click **Activate Now**, and set a password for logging on to Container Registry.

1. Log on to the ACK console.
2. In the left-side navigation pane, choose **Marketplace > Alibaba Cloud Container Registry**.
3. In the left-side navigation pane of the Container Registry console, choose **Default Instance > Repositories**. At the top of the page, select a region to deploy the repository, and then click **Create Repository**.
4. In the **Create Repository** dialog box, set the **Namespaces**, **Repository Name**, **Summary**, and **Repository Type** parameters. In this example, set Repository Type to Private. Click **Next**.

   

5. Select **Local Repository** as the source and click **Create Repository**.
6. Go to the Repositories page, select the region and namespace where the repository is created, find the created repository, and click **Manage** in the Actions column.
7. On the **Details** page, click the **Guide** tab. You can view the instructions about how to use the repository.

   

8. Log on to the repository through a Linux server and run the following command to upload a local image to this repository:

```
sudo docker login --username=abc@aliyun.com registry.cn-hangzhou.aliyuncs.com
Password:  ## The password that is used to log on to the repository.
Login Succeeded
```

```
docker images
# Tomcat is used as an example.
    REPOSITORY    TAG       IMAGE ID       CREATED       SIZE
    tomcat        latest    2d43521f2b1a   6 days ago    463MB
```

```
sudo docker tag [ImageId] registry.cn-hangzhou.aliyuncs.com/[namespace]/tomcat-private:[image tag]

sudo docker push registry.cn-hangzhou.aliyuncs.com/[namespace]/tomcat-private:[image tag]
```

After a private image is created, the system prints the following 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
```

9. Go to the details page of the image repository. In the left-side navigation pane, click **Tags**. On the Tags page, you can find that the image is uploaded. The tag of the image indicates the image version.

Create a private repository Secret

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

1. Log on to the ACK console.
2. In the left-side navigation pane, click **Clusters**.
3. On the **Clusters** page, click the name of a cluster or click **Details** in the **Actions** column. The details page of the cluster appears.
4. In the left-side navigation pane, click **Configurations**.
5. On the **Configurations** page, click the **Secret** tab.
6. In the upper-right corner of the **Secrets** tab, click **Create**.
7. On the **Create** page, 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 general Secret. Enter a key and a value. The value must be encoded by using Base64. ○ Private Repository Logon Secret: the credentials that are required to pull images from the private repository. Enter the address, username, and password of the repository. Note The username is the full name of your Alibaba Cloud account. The password is the one that is specified for logging on to Container Registry. You can go to the **Access Credential** page to change the password. ○ TLS Certificate: A Transport Layer Security (TLS) certificate is used to verify user identities. ■ Cert: Enter the content of the TLS certificate. ■ Key: Enter the private key of the TLS certificate.

Note

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 Secret** from the command-line interface (CLI). For more information, see Use kubectl to connect to an ACK cluster.

Create an application from the private image repository

1. Log on to the ACK console.
2. In the left-side navigation pane, click **Clusters**.
3. On the **Clusters** page, click the name of a cluster or click **Details** in the **Actions** column. The details page of the cluster appears.
4. In the left-side navigation pane, click **Workload**.
5. On the **Workload** page, click the **Deployments** tab.
6. In the upper-right corner of the **Deployments** tab, click **Create from Template**.

   Note You can also click **Create from Image** to create an application. For more information, see Use an image secret.

7. Set **Sample Template** to **Custom** and copy the following content to the **template**.

```
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 /[namespace]/tomcat-private:latest
    ports:
    - containerPort: 8080
   imagePullSecrets:
   - name: regsecret
```

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

   

For more information, see Use a private repository.