## Contents

1 What Is Cloud Container Engine? ................................................................. 1

2 High-Risk Operations and Solutions ............................................................ 3

3 Clusters ........................................................................................................... 8

3.1 Cluster Overview ......................................................................................... 8

3.2 Cluster Lifecycle ......................................................................................... 12

3.3 Buying a CCE Cluster ............................................................................... 13

3.4 Buying a Kunpeng Cluster ......................................................................... 32

3.5 Using kubectl to Run a Cluster ................................................................. 48

3.5.1 Connecting to a Cluster Using kubectl ............................................... 48

3.5.2 Configuring High Availability of kube-dns/CoreDNS Using kubectl .... 51

3.5.3 Common kubectl Commands ............................................................... 53

3.5.4 kubectl Usage Guide ........................................................................... 59

3.6 Setting Cluster Auto Scaling ...................................................................... 61

3.7 Upgrading a Cluster ................................................................................... 63

3.7.1 Upgrade Overview ................................................................................ 64

3.7.2 Before You Start .................................................................................. 69

3.7.3 Upgrading a Cluster Using the Console .............................................. 72

3.7.4 Upgrading a Cluster (from v1.15 to v1.19) ......................................... 77

3.7.5 Migrating Services Across Clusters of Different Versions ................. 81

3.7.6 CCE Kubernetes Release Notes ............................................................ 83

3.8 Managing a Cluster ................................................................................... 83

3.8.1 Deleting a Cluster (Pay-per-Use) ......................................................... 83

3.8.2 Deleting, Unsubscribing From, or Releasing a Yearly/Monthly-Billed Cluster ................................................................. 84

3.8.3 Renewing a Yearly/Monthly-Billed Cluster ........................................ 88

3.8.4 Hibernating and Waking Up a Cluster (Pay-per-Use) ......................... 89

3.8.5 Configuring Kubernetes Parameters .................................................. 91

3.9 Obtaining a Cluster Certificate ............................................................... 94

3.10 Monitoring a Cluster ............................................................................... 97

3.11 Controlling Cluster Permissions ............................................................. 97

4 Nodes ............................................................................................................. 101

4.1 Node Overview ......................................................................................... 101

4.2 Buying a Node ......................................................................................... 103
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.3</td>
<td>Accepting ECSs/BMSs as Nodes into a Cluster</td>
<td>114</td>
</tr>
<tr>
<td>4.4</td>
<td>Removing a Node</td>
<td>116</td>
</tr>
<tr>
<td>4.5</td>
<td>Logging In to a Node</td>
<td>118</td>
</tr>
<tr>
<td>4.6</td>
<td>Monitoring a Node</td>
<td>119</td>
</tr>
<tr>
<td>4.7</td>
<td>Managing Node Labels</td>
<td>121</td>
</tr>
<tr>
<td>4.8</td>
<td>Synchronizing Node Data</td>
<td>124</td>
</tr>
<tr>
<td>4.9</td>
<td>Resetting a Node</td>
<td>125</td>
</tr>
<tr>
<td>4.10</td>
<td>Deleting a Node</td>
<td>127</td>
</tr>
<tr>
<td>4.11</td>
<td>Stopping a Node</td>
<td>129</td>
</tr>
<tr>
<td>4.12</td>
<td>Performing Rolling Upgrade for Nodes</td>
<td>130</td>
</tr>
<tr>
<td>4.13</td>
<td>Upgrading the Node OS Kernel</td>
<td>135</td>
</tr>
<tr>
<td>4.14</td>
<td>Formula for Calculating the Reserved Resources of a Node</td>
<td>139</td>
</tr>
<tr>
<td>4.15</td>
<td>Creating a Linux LVM Disk Partition for Docker</td>
<td>141</td>
</tr>
<tr>
<td>4.16</td>
<td>Docker Disk Space</td>
<td>142</td>
</tr>
<tr>
<td>4.17</td>
<td>Adding a Second Data Disk to a Node in a CCE Cluster</td>
<td>145</td>
</tr>
<tr>
<td>5</td>
<td>Node Pools</td>
<td>147</td>
</tr>
<tr>
<td>5.1</td>
<td>Node Pool Overview</td>
<td>147</td>
</tr>
<tr>
<td>5.2</td>
<td>Creating a Node Pool</td>
<td>151</td>
</tr>
<tr>
<td>5.3</td>
<td>Managing a Node Pool</td>
<td>160</td>
</tr>
<tr>
<td>6</td>
<td>Workloads</td>
<td>167</td>
</tr>
<tr>
<td>6.1</td>
<td>Overview</td>
<td>167</td>
</tr>
<tr>
<td>6.2</td>
<td>Creating a Deployment</td>
<td>169</td>
</tr>
<tr>
<td>6.3</td>
<td>Creating a StatefulSet</td>
<td>178</td>
</tr>
<tr>
<td>6.4</td>
<td>Creating a DaemonSet</td>
<td>187</td>
</tr>
<tr>
<td>6.5</td>
<td>Creating a Job</td>
<td>193</td>
</tr>
<tr>
<td>6.6</td>
<td>Creating a Cron Job</td>
<td>200</td>
</tr>
<tr>
<td>6.7</td>
<td>Managing Pods</td>
<td>206</td>
</tr>
<tr>
<td>6.8</td>
<td>Managing Workloads and Jobs</td>
<td>208</td>
</tr>
<tr>
<td>6.9</td>
<td>Scaling a Workload</td>
<td>216</td>
</tr>
<tr>
<td>6.10</td>
<td>Configuring a Container</td>
<td>222</td>
</tr>
<tr>
<td>6.10.1</td>
<td>Using a Third-Party Image</td>
<td>222</td>
</tr>
<tr>
<td>6.10.2</td>
<td>Setting Container Specifications</td>
<td>224</td>
</tr>
<tr>
<td>6.10.3</td>
<td>Setting Container Lifecycle Parameters</td>
<td>226</td>
</tr>
<tr>
<td>6.10.4</td>
<td>Setting Container Startup Commands</td>
<td>231</td>
</tr>
<tr>
<td>6.10.5</td>
<td>Setting Health Check for a Container</td>
<td>236</td>
</tr>
<tr>
<td>6.10.6</td>
<td>Setting an Environment Variable</td>
<td>239</td>
</tr>
<tr>
<td>6.10.7</td>
<td>Collecting Default Standard Output Container Logs</td>
<td>241</td>
</tr>
<tr>
<td>6.10.8</td>
<td>Collecting Container Logs from Mount Paths</td>
<td>243</td>
</tr>
<tr>
<td>6.10.9</td>
<td>Working with Prometheus for Custom Monitoring</td>
<td>249</td>
</tr>
<tr>
<td>6.10.10</td>
<td>Configuring APM Settings for Custom Monitoring</td>
<td>251</td>
</tr>
<tr>
<td>6.10.11</td>
<td>Enabling ICMP Security Group Rules</td>
<td>255</td>
</tr>
<tr>
<td>6.10.12</td>
<td>Using Kubernetes In-Cluster DNS</td>
<td>256</td>
</tr>
</tbody>
</table>
10 Storage (FlexVolume) .................................................................................................................. 450
  10.1 FlexVolume Overview .............................................................................................................. 450
  10.2 How Do I Change the Storage Class Used by a Cluster of v1.15 from FlexVolume to CSI Everest?.. 453
  10.3 Using EVS Disks as Storage Volumes ..................................................................................... 465
  10.3.1 Overview ............................................................................................................................. 465
  10.3.2 Using EVS Volumes .............................................................................................................. 466
  10.3.3 (kubectl) Automatically Creating an EVS Disk ................................................................. 472
  10.3.4 (kubectl) Creating a PV from an Existing EVS Disk ............................................................ 473
  10.3.5 (kubectl) Creating a Pod Mounted with an EVS Volume.................................................... 483
  10.4 Using SFS Turbo File Systems as Storage Volumes ............................................................. 486
  10.4.1 Overview ............................................................................................................................. 486
  10.4.2 Using SFS Turbo Volumes .................................................................................................. 487
  10.4.3 (kubectl) Creating a PV from an Existing SFS Turbo File System ........................................ 490
  10.4.4 (kubectl) Creating a Deployment Mounted with an SFS Turbo Volume ............................ 492
  10.4.5 (kubectl) Creating a StatefulSet Mounted with an SFS Turbo Volume............................. 494
  10.5 Using OBS Buckets as Storage Volumes ................................................................................. 496
  10.5.1 Overview ............................................................................................................................. 496
  10.5.2 Using OBS Volumes ............................................................................................................ 497
11 Namespaces ................................................................................................................................. 528
11.1 Creating a Namespace .............................................................................................................. 528
11.2 Managing Namespaces ............................................................................................................ 530
11.3 Configuring a Namespace-level Network Policy .................................................................... 532
11.4 Setting a Resource Quota ....................................................................................................... 533

12 Configuration Center .................................................................................................................. 536
12.1 Creating a ConfigMap .............................................................................................................. 536
12.2 Using a ConfigMap ................................................................................................................... 539
12.3 Creating a Secret ...................................................................................................................... 541
12.4 Using a Secret ......................................................................................................................... 544

13 Charts (Helm) ............................................................................................................................ 547
13.1 My Charts ............................................................................................................................... 547
13.1.1 Overview .......................................................................................................................... 547
13.1.2 Preparing a Chart ............................................................................................................ 547
13.1.3 Uploading a Chart .......................................................................................................... 550
13.1.4 Creating a Workload from a Chart ................................................................................ 551
13.1.5 Using EVS Disks ............................................................................................................. 552
13.1.6 Using HUAWEI CLOUD LoadBalancer Services ............................................................... 554

14 Add-ons .................................................................................................................................. 555
14.1 Overview ............................................................................................................................... 555
14.2 coredns (System Resource Add-on, Mandatory) ................................................................... 557
14.3 storage-driver (System Resource Add-on, Mandatory) .......................................................... 562
14.4 everest (System Resource Add-on, Mandatory) ................................................................... 563
14.5 npd ....................................................................................................................................... 565
14.6 dashboard .............................................................................................................................. 566
14.7 autoscaler .............................................................................................................................. 572
14.8 nginx-ingress ........................................................................................................................ 580
14.9 metrics-server ....................................................................................................................... 585
14.10 cce-hpa-controller .............................................................................................................. 586
14.11 prometheus .......................................................................................................................... 587
## Contents

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>14.12</td>
<td>web-terminal</td>
<td>591</td>
</tr>
<tr>
<td>14.13</td>
<td>gpu-beta</td>
<td>594</td>
</tr>
<tr>
<td>14.14</td>
<td>huawei-npu</td>
<td>598</td>
</tr>
<tr>
<td>14.15</td>
<td>volcano</td>
<td>599</td>
</tr>
<tr>
<td>14.16</td>
<td>spark-operator</td>
<td>601</td>
</tr>
<tr>
<td>14.17</td>
<td>tf-operator</td>
<td>602</td>
</tr>
<tr>
<td>14.18</td>
<td>rc-recycler</td>
<td>603</td>
</tr>
<tr>
<td>15</td>
<td>Auto Scaling</td>
<td>606</td>
</tr>
<tr>
<td>15.1</td>
<td>Overview</td>
<td>606</td>
</tr>
<tr>
<td>15.2</td>
<td>Scaling a Workload</td>
<td>608</td>
</tr>
<tr>
<td>15.2.1</td>
<td>Workload Scaling Mechanisms</td>
<td>608</td>
</tr>
<tr>
<td>15.2.2</td>
<td>Creating an HPA Policy for Workload Auto Scaling</td>
<td>611</td>
</tr>
<tr>
<td>15.2.3</td>
<td>Creating a CustomedHPA Policy for Workload Auto Scaling</td>
<td>613</td>
</tr>
<tr>
<td>15.2.4</td>
<td>Managing Workload Scaling Policies</td>
<td>618</td>
</tr>
<tr>
<td>15.3</td>
<td>Scaling a Cluster/Node</td>
<td>620</td>
</tr>
<tr>
<td>15.3.1</td>
<td>Node Scaling Mechanisms</td>
<td>621</td>
</tr>
<tr>
<td>15.3.2</td>
<td>Creating a Node Scaling Policy</td>
<td>623</td>
</tr>
<tr>
<td>15.3.3</td>
<td>Managing Node Scaling Policies</td>
<td>629</td>
</tr>
<tr>
<td>15.4</td>
<td>Using HPA and CA for Auto Scaling of Workloads and Nodes</td>
<td>630</td>
</tr>
<tr>
<td>16</td>
<td>Permissions Management</td>
<td>639</td>
</tr>
<tr>
<td>16.1</td>
<td>Permissions Overview</td>
<td>639</td>
</tr>
<tr>
<td>16.2</td>
<td>Cluster Permissions (IAM-based)</td>
<td>641</td>
</tr>
<tr>
<td>16.3</td>
<td>Namespace Permissions (Kubernetes RBAC-based)</td>
<td>646</td>
</tr>
<tr>
<td>16.4</td>
<td>Example: Designing and Configuring Permissions for Users in a Department</td>
<td>654</td>
</tr>
<tr>
<td>16.5</td>
<td>Permission Dependency of the CCE Console</td>
<td>660</td>
</tr>
<tr>
<td>16.6</td>
<td>Pod Security Policies</td>
<td>668</td>
</tr>
<tr>
<td>17</td>
<td>System Steward</td>
<td>672</td>
</tr>
<tr>
<td>17.1</td>
<td>System Check</td>
<td>672</td>
</tr>
<tr>
<td>17.2</td>
<td>System Hardening</td>
<td>675</td>
</tr>
<tr>
<td>18</td>
<td>Cloud Eye Service (CES)</td>
<td>679</td>
</tr>
<tr>
<td>18.1</td>
<td>Monitoring Metrics</td>
<td>679</td>
</tr>
<tr>
<td>18.2</td>
<td>Setting Alarm Rules</td>
<td>680</td>
</tr>
<tr>
<td>18.3</td>
<td>Viewing Metrics</td>
<td>683</td>
</tr>
<tr>
<td>19</td>
<td>Cloud Trace Service (CTS)</td>
<td>684</td>
</tr>
<tr>
<td>19.1</td>
<td>CCE Operations Supported by CTS</td>
<td>684</td>
</tr>
<tr>
<td>19.2</td>
<td>Querying CTS Logs</td>
<td>689</td>
</tr>
<tr>
<td>20</td>
<td>Related Services</td>
<td>691</td>
</tr>
<tr>
<td>20.1</td>
<td>Software Repository for Container (SWR)</td>
<td>691</td>
</tr>
<tr>
<td>20.2</td>
<td>Application Operations Management (AOM)</td>
<td>692</td>
</tr>
</tbody>
</table>
What Is Cloud Container Engine?

Kubernetes is an open source container orchestration engine. Adapting to native Kubernetes capabilities, HUAWEI CLOUD launches Cloud Container Engine (CCE) for you to manage your containerized applications.

CCE allows you to create highly scalable, high-performance, and enterprise-class Kubernetes clusters and run Docker containers. With CCE, you can easily deploy, manage, and scale containerized applications on HUAWEI CLOUD.

CCE is deeply integrated with high-performance HUAWEI CLOUD computing (ECS/BMS), network (VPC/EIP/ELB), and storage (EVS/OBS/SFS) services, and supports heterogeneous computing architectures such as GPU, Arm, and FPGA. You can build high-availability Kubernetes clusters secured by multi-availability zone (AZ) and cross-region disaster recovery (DR), as well as enjoy auto scaling that simplifies cluster construction and greatly shortens the response time.

Defined Terms

This section helps you obtain a deeper understanding of how CCE works.

- **Cluster**: A cluster is a collection of cloud resources required for running containers, such as cloud servers and load balancers.
- **Pod**: A pod consists of one or more related containers that share the same storage and network space.
- **Workload**: A workload is a Kubernetes resource object, which is used to manage the creation and scheduling of pod replicas and automatically control the entire lifecycle of pod replicas.
- **Service**: A Service is an abstraction which defines a logical set of pods and a policy by which to access them (sometimes this pattern is called a microservice).
- **Ingress**: An ingress is a set of rules used to route external HTTP(S) traffic to Services.
- **Helm**: Helm is a package manager for Kubernetes. You can use Helm charts to define, install, and upgrade even the most complex Kubernetes application.
- **Image repository**: An image repository stores Docker images that can be used to deploy containerized services.

For more information, visit [https://kubernetes.io/docs/concepts/](https://kubernetes.io/docs/concepts/).
Main Functions

CCE supports full lifecycle management of containerized applications.

Cluster management
- You can create Kubernetes clusters with just a few clicks on the HUAWEI CLOUD CCE console. Cross-AZ high-availability is supported.

One-stop container management
- Containerized application lifecycle management
- High-performance container networking: tunnel network, Cloud Native Network 2.0, and VPC network
- Persistent storage using cloud services, such as Elastic Volume Service (EVS), Scalable File Service (SFS), and Object Storage Service (OBS)
- Multi-dimensional monitoring of resources, applications, and containers
- Diversified logs and statistics
- Role-based access control (RBAC) and container runtime security

Third-party components
- Rich Helm charts
- Custom images and shared images supported by interconnection with open source image registries and HUAWEI CLOUD Software Repository for Container (SWR)

Developer services
- Open APIs and native APIs from the community
- kubectl-related add-on and native kubectl from the community

Billing

CCE is free. You only pay for the resources created when you are using CCE.

For details about the billing modes and prices, see Pricing Details.
2 High-Risk Operations and Solutions

During service deployment or running, you may trigger high-risk operations at different levels, causing service faults or interruption. To help you better estimate and avoid operation risks, this section introduces the consequences and solutions of high-risk operations from multiple dimensions, such as clusters, nodes, networking, load balancing, logs, and EVS disks.

## Clusters and Nodes

<table>
<thead>
<tr>
<th>Category</th>
<th>Operation</th>
<th>Impact</th>
<th>Solution</th>
</tr>
</thead>
</table>
| Master nodes      | Modifying the security group of a node in a cluster                       | The master node may be unavailable.         | ![Restore the security group by referring to Buying a CCE Cluster and allow traffic from the security group to pass through.](image)
|                   | Letting the node expire or destroying the node                            | The master node will be unavailable.        | This operation cannot be undone.                                                                                                                                                           |
|                   | Reinstalling the OS                                                       | Components on the master node will be deleted. | This operation cannot be undone.                                                                                                                                                           |
|                   | Upgrading components on the master or etcd node                           | The cluster may be unavailable.             | Roll back to the original version.                                                                                                                                                         |

*NOTE* Naming rule of a master node: Cluster name-cce-control-Random number
<table>
<thead>
<tr>
<th>Category</th>
<th>Operation</th>
<th>Impact</th>
<th>Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deleting or formatting core directory data such as <code>/etc/kubernetes</code> on the node</td>
<td>The master node will become unavailable.</td>
<td>This operation cannot be undone.</td>
<td></td>
</tr>
<tr>
<td>Changing the node IP address</td>
<td>The master node will become unavailable.</td>
<td>Change the IP address back to the original one.</td>
<td></td>
</tr>
<tr>
<td>Modifying parameters of core components (such as etcd, kube-apiserver, and docker)</td>
<td>The master node may be unavailable.</td>
<td>Restore the parameter settings to the recommended values. For details, see Configuring Kubernetes Parameters.</td>
<td></td>
</tr>
<tr>
<td>Replacing the master or etcd certificate</td>
<td>The cluster may become unavailable.</td>
<td>This operation cannot be undone.</td>
<td></td>
</tr>
<tr>
<td>Worker nodes</td>
<td>Modifying the security group of a node in a cluster</td>
<td>The node may be unavailable. <strong>NOTE</strong> Naming rule of a worker node: <code>Cluster name-cce-node-Random number</code></td>
<td>Restore the security group by referring to Buying a CCE Cluster and allow traffic from the security group to pass through.</td>
</tr>
<tr>
<td>Deleting the node</td>
<td>The node will become unavailable.</td>
<td>This operation cannot be undone.</td>
<td></td>
</tr>
<tr>
<td>Reinstalling the OS</td>
<td>Node components are deleted, and the node becomes unavailable.</td>
<td>Reset the node. For details, see Resetting a Node.</td>
<td></td>
</tr>
<tr>
<td>Upgrading the node kernel</td>
<td>The node may be unavailable or the network may be abnormal. <strong>NOTE</strong> Node running depends on the system kernel version. Do not use <code>yum update</code> to update or reinstall the operating system kernel of a node unless necessary. (Reinstalling the operating system kernel using the original image or other images is a risky operation.)</td>
<td>If the OS is EulerOS 2.2, restore the node or network connectivity by referring to What Can I Do If the Container Network Becomes Unavailable After yum update Is Used to Upgrade the OS? If the OS is not EulerOS 2.2, you can reset the node. For details, see Resetting a Node.</td>
<td></td>
</tr>
<tr>
<td>Category</td>
<td>Operation</td>
<td>Impact</td>
<td>Solution</td>
</tr>
<tr>
<td>----------</td>
<td>-----------</td>
<td>--------</td>
<td>----------</td>
</tr>
<tr>
<td></td>
<td>Changing the node IP address</td>
<td>The node will become unavailable.</td>
<td>Change the IP address back to the original one.</td>
</tr>
<tr>
<td></td>
<td>Modifying parameters of core components (such as kubelet and kube-proxy)</td>
<td>The node may become unavailable, and components may be insecure if security-related configurations are modified.</td>
<td>Restore the parameter settings to the recommended values. For details, see Configuring Kubernetes Parameters.</td>
</tr>
<tr>
<td></td>
<td>Modifying OS configuration</td>
<td>The node may be unavailable.</td>
<td>Restore the configuration items or reset the node. For details, see Resetting a Node.</td>
</tr>
<tr>
<td></td>
<td>Deleting the opt directory, /var/ paas directory, or a data disk</td>
<td>The node will become unready.</td>
<td>You can reset the node. For details, see Resetting a Node.</td>
</tr>
<tr>
<td></td>
<td>Modifying the node directory permission and the container directory permission</td>
<td>The permissions will be abnormal.</td>
<td>You are not advised to modify the permissions. Restore the permissions if they are modified.</td>
</tr>
<tr>
<td></td>
<td>Formatting or partitioning disks on cluster nodes</td>
<td>The node will become unready.</td>
<td>You can reset the node. For details, see Resetting a Node.</td>
</tr>
<tr>
<td></td>
<td>Installing other software on nodes</td>
<td>This may cause exceptions on Kubernetes components installed on the node, and make the node unavailable.</td>
<td>Uninstall the software that has been installed and restore or reset the node. For details, see Resetting a Node.</td>
</tr>
</tbody>
</table>

### Networking and Load Balancing

**Table 2-2 High-risk operations and solutions**

<table>
<thead>
<tr>
<th>Operation</th>
<th>Impact</th>
<th>How to Avoid/Fix</th>
</tr>
</thead>
<tbody>
<tr>
<td>Changing the value of the kernel parameter net.ipv4.ip_forward to 0</td>
<td>The network becomes inaccessible.</td>
<td>Change the value to 1.</td>
</tr>
<tr>
<td>Operation</td>
<td>Impact</td>
<td>How to Avoid/Fix</td>
</tr>
<tr>
<td>--------------------------------------------------------------------------</td>
<td>-------------------------------------------------</td>
<td>---------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Changing the value of the kernel parameter \texttt{net.ipv4.tcp_tw_recycle} to 1.</td>
<td>The NAT service becomes abnormal.</td>
<td>Change the value to 0.</td>
</tr>
<tr>
<td>Not configuring the node security group to allow UDP packets to pass through port 53 of the container CIDR block</td>
<td>The DNS in the cluster cannot work properly.</td>
<td>Restore the security group by referring to \texttt{Buying a CCE Cluster} and allow traffic from the security group to pass through.</td>
</tr>
<tr>
<td>Creating a custom listener on the ELB console for the load balancer managed by CCE</td>
<td>The modified items are reset by CCE or the ingress is faulty.</td>
<td>Use the YAML file of the Service to automatically create a listener.</td>
</tr>
<tr>
<td>Binding a user-defined backend on the ELB console to the load balancer managed by CCE.</td>
<td></td>
<td>Do not manually bind any backend.</td>
</tr>
<tr>
<td>Changing the ELB certificate on the ELB console for the load balancer managed by CCE.</td>
<td></td>
<td>Use the YAML file of the ingress to automatically manage certificates.</td>
</tr>
<tr>
<td>Changing the listener name on the ELB console for the ELB listener managed by CCE.</td>
<td></td>
<td>Do not change the name of the ELB listener managed by CCE.</td>
</tr>
<tr>
<td>Changing the description of load balancers, listeners, and forwarding policies managed by CCE on the ELB console.</td>
<td></td>
<td>Do not modify the description of load balancers, listeners, or forwarding policies managed by CCE.</td>
</tr>
<tr>
<td>Delete CRD resources of network-attachment-definitions of default-network.</td>
<td>The container network is disconnected, or the cluster fails to be deleted.</td>
<td>If the resources are deleted by mistake, use the correct configurations to create the default-network resources.</td>
</tr>
</tbody>
</table>
### Logs

<table>
<thead>
<tr>
<th>Operation</th>
<th>Impact</th>
<th>Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deleting the <code>/tmp/ccs-log-collector/pos</code> directory on the host machine</td>
<td>Logs are collected repeatedly.</td>
<td>None</td>
</tr>
<tr>
<td>Deleting the <code>/tmp/ccs-log-collector/buffer</code> directory of the host machine</td>
<td>Logs are lost.</td>
<td>None</td>
</tr>
</tbody>
</table>

### EVS Disks

<table>
<thead>
<tr>
<th>Operation</th>
<th>Impact</th>
<th>Solution</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manually unmounting an EVS disk on the console</td>
<td>An I/O error is reported when the pod data is being written into the disk.</td>
<td>Delete the mount path from the node and schedule the pod again.</td>
<td>The file in the pod records the location where files are to be collected.</td>
</tr>
<tr>
<td>Unmounting the disk mount path on the node</td>
<td>Pod data is written into a local disk.</td>
<td>Remount the corresponding path to the pod.</td>
<td>The buffer contains log cache files to be consumed.</td>
</tr>
<tr>
<td>Operating EVS disks on the node</td>
<td>Pod data is written into a local disk.</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>
3.1 Cluster Overview

As application development shifts its technical base towards containers, it becomes increasingly important to orchestrate and manage resources. Kubernetes is an open-source, powerful container orchestration system. It provides a mechanism for containerized application deployment, planning, update, and maintenance, making it easier and more efficient to deploy and manage applications.

Cloud Container Engine (CCE) is a hosted Kubernetes service that simplifies the deployment and management of containerized applications. With CCE, you can easily create Kubernetes clusters, deploy containerized applications, and manage and maintain them.

Prerequisites

- Before creating the first cluster, ensure that a Virtual Private Cloud (VPC) has been created. If you already have a VPC available, skip this step.
  A VPC provides an isolated, configurable, and manageable virtual network for CCE clusters. For details, see Creating a VPC.
- The Container CIDR block and Service CIDR block have been reserved. CIDR blocks cannot be changed after the cluster is created. If you want to use another CIDR block, you need to create a new cluster.

Considerations for Creating a Cluster

Some basic resources are created during cluster creation, as described in the following table.

<table>
<thead>
<tr>
<th>Table 3-1 Basic resources created for a new cluster</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Resource</strong></td>
</tr>
<tr>
<td>Master nodes and related resources</td>
</tr>
</tbody>
</table>
## Resource |
<table>
<thead>
<tr>
<th>Resource</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ECS (optional)</strong></td>
</tr>
<tr>
<td>Description</td>
</tr>
<tr>
<td>An ECS corresponds to a cluster node that provides compute capacity.</td>
</tr>
<tr>
<td>An ECS is named in the format of <em>cluster name-random number</em>, which is user-defined. ECSs created in batches are named in the format of <em>cluster name-random number-a string of random number</em>.</td>
</tr>
</tbody>
</table>

| **Security group** |
| Description |
| Two security groups are created for a cluster: one for managing master nodes, and the other for managing worker nodes. |
| **WARNING** |
| Do not delete the security groups and related rules automatically configured during cluster creation. Otherwise, the cluster will exhibit unexpected behavior. |
| 1. Security group for master nodes |
| Naming rule: *Cluster name-cce-control-Random number* |
| Functions: |
| ● Allows outbound traffic. |
| ● Allows other nodes to access Kubernetes components on the master nodes. |
| 2. Security group for worker nodes |
| Naming rule: *Cluster name-cce-node-Random number* |
| Functions: |
| ● Allows outbound traffic. |
| ● Allows remote login to Linux or Windows OS using ports 22 and 3389. |
| ● Allows communication between Kubernetes components using ports 4789 and 10250. |
| ● Allows Kubernetes to expose ports (30000-32767) to external systems. |
| ● Allows communication between nodes in the same security group. |

| **Disk (optional)** |
| Description |
| Two disks are configured for each node. One is the system disk, and the other is the data disk used to run Docker. |

| **Elastic IP address (optional)** |
| Description |
| Any node that requires access to external networks must have an elastic IP address (EIP). |

## Cluster Advantages

Typically, applications are installed using plug-ins or scripts. However, the lifecycle of applications, such as running, configuration, and management, are bound to the current operating system, which does not facilitate operations such as...
upgrade, update, and rollback. You can also create VMs to implement some functions. However, VMs have portability.

A new method is to deploy containers. Containers are isolated from each other. Each container has its own file system. Processes of containers do not affect each other, and compute resources can be separated. Compared with VMs, containers can be quickly deployed and migrated between different clouds and operating systems of different versions as they are decoupled from underlying infrastructure and file systems.

Containers occupy less resources than VMs. Each application can be packaged into a container image. Such a binding can be an advantage. You can create a container image for an application in the build or release stage, and each application does not need to be combined with other applications or depend on the infrastructure of the production environment, a consistent environment can be provided from R&D, testing, to production. Containers are more lightweight and transparent than VMs, which facilitate monitoring and management.

A cluster is a collection of cloud resources required for running containers. It contains resources such as cloud servers (physical servers or VMs), load balancers, and virtual private clouds (VPCs). You can run your applications in a cluster. In CCE, multiple clusters can be created, each hosting multiple containers. Each container runs an application instance. The built-in load balancing policy is used to manage, discover, and access this group of application instances, requiring no manual configuration and handling.

**Cluster Architecture**

CCE uses standard Kubernetes clusters that feature a distributed architecture. A cluster consists of master and worker nodes, kubectl (client command line tool), and other additional items. At least one master node and several worker nodes will be created for a cluster, and all these nodes run in the Kubernetes cluster orchestration system.

The following figure presents the architecture of a CCE cluster.
**Figure 3-1 Cluster architecture**

- **Master**: A master node manages, schedules, and controls the cluster, and provides core Kubernetes services and service processes of application workloads. A master node consists of API server, scheduler, cluster state store, and controller-manager server.
- **Node**: A worker node is used to run containerized applications. A worker node consists of kubelet, kube-proxy, and container runtime.
- **kubectl**: used to interact with the API server through the CLI to add, delete, modify, and query resources in a cluster.
- **Replication controller**: used to scale the number of replicas.
- **Pod**: smallest and simplest unit in Kubernetes that you create or deploy. A pod encapsulates an application container (or, in some cases, multiple containers), storage resources, a unique network IP address, and options that govern how the container(s) should run.
- **Container**: a running instance of a Docker image. Multiple containers can run on one node. Containers are basically software processes but have separate namespaces and do not run directly on a host.
- **Label**: a key/value pair associated with an object, for example, a pod. A label identifies a special characteristic of the object, which can be helpful to users but has no meaning to the kernel system.
- **Service**: an abstraction that exposes a group of applications running on pods as network services.
When performing operations such as creating, deleting, and scaling clusters, do not change user permissions on the Identity and Access Management (IAM) console. Otherwise, these operations may fail.

Canal, the CNI plug-ins used by CCE nodes, uses a CIDR block as the CIDR block of the container network. This CIDR block can be configured during cluster creation and defaults to 172.16.0.0/16. The Docker service creates a docker0 bridge by default. The default docker0 address is 172.17.0.1. When creating a cluster, ensure that the CIDR block of the VPC in the cluster is different from those of the container network and the docker0 bridge. If VPC peering connections are used, ensure that the CIDR block of the peer VPC is different from those of the container network and the docker0 bridge.

Follow-up Procedure

For more information about CCE concepts and operations, see the following sections:

- Buying a CCE Cluster
- Connecting to a Cluster Using kubectl
- Creating a Deployment
- Setting a LoadBalancer Service

3.2 Cluster Lifecycle

This section describes the status of each cluster lifecycle, helping you understand the running status of the cluster in different phases.

<table>
<thead>
<tr>
<th>Status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Creating</td>
<td>A cluster is being created and is applying for cloud resources.</td>
</tr>
<tr>
<td>Normal</td>
<td>The cluster is running properly.</td>
</tr>
<tr>
<td>Scaling-out</td>
<td>A node is being added to the cluster.</td>
</tr>
<tr>
<td>Scaling-in</td>
<td>A node is being deleted from the cluster.</td>
</tr>
<tr>
<td>Hibernating</td>
<td>The cluster is hibernating.</td>
</tr>
<tr>
<td>Awaking</td>
<td>The cluster is being woken up.</td>
</tr>
<tr>
<td>Upgrading</td>
<td>The cluster is being upgraded.</td>
</tr>
<tr>
<td>Unavailable</td>
<td>The current cluster is unavailable.</td>
</tr>
<tr>
<td>Deleting</td>
<td>The cluster is being deleted.</td>
</tr>
</tbody>
</table>
3.3 Buying a CCE Cluster

On the CCE console, you can easily create Kubernetes clusters. Kubernetes can manage container clusters at scale. A cluster manages a group of node resources.

In CCE, you can create a CCE cluster to manage both VMs and bare metal servers (BMSs) as nodes, and heterogeneous nodes that are GPU-enabled and NPU-enabled. By using high-performance network models, hybrid clusters provide a multi-scenario, secure, and stable runtime environment for containers.

Notes and Constraints

- During the node creation, software packages are downloaded from OBS using the domain name. Therefore, you need to use a private DNS server to resolve the OBS domain name. Otherwise, the node fails to be created. Therefore, the subnet where the node resides must be configured with a private DNS server address so that the purchased node can use the private DNS server. When you create a subnet, the private DNS server is used by default. If you have changed the subnet DNS, ensure that the DNS server in the subnet can resolve the OBS domain name. Otherwise, you need to use the private DNS server.
- You can create a maximum of 50 clusters in a single region. If more clusters are required, you can click here to increase your quota.
- After a cluster is created, the following items cannot be changed:
  - Cluster type. For example, change a Kunpeng cluster to a CCE cluster.
  - Number of master nodes in the cluster.
  - AZ of a master node.
- Network configuration of the cluster, such as the VPC, subnet, container CIDR block, Service CIDR block, IPv6 settings, and kube-proxy (forwarding) settings.
- Network model. For example, change the tunnel network to the VPC network.

For more information, see Notes and Constraints.

Procedure

**Step 1** Log in to the CCE console. On the Dashboard page, click Buy Cluster. Alternatively, in the navigation pane, choose Resource Management > Clusters. Click Buy next to CCE cluster.

**Step 2** Set cluster parameters by referring to Table 3-3. Pay attention to the parameters marked with an asterisk (*).

<table>
<thead>
<tr>
<th>Table 3-3 Parameters for creating a cluster</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Parameter</strong></td>
</tr>
</tbody>
</table>
| Billing Mode | ● Yearly/Monthly: a prepaid billing mode suitable in scenarios where you have a good idea of what resources you will need during the billing period. Fees need to be paid in advance, but services will be less expensive. Yearly/monthly billed clusters cannot be deleted after creation. To stop using these clusters, go to the Billing Center and unsubscribe them.  
● Pay-per-use: a postpaid billing mode suitable in scenarios where resources will be billed based on usage frequency and duration. You can provision or delete resources at any time. This section uses the pay-per-use billing mode as an example. |
| Region | Select a region near you to ensure the lowest latency possible. |
| Enterprise project | This parameter is displayed only for enterprise users who have enabled the enterprise project function.  
After an enterprise project (for example, default) is selected, the cluster, nodes in the cluster, cluster security groups, node security groups, and elastic IPs (EIPs) of the automatically created nodes will be created in this enterprise project. After a cluster is created, you are advised not to modify the enterprise projects of nodes, cluster security groups, and node security groups in the cluster.  
An enterprise project facilitates project-level management and grouping of cloud resources and users. For more information, see Enterprise Management. |
| *Cluster Name | Name of the new cluster, which cannot be changed after the cluster is created.  
A cluster name contains 4 to 128 characters starting with a letter and not ending with a hyphen (-). Only lowercase letters, digits, and hyphens (-) are allowed. |
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version</td>
<td>Kubernetes community baseline version. The latest version is recommended. For details about version changelog, see Upgrade Overview. If a Beta version is available, you can use it for trial. However, it is not recommended for commercial use.</td>
</tr>
</tbody>
</table>
| Management Scale | Maximum number of worker nodes that can be managed by the master nodes of the current cluster. You can select 50 nodes, 200 nodes, or 1,000 nodes for your cluster, or 2,000 nodes if you are buying a cluster of v1.15.11 or later. The management scale cannot be changed after the cluster is created. If you select 1000 nodes, the master nodes of the cluster can manage a maximum of 1000 worker nodes. The configuration fee varies depending on the specifications of master nodes for different management scales. Each cluster contains at least one master node and at least one worker node. A node is a cloud server.  
  ● Master node: a node that controls worker nodes in the cluster. The master node is automatically created along with the cluster, and manages and schedules the entire cluster.  
  ● Worker node: a node purchased or accepted into a cluster by the user. The master nodes control your workloads. When a worker node is down, the master node migrates your workloads to another worker node. |
### Parameter | Description
--- | ---
Number of master nodes | 3: Three master nodes will be created to make the cluster highly available. If a master node is faulty, the cluster can still be available without affecting service functions. Click **Change**. In the dialog box displayed, you can configure the following parameters:

**Disaster recovery level**
- **AZ**: Master nodes are deployed in different AZs for disaster recovery.
- **Fault domain**: Master nodes are deployed in different failure domains in the same AZ for disaster recovery. This option is displayed only when the environment supports failure domains.
- **Host computer**: Master nodes are deployed on different hosts in the same AZ for disaster recovery.
- **Customize**: You can select different locations to deploy different master nodes. In the fault domain mode, master nodes must be in the same AZ.

1: Only one master node is created in the cluster, which cannot ensure SLA for the cluster. Single-master clusters (non-HA clusters) are not recommended for commercial scenarios. Click **Change**. In the **AZ Settings** dialog box, select an AZ for the master node.

**NOTE**
- You are advised to create multiple master nodes to improve the cluster DR capability in commercial scenarios.
- The multi-master mode cannot be changed after the cluster is created. A single-master cluster cannot be upgraded to a multi-master cluster. For a single-master cluster, if a master node is faulty, services will be affected.
- To ensure reliability, the multi-master mode is enabled by default for a cluster with 1,000 or more nodes.

*VPC | VPC where the cluster is located. The value cannot be changed after the cluster is created.
A VPC provides a secure and logically isolated network environment.
If no VPC is available, click **Create a VPC** to create a VPC. After the VPC is created, click the refresh icon.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| *Subnet   | Subnet where the node VM runs. The value cannot be changed after the cluster is created. A subnet provides dedicated network resources that are logically isolated from other networks for network security. If no subnet is available, click **Create a subnet** to create a subnet. After the subnet is created, click the refresh icon. For details about the relationship between VPCs, subnets, and clusters, see **Cluster Overview**.

During the node creation, software packages are downloaded from OBS using the domain name. Therefore, you need to use a private DNS server to resolve the OBS domain name. Otherwise, the node fails to be created. Therefore, the subnet where the node resides must be configured with a **private DNS server address** so that the purchased node can use the private DNS server. When you create a subnet, the private DNS server is used by default. If you have changed the subnet DNS, ensure that the DNS server in the subnet can resolve the OBS domain name. Otherwise, you need to use the private DNS server.

**The selected subnet cannot be modified after the cluster is created. Exercise caution when selecting a subnet.** |
<p>| IPv6      | By default, this parameter is disabled. <strong>This function is available only in clusters of v1.15 or later. For details, see <a href="https://example.com">Creating an IPv4/IPv6 Dual-Stack Cluster on CCE</a>.</strong> After this parameter is enabled, the container and service CIDR blocks with IPv6 addresses are automatically created. Cluster resources, including nodes and workloads, can be accessed through the IPv6 CIDR blocks. |</p>
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Network Model</td>
<td>After a cluster is created, the network model cannot be changed. Exercise caution when selecting a network model. For details about how to select a network model, see Selecting a Network Model When Creating a Cluster on CCE.</td>
</tr>
</tbody>
</table>
| VPC network     | In this network model, each node occupies one VPC route. The number of VPC routes supported by the current region and the number of container IP addresses that can be allocated to each node (that is, the maximum number of pods that can be created) are displayed on the console.  
  - The container network uses VPC routes to integrate with the underlying network. This network model is applicable to performance-intensive scenarios. However, each node occupies one VPC route, and the maximum number of nodes allowed in a cluster depends on the VPC route quota.  
  - Each node is assigned a CIDR block of a fixed size. VPC networks are free from packet encapsulation overheads and outperform container tunnel networks. In addition, as VPC routing includes routes to node IP addresses and the container CIDR block, container pods in the cluster can be directly accessed from outside the cluster.  
  **NOTE**  
  - In the VPC network model, extended CIDR blocks and network policies are not supported.  
  - When creating multiple clusters using the VPC network model in one VPC, select a CIDR block for each cluster that does not overlap with the VPC address or other container CIDR blocks. |
| Tunnel network  | Only nodes of the same type can be added when the tunnel network is used, that is, all nodes are VM nodes or bare metal nodes.  
  - The container network is an overlay tunnel network on top of a VPC network and uses the VXLAN technology. This network model is applicable when there is no high requirements on performance.  
  - VXLAN encapsulates Ethernet packets as UDP packets for tunnel transmission. Though at some cost of performance, the tunnel encapsulation enables higher interoperability and compatibility with advanced features (such as network policy-based isolation), meeting the requirements of most applications. |
### Parameter Description

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Container Network Segment</td>
<td>An IP address range that can be allocated to container pods. After the cluster is created, the value cannot be changed.</td>
</tr>
<tr>
<td></td>
<td>• If <strong>Automatically select</strong> is deselected, enter a CIDR block manually. If the CIDR block you specify conflicts with a subnet CIDR block, the system prompts you to select another CIDR block. The recommended CIDR blocks are 10.0.0.0/8-18, 172.16.0.0/16-18, and 192.168.0.0/16-18. <strong>If different clusters share a container CIDR block, an IP address conflict will occur and access to applications will fail.</strong></td>
</tr>
<tr>
<td></td>
<td>• If <strong>Automatically select</strong> is selected, the system automatically assigns a CIDR block that does not conflict with any subnet CIDR block. The mask of the container CIDR block must be appropriate. It determines the number of available nodes in a cluster. A too small mask value will cause the cluster to soon fall short of nodes. After the mask is set, the estimated maximum number of containers supported by the current CIDR block will be displayed. For details, see <a href="#">Which CIDR Blocks Does CCE Support?</a>.</td>
</tr>
<tr>
<td>Service Network Segment</td>
<td>An IP address range that can be allocated to Kubernetes Services. After the cluster is created, the value cannot be changed. The Service CIDR block cannot conflict with the created route. If they conflict, select another CIDR block.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Default</strong>: The default CIDR block 10.247.0.0/16 will be used.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Custom</strong>: Manually set a CIDR block and mask based on service requirements. The mask determines the maximum number of Service IP addresses available in the cluster. For details, see <a href="#">Which CIDR Blocks Does CCE Support?</a>.</td>
</tr>
<tr>
<td>Authorization Mode</td>
<td><strong>RBAC</strong> is selected by default and cannot be deselected. After RBAC is enabled, IAM users access resources in the cluster according to fine-grained permissions policies. For details, see <a href="#">Namespace Permissions (Kubernetes RBAC-based)</a>.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
</tr>
<tr>
<td>Authentication Mode</td>
<td>The authentication mechanism controls user permission on resources in a cluster. The X509-based authentication mode is enabled by default. X509 is a commonly used certificate format. If you want to perform permission control on the cluster, select <strong>Enhanced authentication</strong>. The cluster will identify users based on the header of the request for authentication. You need to upload your own <strong>CA certificate</strong>, <strong>client certificate</strong>, and <strong>client certificate private key</strong> (for details about how to create a certificate, see <strong>Certificates</strong>), and select <strong>I have confirmed that the uploaded certificates are valid</strong>. <strong>CAUTION</strong>&lt;br&gt;• Upload a file <strong>smaller than 1 MB</strong>. The CA certificate and client certificate can be in .crt or .cer format. The private key of the client certificate can only be uploaded <strong>unencrypted</strong>. &lt;br&gt;• The validity period of the client certificate must be longer than five years. &lt;br&gt;• The uploaded CA certificate is used for both the authentication proxy and the kube-apiserver aggregation layer configuration. <strong>If the certificate is invalid, the cluster cannot be created.</strong></td>
</tr>
<tr>
<td>Cluster Description</td>
<td>Optional. Enter the description of the new container cluster.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Advanced Settings</td>
<td>Click <strong>Advanced Settings</strong> to expand the details page. The following functions are supported (unsupported functions in current AZs are hidden):</td>
</tr>
<tr>
<td></td>
<td><strong>Service Forwarding Mode</strong></td>
</tr>
<tr>
<td></td>
<td>- <strong>iptables</strong>: Traditional kube-proxy uses iptables rules to implement Service load balancing. In this mode, too many iptables rules will be generated when many Services are deployed. In addition, non-incremental updates will cause a latency and even obvious performance issues in the case of heavy service traffic.</td>
</tr>
<tr>
<td></td>
<td>- <strong>ipvs</strong>: kube-proxy mode optimized by Huawei to achieve higher throughput and faster speed. This mode supports incremental updates and can keep connections uninterrupted during Service updates. It is suitable for large-sized clusters.</td>
</tr>
<tr>
<td></td>
<td>In this mode, when the ingress and Service use the same ELB instance, the ingress cannot be accessed from the nodes and containers in the cluster.</td>
</tr>
<tr>
<td></td>
<td><strong>NOTE</strong></td>
</tr>
<tr>
<td></td>
<td>- ipvs provides better scalability and performance for large clusters.</td>
</tr>
<tr>
<td></td>
<td>- Compared with iptables, ipvs supports more complex load balancing algorithms such as least load first (LLF) and weighted least connections (WLC).</td>
</tr>
<tr>
<td></td>
<td>- ipvs supports server health checking and connection retries.</td>
</tr>
<tr>
<td></td>
<td><strong>Resource Tags</strong></td>
</tr>
<tr>
<td></td>
<td>You can classify resources by their tags.</td>
</tr>
<tr>
<td></td>
<td>You can create predefined tags in Tag Management Service (TMS). Predefined tags are visible to all service resources that support the tagging function. You can use predefined tags to improve tag creation and migration efficiency. For details, see Creating Predefined Tags.</td>
</tr>
<tr>
<td></td>
<td><strong>CPU Policy</strong></td>
</tr>
<tr>
<td></td>
<td>This parameter is displayed only for clusters of v1.13.10-r0 and later.</td>
</tr>
<tr>
<td></td>
<td>- <strong>On</strong>: Exclusive CPU cores can be allocated to workload pods. Select On if your workload is sensitive to latency in CPU cache and scheduling.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Off</strong>: Exclusive CPU cores will not be allocated to workload pods. Select Off if you want a large pool of shareable CPU cores.</td>
</tr>
<tr>
<td></td>
<td>For details about CPU management policies, see Feature Highlight: CPU Manager.</td>
</tr>
<tr>
<td></td>
<td>After CPU Policy is enabled, workloads cannot be started or created on nodes after the node specifications are changed. For details about how to solve this problem, see What Should I Do.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
</tr>
<tr>
<td><strong>If I Fail to Restart or Create Workloads on a Node After Modifying the Node Specifications?</strong></td>
<td></td>
</tr>
<tr>
<td>Validity Period</td>
<td>For a <em>yearly/monthly billed</em> cluster, set the required duration.</td>
</tr>
</tbody>
</table>

**Step 3** Click Next: Create Node and set the following parameters.

- **Create Node**
  - **Create now**: Create a node when creating a cluster. Currently, only VM nodes are supported. If a node fails to be created, the cluster will be rolled back.
  - **Create later**: No node will be created. Only an empty cluster will be created. After the cluster is created, you can add nodes that run on VMs or bare metal servers.

- **Billing Mode**: Select **Yearly/Monthly** or **Pay-per-use**.
  - **Yearly/Monthly**: a prepaid billing mode, in which a resource is billed based on the purchase period. This mode is more cost-effective than the pay-per-use mode and applies if the resource usage period can be estimated.
  - **Pay-per-use**: a postpaid billing mode suitable in scenarios where resources will be billed based on usage duration. You can provision or delete resources at any time.

Nodes created along with the cluster must inherit the billing mode from the cluster. For example, if the billing mode of the cluster is pay-per-use, then nodes created along with the cluster must be billed on the pay-per-use basis. For details, see *Buying a Node*.

Yearly/monthly billed nodes cannot be deleted after creation. To stop using these nodes, go to the **Billing Center** and unsubscribe them.

- **Current Region**: geographic location of the nodes to be created.
- **AZ**: Set this parameter based on the site requirements. An AZ is a physical region where resources use independent power supply and networks. AZs are physically isolated but interconnected through an internal network.

You are advised to deploy worker nodes in different AZs after the cluster is created to make your workloads more reliable. When creating a cluster, you can deploy nodes only in one AZ.

**Figure 3-3 Worker nodes in different AZs**

- **Node Type**
- **VM node**: A VM node will be created in the cluster.
- **Bare-metal node**: Nodes running on bare metal servers (called BMS nodes) cannot be created along with a cluster. You can add BMS nodes to a cluster only after the cluster is created.

**NOTE**

BMS nodes can be created only after the CCE cluster is created and meets the following conditions:

- All IP addresses are not IPv6.
- The cluster version is v1.11.7 or later (with VPC network used), or v1.13.10 or later (with tunnel network used).
- BMS nodes in the cluster are billed on a yearly/monthly basis.

For details on how to buy BMS nodes, see [Buying a Node](#).

- **Node Name**: Enter a node name. A node name contains 1 to 56 characters starting with a lowercase letter and not ending with a hyphen (-). Only lowercase letters, digits, and hyphens (-) are allowed.

  If you change the node name on the ECS console after the node is created, be sure to synchronize the new node name from ECS to CCE. For details, see [Synchronizing Node Data](#).

- **Specifications**: Select node specifications that best fit your business needs.
  - **General-purpose**: provides a balance of computing, memory, and network resources. It is a good choice for many applications, such as web servers, workload development, workload testing, and small-scale databases.
  - **Memory-optimized**: provides higher memory capacity than general-purpose nodes and is suitable for relational databases, NoSQL, and other workloads that are both memory-intensive and data-intensive.
  - **General computing-basic**: provides a balance of computing, memory, and network resources and uses the vCPU credit mechanism to ensure baseline computing performance. Nodes of this type are suitable for applications requiring burstable high performance, such as light-load web servers, enterprise R&D and testing environments, and low- and medium-performance databases.
  - **GPU-accelerated**: provides powerful floating-point computing and is suitable for real-time, highly concurrent massive computing. Graphical processing units (GPUs) of P series are suitable for deep learning, scientific computing, and CAE. GPUs of G series are suitable for 3D animation rendering and CAD. **GPU-accelerated nodes can be added only to clusters of v1.11 or later**.
  - **High-performance computing**: provides stable and ultra-high computing performance and is suitable for scientific computing and workloads that demand ultra-high computing power and throughput.
  - **General computing-plus**: provides stable performance and exclusive resources to enterprise-class workloads with high and stable computing performance.
  - **Disk-intensive**: supports local disk storage and provides high network performance. It is designed for workloads requiring high throughput and data switching, such as big data workloads.
- **Ultra-high I/O**: delivers ultra-low SSD access latency and ultra-high IOPS performance. This type of specifications is ideal for high-performance relational databases, NoSQL databases (such as Cassandra and MongoDB), and Elasticsearch.

**Figure 3-4 Selecting node specifications**

To ensure node stability, CCE automatically reserves some resources to run necessary system components. For details, see *Formula for Calculating the Reserved Resources of a Node*.

- **OS**: Select an OS for the node to be created. In certain regions, only OSs are displayed and options Public image and Private image are unavailable.
  - **Public image**: Select an OS for the node.
    A public image is a standard, widely used image. It contains an OS and preinstalled public applications and is available to all users. For more information, see *Overview*.
  - **Private image** (OBT): A private image contains an OS or service data, preinstalled public applications, and the owner's private applications. It is available only to the user who created it. Private images are supported only for clusters of v1.15 or later.
    If no private image is available, create one by following the instructions provided in *Using a Private Image to Build a Worker Node Image (OBT)*.
  - **Shared image**: A shared image is a private image shared by another user. For details, see *Overview of Sharing Images*.

Reinstalling the OS or modifying OS configurations could make the node unavailable. Exercise caution when performing these operations. For details, see *High-Risk Operations and Solutions*.

- **System Disk**: Set the system disk space of the worker node. The value ranges from 40GB to 1024 GB. The default value is 40GB.
  
  By default, system disks support High I/O (SAS) and Ultra-high I/O (SSD) EVS disks. For details, see *EVS Disk Overview*.

  **Encryption**: Data disk encryption safeguards your data. Snapshots generated from encrypted disks and disks created using these snapshots automatically inherit the encryption function. This function is available only in certain regions.
- **Encryption** is not selected by default.
- After you select **Encryption**, you can select an existing key in the displayed **Encryption Setting** dialog box. If no key is available, click the link next to the drop-down box to create a key. After the key is created, click the refresh icon.

- **Data Disk**: Set the data disk space of the worker node. The value ranges from 100 GB to 32,768 GB. The default value is 100 GB. The EVS disk types provided for the data disk are the same as those for the system disk.

  **CAUTION**

  If the data disk is uninstalled or damaged, the Docker service becomes abnormal and the node becomes unavailable. You are advised not to delete the data disk.

- **LVM**: If this option is selected, CCE data disks are managed by the Logical Volume Manager (LVM). On this condition, you can adjust the disk space allocation for different resources. This option is selected for the first disk by default and cannot be unselected. You can choose to enable or disable LVM for new data disks.
  - This option is selected by default, indicating that LVM management is enabled.
  - You can deselect the check box to disable LVM management.

  **CAUTION**

  - Disk space of the data disks managed by LVM will be allocated according to the ratio you set.
  - When creating a node in a cluster of v1.13.10 or later, if LVM is not selected for a data disk, follow instructions in *Adding a Second Data Disk to a Node in a CCE Cluster* to fill in the pre-installation script and format the data disk. Otherwise, the data disk will still be managed by LVM.
  - When creating a node in a cluster earlier than v1.13.10, you must format the data disks that are not managed by LVM. Otherwise, either these data disks or the first data disk will be managed by LVM.

- **Encryption**: Data disk encryption provides powerful protection for your data. Snapshots generated from encrypted disks and disks created using these snapshots automatically inherit the encryption function.

  **This function is supported only for clusters of v1.13.10 or later in certain regions**, and is not displayed for clusters of v1.13.10 or earlier.
  - **Encryption** is not selected by default.
  - After you select **Encryption**, you can select an existing key in the displayed **Encryption Setting** dialog box. If no key is available, click
the link next to the drop-down box to create a key. After the key is created, click the refresh icon.

- **Add Data Disk**: Currently, a maximum of two data disks can be attached to a node. After the node is created, you can go to the ECS console to attach more data disks. This function is available only to clusters of certain versions.

- **Data disk space allocation**: Click to specify the resource ratio for **Kubernetes Space** and **User Space**. Disk space of the data disks managed by LVM will be allocated according to the ratio you set. This function is available only to clusters of certain versions.

  - **Kubernetes Space**: You can specify the ratio of the data disk space for storing Docker and kubelet resources. Docker resources include the Docker working directory, Docker images, and image metadata. Kubelet resources include pod configuration files, secrets, and emptyDirs.

  The Docker space size is determined by your service requirements. For details, see **Docker Disk Space**.

  - **User Space**: You can set the ratio of the disk space that is not allocated to Kubernetes resources and the path to which the user space is mounted.

**NOTE**

Note that the mount path cannot be `/`, `/home/paas`, `/var/paas`, `/var/lib`, `/var/script`, `/var/log`, `/mnt/paas`, or `/opt/cloud`, and cannot conflict with the system directories (such as `bin`, `lib`, `home`, `root`, `boot`, `dev`, `etc`, `lost+found`, `mnt`, `proc`, `sbin`, `srv`, `tmp`, `var`, `media`, `opt`, `selinux`, `sys`, and `usr`). Otherwise, the system or node installation will fail.

If the cluster version is v1.13.10-r0 or later and the node type is Disk-intensive or Ultra-high I/O, the following options are displayed for data disks:

- **EVS**: Parameters are the same as those when the node type is not Disk-intensive or Ultra-high I/O. For details, see **Data Disk** above.

- **Local disk**: Local disks may break down and do not ensure data reliability. It is recommended that you store service data in EVS disks, which are more reliable than local disks.

  Local disk parameters are as follows:

  - **Disk Mode**: If the node type is **disk-intensive**, the supported disk mode is HDD. If the node type is **ultra-high I/O**, the supported disk mode is SSD.

  - **Read/Write Mode**: When multiple local disks exist, you can set the read/write mode. The serial and sequential modes are supported. **Sequential** indicates that data is read and written in linear mode. When a disk is used up, the next disk is used. **Serial** indicates that data is read and written in striping mode, allowing multiple local disks to be read and written at the same time.

  - **Kubernetes Space**: You can specify the ratio of the data disk space for storing Docker and kubelet resources. Docker resources include
the Docker working directory, Docker images, and image metadata. Kubelet resources include pod configuration files, secrets, and emptyDirs.

- **User Space**: You can set the ratio of the disk space that is not allocated to Kubernetes resources and the path to which the user space is mounted.

---

**NOTICE**

- The ratio of disk space allocated to the Kubernetes space and user space must be equal to 100% in total. You can click to refresh the data after you have modified the ratio.
- By default, disks run in the direct-lvm mode. If data disks are removed, the loop-lvm mode will be used and this will impair system stability.

---

**Figure 3-5 Setting a local disk**

- **VPC**: A VPC where the current cluster is located. This parameter cannot be changed and is displayed only for clusters of v1.13.10-r0 or later.
- **Subnet**: A subnet improves network security by providing exclusive network resources that are isolated from other networks. You can select any subnet in the cluster VPC. Cluster nodes can belong to different subnets.

During the node creation, software packages are downloaded from OBS using the domain name. Therefore, you need to use a private DNS server to resolve the OBS domain name. Otherwise, the node fails to be created. Therefore, the subnet where the node resides must be configured with a private DNS server address so that the purchased node can use the private DNS server. When you create a subnet, the private DNS server is used by default. If you have changed the subnet DNS, ensure that the DNS server in the subnet can resolve the OBS domain name. Otherwise, you need to use the private DNS server.

When a node is added to an existing cluster, if an extended CIDR block is added to the VPC corresponding to the subnet and the subnet is an extended CIDR block, you need to add the following three security group rules to the master node security group (the group name is in the format of `Cluster name-cce-control-Random number`). These rules ensure that the nodes added to the cluster are available. (This step is not required if an extended CIDR block has been added to the VPC during cluster creation.)
- EIP: an independent public IP address. If the nodes to be created require public network access, select Automatically assign or Use existing. This parameter is not displayed when IPv6 is enabled for the cluster.

An EIP bound to the node allows public network access. EIP bandwidth can be modified at any time. An ECS without a bound EIP cannot access the Internet or be accessed by public networks. For details, see EIP Overview.

  - **Do not use**: A node without an EIP cannot be accessed from public networks. It can be used only as a cloud server for deploying services or clusters on a private network.

  - **Automatically assign**: An EIP with specified configurations is automatically assigned to each node. If the number of EIPs is less than the number of nodes, the EIPs are randomly bound to the nodes.

  Configure the EIP specifications, billing factor, bandwidth type, and bandwidth size as required. When creating an ECS, ensure that the elastic IP address quota is sufficient.

  - **Use existing**: Existing EIPs are assigned to the nodes to be created.

**NOTE**

By default, VPC’s SNAT feature is disabled for CCE. If SNAT is enabled, you do not need to use EIPs to access public networks. For details about SNAT, see Custom Policies.

- Shared Bandwidth: Select Do not use or Use existing. This parameter is displayed only when IPv6 is enabled for the cluster.

An EIP bound to the node allows public network access. EIP bandwidth can be modified at any time. An ECS without a bound EIP cannot access the Internet or be accessed by public networks.

- **Login Mode**: You can use a password or key pair.

  - **Password**: The default username is root. Enter the password for logging in to the node and confirm the password.

    Be sure to remember the password as you will need it when you log in to the node.
- **Key pair**: Select the key pair used to log in to the node. You can select a shared key.

  A key pair is used for identity authentication when you remotely log in to a node. If no key pair is available, click **Create a key pair**. For details on how to create a key pair, see [Creating a Key Pair](#).

---

**NOTICE**

When creating a node using a key pair, IAM users can select only the key pairs created by their own, regardless of whether these users are in the same group. For example, user B cannot use the key pair created by user A to create a node, and the key pair is not displayed in the drop-down list on the CCE console.

---

**Figure 3-6 Key pair**

- Advanced ECS Settings (optional): Click to show advanced ECS settings.
  - **ECS Group**: An ECS group logically groups ECSs. The ECSs in the same ECS group comply with the same policy associated with the ECS group.
    - **Anti-affinity**: ECSs in an ECS group are deployed on different physical hosts to improve service reliability.
    - **Fault domain**: ECSs in an ECS group are deployed in multiple failure domains so that a failure in one failure domain will not affect the ECSs in other failure domains, thereby improving service reliability. This option is displayed only when the environment supports failure domains. This option is not supported if a worker node is deployed in a random AZ.

Select an existing ECS group, or click **Create ECS Group** to create a new one. After the ECS group is created, click the refresh button.

- **Resource Tags**: By adding tags to resources, you can classify resources.

  You can create predefined tags in Tag Management Service (TMS). Predefined tags are visible to all service resources that support the tagging function. You can use predefined tags to improve tag creation and migration efficiency.

  CCE will automatically create the "CCE-Dynamic-Provisioning-Node=node id" tag. A maximum of 5 tags can be added.

- **Agency**: An agency is created by a tenant administrator on the IAM console. By creating an agency, you can share your cloud server resources with another account, or entrust a more professional person or team to manage your resources. For details on how to create an agency, see [Cloud Service Delegation](#). To authorize an ECS or BMS to call cloud services, select **Cloud service** as the agency type, click **Select**, and then select **ECS BMS**.
- **Pre-installation Script**: Enter a maximum of 1,000 characters. The script will be executed before Kubernetes software is installed. Note that if the script is incorrect, Kubernetes software may not be installed successfully. The script is usually used to format data disks.

- **Post-installation Script**: Enter a maximum of 1,000 characters. The script will be executed after Kubernetes software is installed and will not affect the installation. The script is usually used to modify Docker parameters.

- **Subnet IP Address**: Select "Automatically assign IP address (recommended)" or "Manually assigning IP addresses." 

- **Advanced Kubernetes Settings**: (Optional) Click to show advanced ECS settings.

  - **Max Pods**: maximum number of pods that can be created on a node, including the system's default pods. If the cluster uses the VPC network model, the maximum value is determined by the number of IP addresses that can be allocated to containers on each node. This limit prevents the node from being overloaded by managing too many pods.

  - **Maximum Data Space per Container**: maximum data space that can be used by a container. The value ranges from 10 GB to 80 GB. If the value of this field is larger than the data disk space allocated to Docker resources, the latter will override the value specified here. Typically, 90% of the data disk space is allocated to Docker resources. This parameter is displayed only for clusters of v1.13.10-r0 and later.

- **Nodes**: The value cannot exceed the management scale you select when configuring cluster parameters. Set this parameter based on service requirements and the remaining quota displayed on the page. Click to view the factors that affect the number of nodes to be added (depending on the factor with the minimum value). To apply for more quotas, click Increase quota.

- **Validity Period**: If the cluster billing mode is yearly/monthly, set the number of months or years for which you will use the new node.

**Step 4** Click Next: Install Add-on, and select the add-ons to be installed in the Install Add-on step.

System resource add-ons must be installed. Advanced functional add-ons are optional.

You can also install all add-ons after the cluster is created. To do so, choose Add-ons in the navigation pane of the CCE console and select the add-on you will install. For details, see Add-ons.

**Step 5** Click Next: Confirm. Read the product instructions and select I am aware of the above limitations. Confirm the configured parameters, specifications, and fees.

**Step 6** Click Submit.

If the cluster will be billed on a yearly/monthly basis, click Pay Now and follow on-screen prompts to pay the order.

It takes about 6 to 10 minutes to create a cluster. You can click Back to Cluster List to perform other operations on the cluster or click Go to Cluster Events to
view the cluster details. If the cluster status is Available, the cluster is successfully created.

---End

Related Operations

- After creating a cluster, you can use the Kubernetes command line (CLI) tool kubectl to connect to the cluster. For details, see Connecting to a Cluster Using kubectl.
- Add more nodes to the cluster. For details, see Buying a Node.
- Log in to a node. For details, see Logging In to a Node.
- Create a namespace. You can create multiple namespaces in a cluster and organize resources in the cluster into different namespaces. These namespaces serve as logical groups and can be managed separately. For more information about how to create a namespace for a cluster, see Namespaces.
- Create a workload. Once the cluster is created, you can use an image to create an application that can be accessed from public networks. For details, see Creating a Deployment or Creating a StatefulSet.
- Click the cluster name to view cluster details.

Table 3-4 Cluster details

<table>
<thead>
<tr>
<th>Tab</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cluster Details</td>
<td>View the details and operating status of the cluster.</td>
</tr>
<tr>
<td>Monitoring</td>
<td>You can view the CPU and memory allocation rates of all nodes in the cluster (that is, the maximum allocated amount), as well as the CPU usage, memory usage, and specifications of the master node(s).</td>
</tr>
<tr>
<td>Events</td>
<td>- View cluster events on the Events tab page.</td>
</tr>
<tr>
<td></td>
<td>- Set search criteria. For example, you can set the time segment or enter an event name to view corresponding events.</td>
</tr>
<tr>
<td>Auto Scaling</td>
<td>You can configure auto scaling to add or reduce worker nodes in a cluster to meet service requirements. For details, see Setting Cluster Auto Scaling. Clusters of v1.17 do not support auto scaling using AOM. You can use node pools for auto scaling. For details, see Node Pool Overview.</td>
</tr>
<tr>
<td>kubectl</td>
<td>To access a Kubernetes cluster from a PC, you need to use the Kubernetes command line tool kubectl. For details, see Connecting to a Cluster Using kubectl.</td>
</tr>
</tbody>
</table>
### 3.4 Buying a Kunpeng Cluster

Containers in CCE's Kunpeng clusters can run on cloud servers that use Arm architecture and Kunpeng processors. Kunpeng-accelerated cloud servers are easy to deploy and provide comparable scaling and scheduling performance as x86-based cloud servers at only a fraction of what x86-based cloud servers would cost.

CCE combines the technical strength of Kunpeng-accelerated servers and the multi-architecture container technology to further exploit the potential of cloud computing in fields of mobile Internet, Internet of Things (IoT), artificial intelligence (AI), and high-performance computing, bringing enterprise cloud users closer to the goal of "Build, Ship, and Run Any App, Anywhere". Enterprise cloud users can choose a computing architecture that is more affordable and best fits to their needs. They can also easily build, test, and run services across various computing architecture platforms.

### Notes and Constraints

- During the node creation, software packages are downloaded from OBS using the domain name. Therefore, you need to use a private DNS server to resolve the OBS domain name. Otherwise, the node fails to be created. Therefore, the subnet where the node resides must be configured with a **private DNS server address** so that the purchased node can use the private DNS server. When you create a subnet, the private DNS server is used by default. If you have changed the subnet DNS, ensure that the DNS server in the subnet can resolve the OBS domain name. Otherwise, you need to use the private DNS server.
- You can create a maximum of 50 clusters in a single region. If more clusters are required, you can click [here](#) to increase your quota.
- Kunpeng clusters do not support obsfs. Therefore, parallel file systems cannot be mounted.
- Kunpeng clusters are not supported in certain regions.

### Procedure

**Step 1** Log in to the CCE console. In the navigation pane on the left, choose **Resource Management > Clusters**.
**Step 2** In the Kunpeng Cluster card, click **Buy**.

**Figure 3-7** Buying a Kunpeng cluster

**Step 3** Set cluster parameters. Pay attention to the parameters marked with an asterisk (*).

**Table 3-5** Parameters for creating a cluster

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| **Billing Mode**    | ● **Yearly/Monthly**: a prepaid billing mode suitable in scenarios where you have a good idea of what resources you will need during the billing period. Fees need to be paid in advance, but services will be less expensive. Yearly/monthly billed clusters cannot be deleted after creation. To stop using these clusters, go to the Billing Center and unsubscribe them.  
● **Pay-per-use**: a postpaid billing mode suitable in scenarios where resources will be billed based on usage frequency and duration. You can provision or delete resources at any time. |
<p>| <strong>Region</strong>          | To minimize network latency and resource access time, select the nearest region. Cloud resources are region-specific and cannot be used across regions through internal network connections. |
| <strong>Enterprise project</strong> | This parameter is displayed only for enterprise users who have enabled the enterprise project function. After an enterprise project (for example, <strong>default</strong>) is selected, the cluster, nodes in the cluster, cluster security groups, node security groups, and elastic IPs (EIPs) of the automatically created nodes will be created in this enterprise project. After a cluster is created, you are advised not to modify the enterprise projects of nodes, cluster security groups, and node security groups in the cluster. An enterprise project facilitates project-level management and grouping of cloud resources and users. For more information, see <strong>Enterprise Management</strong>. |</p>
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>* Cluster Name</td>
<td>Name of the new cluster, which cannot be changed after the cluster is created. A cluster name contains 4 to 128 characters starting with a letter and not ending with a hyphen (-). Only lowercase letters, digits, and hyphens (-) are allowed.</td>
</tr>
<tr>
<td>Version</td>
<td>Kubernetes community baseline version. The latest version is recommended. For details about version changelog, see Upgrade Overview. If a Beta version is available, you can use it for trial. However, it is not recommended for commercial use.</td>
</tr>
<tr>
<td>Management Scale</td>
<td>Maximum number of worker nodes that can be managed by the master nodes of the current cluster. You can select 50 nodes, 200 nodes, or 1,000 nodes for your cluster. The management scale cannot be changed after the cluster is created. If you select 1000 nodes, the master nodes of the cluster can manage a maximum of 1000 worker nodes. The configuration fee varies depending on the specifications of master nodes for different management scales. Each cluster contains at least one master node and at least one worker node. A node is a cloud server. - Master node: a node that controls worker nodes in the cluster. The master node is automatically created along with the cluster, and manages and schedules the entire cluster. - Worker node: a node purchased or accepted into a cluster by the user. The master nodes control your workloads. When a worker node is down, the master node migrates your workloads to another worker node.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| Number of Master Nodes    | 3: Three master nodes will be created. If a master node is faulty, the cluster can still be available without affecting service functions. Click **Change**. In the **Disaster Recovery Settings** dialog box, select a DR level.  
  ● **AZ**: Master nodes are deployed in different AZs for disaster recovery.  
  ● **Fault domain**: Master nodes are deployed in different failure domains in the same AZ for disaster recovery. This option is displayed only when the environment supports failure domains.  
  ● **Host computer**: Master nodes are deployed on different hosts in the same AZ for disaster recovery.  
  ● **Customize**: You can select different locations to deploy different master nodes. In the fault domain mode, master nodes must be in the same AZ.  
  1: Only one master node is created in the cluster, which cannot ensure SLA for the cluster. Single-master clusters are not recommended for commercial scenarios. Click **Change**. In the **AZ Settings** dialog box, select an AZ for the master node.  
  **NOTE**  
  ● You are advised to create multiple master nodes to improve the cluster DR capability in commercial scenarios.  
  ● The multi-master mode cannot be changed after the cluster is created. A single-master cluster cannot be upgraded to a multi-master cluster. For a single-master cluster, if a master node is faulty, services will be affected.  
  ● To ensure reliability, the multi-master mode is enabled by default for a cluster with 1,000 or more nodes. |
| VPC                       | VPC where the cluster is located. The value cannot be changed after the cluster is created. A VPC provides a secure and logically isolated network environment.  
  If no VPC is available, click **Create a VPC** to create a VPC. After the VPC is created, click the refresh icon. For details, see **Creating a VPC**. |
| Subnet                    | Subnet where the node VM runs. The value cannot be changed after the cluster is created. A subnet provides dedicated network resources that are logically isolated from other networks for network security.  
  If no subnet is available, click **Create a subnet** to create a subnet. After the subnet is created, click the refresh icon. For details about the relationship between VPCs, subnets, and clusters, see **Cluster Overview**.  
  **Ensure that the DNS server in the subnet can resolve the OBS domain name. Otherwise, nodes cannot be created.** |
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Network Model</td>
<td>After a cluster is created, the network model cannot be changed. Exercise caution when selecting a network model. For details about how to select a network model, see <a href="#">Selecting a Network Model When Creating a Cluster on CCE</a>.</td>
</tr>
<tr>
<td>VPC network</td>
<td>In this network model, each node occupies one VPC route. The number of VPC routes supported by the current region and the number of container IP addresses that can be allocated to each node (that is, the maximum number of pods that can be created) are displayed on the console.</td>
</tr>
<tr>
<td></td>
<td>• The container network uses VPC routes to integrate with the underlying network. This network model is applicable to performance-intensive scenarios. The maximum number of nodes allowed in a cluster depends on the route quota in a VPC network.</td>
</tr>
<tr>
<td></td>
<td>• Each node is assigned a CIDR block of a fixed size. VPC networks are free from packet encapsulation overheads and outperform container tunnel networks. In addition, as VPC routing includes routes to node IP addresses and the container CIDR block, container pods in the cluster can be directly accessed from outside the cluster.</td>
</tr>
</tbody>
</table>
|               | **NOTE**  
|               | – In the VPC network model, extended CIDR blocks and network policies are not supported.  
|               | – When creating multiple clusters using the VPC network model in one VPC, select a CIDR block for each cluster that does not overlap with the VPC address or other container CIDR blocks. |
| Tunnel network | Only nodes of the same type can be added when the tunnel network is used, that is, all nodes are VM nodes or bare metal nodes.  
|               | • The container network is an overlay tunnel network on top of a VPC network and uses the VXLAN technology. This network model is applicable when there is no high requirements on performance.  
<p>|               | • VXLAN encapsulates Ethernet packets as UDP packets for tunnel transmission. Though at some cost of performance, the tunnel encapsulation enables higher interoperability and compatibility with advanced features (such as network policy-based isolation), meeting the requirements of most applications. |</p>
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Container Network Segment</td>
<td>An IP address range that can be allocated to container pods. After the cluster is created, the value cannot be changed.</td>
</tr>
<tr>
<td></td>
<td>- If <strong>Automatically select</strong> is deselected, enter a CIDR block manually. If the CIDR block you specify conflicts with a subnet CIDR block, the system prompts you to select another CIDR block. The recommended CIDR blocks are 10.0.0.0/8-18, 172.16.0.0/16-18, and 192.168.0.0/16-18. If different clusters share a container CIDR block, an IP address conflict will occur and access to the applications in the clusters may fail.</td>
</tr>
<tr>
<td></td>
<td>- If <strong>Automatically select</strong> is selected, the system automatically assigns a CIDR block that does not conflict with any subnet CIDR block.</td>
</tr>
<tr>
<td></td>
<td>The mask of the container CIDR block must be appropriate. It determines the number of available nodes in a cluster. A too small mask value will cause the cluster to soon fall short of nodes. After the mask is set, the estimated maximum number of nodes supported by the current CIDR block will be displayed. For details, see <a href="#">Which CIDR Blocks Does CCE Support?</a></td>
</tr>
<tr>
<td>Service Network Segment</td>
<td>An IP address range that can be allocated to Kubernetes Services. After the cluster is created, the value cannot be changed. The Service CIDR block cannot conflict with the created route. If they conflict, select another CIDR block.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Default</strong>: The default CIDR block 10.247.0.0/16 will be used.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Custom</strong>: Manually set a CIDR block and mask based on service requirements. The mask determines the maximum number of Service IP addresses available in the cluster.</td>
</tr>
<tr>
<td>Authorization Mode</td>
<td><strong>RBAC</strong> is selected by default and cannot be deselected. After RBAC is enabled, IAM users access resources in the cluster according to fine-grained permissions policies.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| Authentication Mode    | The authentication mechanism performs permission control on resources in a cluster. The X509-based authentication mode is enabled by default. X509 is a commonly used certificate format. If you want to perform permission control on the cluster, select **Enhanced authentication**. The cluster will identify users based on the header of the request for authentication. You need to upload your own **CA certificate**, **client certificate**, and **client certificate private key** (for details about how to create a certificate, see [Certificates](#)), and select I have **confirmed that the uploaded certificates are valid**. **CAUTION**  
  - Upload a file smaller than 1 MB. The CA certificate and client certificate can be in .crt or .cer format. The private key of the client certificate can only be uploaded **unencrypted**.  
  - The validity period of the client certificate must be longer than five years.  
  - The uploaded CA certificate is used for both the authentication proxy and the kube-apiserver aggregation layer configuration. If the certificate is invalid, the cluster cannot be created. |
<p>| Cluster Description    | Optional. Enter the description of the new container cluster. |</p>
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Advanced Settings</td>
<td>Click <strong>Advanced Settings</strong> to expand the details page. The following functions are supported (unsupported functions in current AZs are hidden):</td>
</tr>
<tr>
<td></td>
<td><strong>Service Forwarding Mode</strong></td>
</tr>
<tr>
<td></td>
<td>● <strong>iptables</strong>: Traditional kube-proxy uses iptables rules to implement Service load balancing. In this mode, too many iptables rules will be generated when many Services are deployed. In addition, non-incremental updates will cause a latency and even tangible performance issues in the case of service traffic spikes.</td>
</tr>
<tr>
<td></td>
<td>● <strong>ipvs</strong>: kube-proxy mode optimized by Huawei to achieve higher throughput and faster speed. This mode supports incremental updates and can keep connections uninterrupted during Service updates. It is suitable for large-sized clusters. In this mode, when the ingress and Service use the same ELB instance, the ingress cannot be accessed from the nodes and containers in the cluster.</td>
</tr>
<tr>
<td></td>
<td><strong>NOTE</strong></td>
</tr>
<tr>
<td></td>
<td>● ipvs provides better scalability and performance for large clusters.</td>
</tr>
<tr>
<td></td>
<td>● Compared with iptables, ipvs supports more complex load balancing algorithms such as least load first (LLF) and weighted least connections (WLC).</td>
</tr>
<tr>
<td></td>
<td>● ipvs supports server health checking and connection retries.</td>
</tr>
<tr>
<td></td>
<td><strong>Resource Tags</strong></td>
</tr>
<tr>
<td></td>
<td>By adding tags to resources, you can classify resources.</td>
</tr>
<tr>
<td></td>
<td>You can create predefined tags in Tag Management Service (TMS). Predefined tags are visible to all service resources that support the tagging function. You can use predefined tags to improve tag creation and migration efficiency. For details, see <strong>Creating Predefined Tags</strong>.</td>
</tr>
<tr>
<td>CPU Policy</td>
<td>This parameter is displayed only for clusters of v1.13.10-r0 and later.</td>
</tr>
<tr>
<td></td>
<td>● <strong>On</strong>: Exclusive CPU cores can be allocated to workload pods. Select <strong>On</strong> if your workload is sensitive to latency in CPU cache and scheduling.</td>
</tr>
<tr>
<td></td>
<td>● <strong>Off</strong>: Exclusive CPU cores will not be allocated to workload pods. Select <strong>Off</strong> if you want a large pool of shareable CPU cores.</td>
</tr>
<tr>
<td></td>
<td>For details about CPU management policies, see <strong>Feature Highlight: CPU Manager</strong>.</td>
</tr>
<tr>
<td></td>
<td>After <strong>CPU Policy</strong> is enabled, workloads cannot be started or created on nodes after the node specifications are changed. For details about how to solve this problem, see <strong>What Should I Do</strong></td>
</tr>
</tbody>
</table>
### If I Fail to Restart or Create Workloads on a Node After Modifying the Node Specifications?

**Validity Period**
For a yearly/monthly billed cluster, set the required duration.

---

**Step 4**

Click **Next: Create Node**. On the **Create Node** page, set the following parameters.

- **Create Node**
  - **Create now**: Create a node when creating a cluster. Currently, only VM nodes are supported. If a node fails to be created, the cluster will be rolled back.
  - **Create later**: No node will be created. Only an empty cluster will be created. After the cluster is created, you can add nodes that run on VMs or bare metal servers.

- **Billing Mode**: Select **Yearly/Monthly** or **Pay-per-use**.
  - **Yearly/Monthly**: a prepaid billing mode, in which a resource is billed based on the purchase period. This mode is more cost-effective than the pay-per-use mode and applies if the resource usage period can be estimated.
  - **Pay-per-use**: a postpaid billing mode suitable in scenarios where resources will be billed based on usage duration. You can provision or delete resources at any time.

Nodes created along with the cluster must inherit the billing mode from the cluster. For example, if the billing mode of the cluster is pay-per-use, then nodes created along with the cluster must be billed on the pay-per-use basis. For details, see **Buying a Node**.

Yearly/monthly billed nodes cannot be deleted after creation. To stop using these nodes, go to the **Billing Center** and unsubscribe them.

- **Current Region**: geographic location of the nodes to be created.
- **AZ**: Set this parameter based on the site requirements. An AZ is a physical region where resources use independent power supply and networks. AZs are physically isolated but interconnected through an internal network.

You are advised to deploy worker nodes in different AZs after the cluster is created to make your workloads more reliable. When creating a cluster, you can deploy nodes only in one AZ.

**Figure 3-8** Worker nodes in different AZs

- **Node Type**: VM node is selected by default.
Node Name: Enter a node name. A node name contains 1 to 56 characters starting with a lowercase letter and not ending with a hyphen (-). Only lowercase letters, digits, and hyphens (-) are allowed. If you change the node name on the ECS console after the node is created, be sure to synchronize the new node name from ECS to CCE. For details, see Synchronizing Node Data.

Specifications: Select node specifications that best fit your business needs.
- Kunpeng general computing-plus: This type of specifications is suitable for governments, enterprises, and the financial industry with strict requirements on security and privacy, for Internet applications with high requirements on network performance, for big data and HPC requiring a large number of vCPUs, and for website setups and e-Commerce requiring cost-effectiveness.
- Kunpeng memory-optimized: This type of specifications is suited for large-memory datasets with high network performance requirements. Using Huawei-proprietary Kunpeng 920 processors and high-speed intelligent Hi1822 NICs, the KM1 ECSs provide a maximum memory size of 480 GB based on DDR4 for large-memory applications with high requirements on network performance.

Figure 3-9 Selecting node specifications

NOTE
To ensure node stability, CCE automatically reserves some resources to run necessary system components. For details, see Formula for Calculating the Reserved Resources of a Node.

OS: Select the operating system (OS) of the nodes to be created.

System Disk: Set the system disk space of the worker node. The value ranges from 40GB to 1024 GB. The default value is 40GB.
By default, system disks support High I/O (SAS) and Ultra-high I/O (SSD) EVS disks. For details, see EVS Disk Overview.

Encryption: Data disk encryption safeguards your data. Snapshots generated from encrypted disks and disks created using these snapshots automatically inherit the encryption function. This function is available only in certain regions.
- Encryption is not selected by default.
After you select **Encryption**, you can select an existing key in the displayed **Encryption Setting** dialog box. If no key is available, click the link next to the drop-down box to create a key. After the key is created, click the refresh icon.

- **Data Disk**: Set the data disk space of the worker node. The value ranges from 100 GB to 32,768 GB. The default value is 100 GB. The types of EVS disks supported for data disks are the same as those for system disks.

---

**CAUTION**

If the data disk is uninstalled or damaged, the Docker service becomes abnormal and the node becomes unavailable. You are advised not to delete the data disk.

---

- **Add Data Disk**: Currently, a maximum of two data disks can be attached to a node. After the node is created, you can go to the ECS console to attach more data disks.
- **LVM**: If this option is selected, CCE data disks are managed by the Logical Volume Manager (LVM). On this condition, you can adjust the disk space allocation for different resources. This option is selected for the first disk by default and cannot be unselected. You can choose to enable or disable LVM for new data disks.
  - This option is selected by default, indicating that LVM management is enabled.
  - You can deselect the check box to disable LVM management.

---

**CAUTION**

- Disk space of the data disks managed by LVM will be allocated according to the ratio you set.
- When creating a node in a cluster of v1.13.10 or later, if **LVM** is not selected for a data disk, follow instructions in **Adding a Second Data Disk to a Node in a CCE Cluster** to fill in the pre-installation script and format the data disk. Otherwise, the data disk will still be managed by LVM.
- When creating a node in a cluster earlier than v1.13.10, you must format the data disks that are not managed by LVM. Otherwise, either these data disks or the first data disk will be managed by LVM.
- By default, disks run in the direct-lvm mode. If data disks are removed, the loop-lvm mode will be used and this will impair system stability.

---

- **Encryption**: Data disk encryption provides powerful protection for your data. Snapshots generated from encrypted disks and disks created using these snapshots automatically inherit the encryption function. **This function is supported only for clusters of v1.13.10 or later in certain regions**, and is not displayed for clusters of v1.13.10 or earlier.
- **Encryption** is not selected by default.

- After you select **Encryption**, you can select an existing key in the displayed **Encryption Setting** dialog box. If no key is available, click the link next to the drop-down box to create a key. After the key is created, click the refresh icon.

- **Data disk space allocation**: Click [Change Configuration](#) to specify the resource ratio for **Kubernetes Space** and **User Space**.

  - **Kubernetes Space**: You can specify the ratio of the data disk space for storing Docker and kubelet resources. Docker resources include the Docker working directory, Docker images, and image metadata. Kubelet resources include pod configuration files, secrets, and emptyDirs.

  - **User Space**: You can set the ratio of the disk space that is not allocated to Kubernetes resources and the path to which the user space is mounted.

**NOTE**

- The ratio of disk space allocated to the Kubernetes space and user space must be equal to 100% in total. You can click [refresh](#) to refresh the data after you have modified the ratio.

- **Path inside a node** cannot be set to the root directory /. Otherwise, the mounting fails. Mount paths can be as follows:
  - /opt/xxxx (excluding /opt/cloud)
  - /mnt/xxxx (excluding /mnt/paas)
  - /tmp/xxxx
  - /var/xxx (excluding key directories such as /var/lib, /var/script, and /var/paas)
  - /xxxx (It cannot conflict with the system directory, such as bin, lib, home, root, boot, dev, etc, lost+found, mnt, proc, sbin, srv, tmp, var, media, opt, selinux, sys and usr.)

Do not set this parameter to /home/xxxx, /var/xxxx, /var/lib, /var/script, /mnt/xxxx, or /opt/cloud. Otherwise, the system or node installation will fail.

*VPC*: A VPC where the current cluster is located. This parameter cannot be changed and is displayed only for clusters of v1.13.10-r0 or later.

*Subnet*: A subnet improves network security by providing exclusive network resources that are isolated from other networks. You can select any subnet in the cluster VPC. Cluster nodes can belong to different subnets.

During the node creation, software packages are downloaded from OBS using the domain name. Therefore, you need to use a private DNS server to resolve the OBS domain name. Otherwise, the node fails to be created. Therefore, the subnet where the node resides must be configured with a **private DNS server address** so that the purchased node can use the private DNS server. When you create a subnet, the private DNS server is used by default. If you have changed the subnet DNS, ensure that the DNS server in the subnet can resolve the OBS domain name. Otherwise, you need to use the private DNS server.
When a node is added to an existing cluster, if an extended CIDR block is added to the VPC corresponding to the subnet and the subnet is an extended CIDR block, you need to add the following three security group rules to the master node security group (the group name is in the format of \textit{Cluster name-cce-control-Random number}). These rules ensure that the nodes added to the cluster are available. (This step is not required if an extended CIDR block has been added to the VPC during cluster creation.)

- **EIP**: an independent public IP address. If the nodes to be created require public network access, select \textbf{Automatically assign} or \textbf{Use existing}. \textbf{This parameter is not displayed when IPv6 is enabled for the cluster.} An EIP bound to the node allows public network access. EIP bandwidth can be modified at any time. An ECS without a bound EIP cannot access the Internet or be accessed by public networks. For details, see \textbf{EIP Overview}.
  
  - \textbf{Do not use}: A node without an EIP cannot be accessed from public networks. It can be used only as a cloud server for deploying services or clusters on a private network.
  
  - \textbf{Automatically assign}: An EIP with specified configurations is automatically assigned to each node. If the number of EIPs is less than the number of nodes, the EIPs are randomly bound to the nodes. Configure the EIP specifications, billing factor, bandwidth type, and bandwidth size as required. When creating an ECS, ensure that the elastic IP address quota is sufficient.
  
  - \textbf{Use existing}: Existing EIPs are assigned to the nodes to be created.

- **NOTE**

  By default, VPC’s SNAT feature is disabled for CCE. If SNAT is enabled, you do not need to use EIPs to access public networks. For details about SNAT, see \textbf{Custom Policies}.

- **Shared Bandwidth**: Select \textbf{Do not use} or \textbf{Use existing}. \textbf{This parameter is displayed only when IPv6 is enabled for the cluster.} An EIP bound to the node allows public network access. EIP bandwidth can be modified at any time. An ECS without a bound EIP cannot access the Internet or be accessed by public networks.
- **Login Mode**: You can use a password or key pair.
  - **Password**: The default username is *root*. Enter the password for logging in to the node and confirm the password. Be sure to remember the password as you will need it when you log in to the node.
  - **Key pair**: Select the key pair used to log in to the node. You can select a shared key. A key pair is used for identity authentication when you remotely log in to a node. If no key pair is available, click **Create a key pair**. For details on how to create a key pair, see *Creating a Key Pair*.

---

**NOTICE**

When creating a node using a key pair, IAM users can select only the key pairs created by their own, regardless of whether these users are in the same group. For example, user B cannot use the key pair created by user A to create a node, and the key pair is not displayed in the drop-down list on the CCE console.

---

**Figure 3-10 Key pair**

---

- **Advanced ECS Settings** (optional): Click ⟷ to show advanced ECS settings.
  - **ECS Group**: An ECS group logically groups ECSs. The ECSs in the same ECS group comply with the same policy associated with the ECS group.
    - **Anti-affinity**: ECSs in an ECS group are deployed on different physical hosts to improve service reliability.
    - **Fault domain**: ECSs in an ECS group are deployed in multiple failure domains so that a failure in one failure domain will not affect the ECSs in other failure domains, thereby improving service reliability. This option is displayed only when the environment supports failure domains. This option is not supported if a worker node is deployed in a random AZ.

Select an existing ECS group, or click **Create ECS Group** to create a new one. After the ECS group is created, click the refresh button.

- **Resource Tags**: By adding tags to resources, you can classify resources. You can create predefined tags in Tag Management Service (TMS). Predefined tags are visible to all service resources that support the tagging function. You can use predefined tags to improve tag creation and migration efficiency.

CCE will automatically create the "CCE-Dynamic-Provisioning-Node=node id" tag. A maximum of 5 tags can be added.
- **Agency**: An agency is created by a tenant administrator on the IAM console. By creating an agency, you can share your cloud server resources with another account, or entrust a more professional person or team to manage your resources. For details on how to create an agency, see Cloud Service Delegation. To authorize an ECS or BMS to call cloud services, select Cloud service as the agency type, click Select, and then select ECS BMS.

- **Pre-installation Script**: Enter a maximum of 1,000 characters.
  
  The script will be executed before Kubernetes software is installed. Note that if the script is incorrect, Kubernetes software may not be installed successfully. The script is usually used to format data disks.

- **Post-installation Script**: Enter a maximum of 1,000 characters.
  
  The script will be executed after Kubernetes software is installed and will not affect the installation. The script is usually used to modify Docker parameters.

- **Subnet IP Address**: Select Automatically assign IP address (recommended) or Manually assigning IP addresses.

  - **Advanced Kubernetes Settings**: (Optional) Click to show advanced ECS settings.

    - **Max Pods**: maximum number of pods that can be created on a node, including the system's default pods. If the cluster uses the VPC network model, the maximum value is determined by the number of IP addresses that can be allocated to containers on each node.
      
      This limit prevents the node from being overloaded by managing too many pods.

    - **Maximum Data Space per Container**: maximum data space that can be used by a container. The value ranges from 10 GB to 80 GB. If the value of this field is larger than the data disk space allocated to Docker resources, the latter will override the value specified here. Typically, 90% of the data disk space is allocated to Docker resources. This parameter is displayed only for clusters of v1.13.10-r0 and later.

    - **Nodes**: The value cannot exceed the management scale you select when configuring cluster parameters. Set this parameter based on service requirements and the remaining quota displayed on the page. Click to view the factors that affect the number of nodes to be added (depending on the factor with the minimum value). To apply for more quotas, click Increase quota.

    - **Validity Period**: If the cluster billing mode is yearly/monthly, set the number of months or years for which you will use the new node.

**Step 5** Click Next: Install Add-on, and select the add-ons to be installed in the Install Add-on step.

System resource add-ons must be installed. Advanced functional add-ons are optional.

You can also install all add-ons after the cluster is created. To do so, choose Add-ons in the navigation pane of the CCE console and select the add-on you will install. For details, see Add-ons.
Step 6 Click Next: Confirm. Read the product instructions and select I am aware of the above limitations. Confirm the configured parameters, specifications, and fees.

Step 7 Click Submit.

If the cluster is billed on a yearly/monthly basis, click Pay Now and follow on-screen prompts to pay the order.

It takes about 6 to 10 minutes to create a cluster. You can click Back to Cluster List to perform other operations on the cluster or click Go to Cluster Events to view the cluster details.

Step 8 If the cluster status is Available, the Kunpeng cluster is successfully created, and the Kunpeng icon is displayed in front of the cluster name.

Related Operations

- After creating a cluster, you can use the Kubernetes command line (CLI) tool kubectl to connect to the cluster. For details, see Connecting to a Cluster Using kubectl.
- Add more nodes to the cluster. For details, see Buying a Node.
- Log in to a node. For details, see Logging In to a Node.
- Create a namespace. You can create multiple namespaces in a cluster and organize resources in the cluster into different namespaces. These namespaces serve as logical groups and can be managed separately. For more information about how to create a namespace for a cluster, see Namespaces.
- Create a workload. Once the cluster is created, you can use an image to create an application that can be accessed from public networks. For details, see Creating a Deployment or Creating a StatefulSet.
- Click the cluster name to view cluster details.

Table 3-6 Cluster details

<table>
<thead>
<tr>
<th>Tab</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cluster Details</td>
<td>View the details and operating status of the cluster.</td>
</tr>
</tbody>
</table>
### Monitoring

You can view the CPU and memory allocation rates of all nodes in the cluster (that is, the maximum allocated amount), as well as the CPU usage, memory usage, and specifications of the master node(s).

### Events

- View cluster events on the **Events** tab page.
- Set search criteria. For example, you can set the time segment or enter an event name to view corresponding events.

### Auto Scaling

You can configure auto scaling to add or reduce worker nodes in a cluster to meet service requirements. For details, see [Setting Cluster Auto Scaling](#).

Clusters of v1.17 do not support auto scaling using AOM. You can use node pools for auto scaling. For details, see [Node Pool Overview](#).

### kubectl

To access a Kubernetes cluster from a PC, you need to use the Kubernetes command line tool **kubectl**. For details, see [Connecting to a Cluster Using kubectl](#).

### Resource Tags

You can add resource tags to classify resources. You can create predefined tags in Tag Management Service (TMS). Predefined tags are visible to all service resources that support the tagging function. You can use predefined tags to improve tag creation and migration efficiency.

CCE will automatically create the "CCE-Dynamic-Provisioning-Node=Node ID" tag. A maximum of 5 tags can be added.

---

## 3.5 Using kubectl to Run a Cluster

### 3.5.1 Connecting to a Cluster Using kubectl

#### Scenario

This section uses a CCE cluster as an example to describe how to connect to a CCE cluster using kubectl or CloudShell.

#### Permission Description

When you access a cluster using kubectl, CCE uses the kubeconfig.json file generated on the cluster for authentication. The file contains user information, based on which CCE determines which Kubernetes resources can be accessed by kubectl. The permissions recorded in a kubeconfig.json file vary from user to user.

For details about user permissions, see [Cluster Permissions (IAM-based) and Namespace Permissions (Kubernetes RBAC-based)](#).
When using kubectl in CloudShell, the kubectl permissions are determined by the user that logs in.

**Using CloudShell**

CloudShell is a web shell used to manage and maintain cloud resources. CCE allows you to use CloudShell to connect to clusters and use kubectl in CloudShell to access clusters.

**NOTE**

The kubectl certificate in CloudShell is valid for one day. You can reset the validity period by accessing CloudShell from the CCE console.

**Figure 3-11 CloudShell**

**Figure 3-12 Using kubectl in CloudShell**

**Using kubectl**

**Background**

To connect a client to a Kubernetes cluster, you can use kubectl. For details, see **Install Tools**.

**Prerequisites**

CCE allows you to access a cluster through a **VPC network** or a **public network**.
VPC internal access: Clusters in the same VPC can access each other.

Public network access: You need to prepare an ECS that can connect to a public network.

**NOTICE**

If public network access is used, the kube-apiserver of the cluster will be exposed to the public network and may be attacked. You are advised to configure Advanced Anti-DDoS for the EIP of the node where the kube-apiserver is located.

**Downloading kubectl**

You need to download kubectl and configuration file, copy the file to your client, and configure kubectl. After the configuration is complete, you can use kubectl to access your Kubernetes clusters.

Go to the Kubernetes release page to download kubectl corresponding to the cluster version or a later version.

**Installing and configuring kubectl**

**Step 1** Log in to the CCE console, click Resource Management > Clusters, and choose Command Line Tool > Kubectl under the cluster to be connected.

**Step 2** On the Kubectl tab page of the cluster details page, connect to the cluster as prompted.

**NOTE**

- You can download the kubectl configuration file kubeconfig.json on the Kubectl tab page. The validity period of the configuration file can be set to 1 to 10,950 days (30 years). This file is used for user cluster authentication. If the file is leaked, your clusters may be attacked. When the file is leaked, you can replace the authentication credential by updating the certificate.
- EIPs are not fixed IP addresses. Therefore, the EIP cannot be preset in the server certificate. If your cluster is accessed from the Internet, the server verification function cannot be enabled on the client. To enable the client to verify the server, use the VPC access mode.
- The Kubernetes permissions assigned by the configuration file downloaded by IAM users are the same as those assigned to the IAM users on the CCE console.
- Download the client file when downloading kubectl.
When you use kubectl to create or query Kubernetes resources, the following output is returned:

```
# kubectl get deploy Error from server (Forbidden): deployments.apps is forbidden: User "0c97ac3cb280f4d91fa7c0096739e1f8" cannot list resource "deployments" in API group "apps" in the namespace "default"
```

The cause is that the user does not have the permissions to operate the Kubernetes resources. For details about how to assign permissions, see [Namespace Permissions (Kubernetes RBAC-based)](#).

**Related Operations**

After connecting to the cluster, you can use Kubernetes to manage workloads. For details, see [kubectl Usage Guide](#).

### 3.5.2 Configuring High Availability of kube-dns/CoreDNS Using kubectl

**Scenario**

Use the Kubernetes command line tool (kubectl) to configure high availability of kube-dns/CoreDNS.

kube-dns/CoreDNS provides the Domain Name Service (DNS) for clusters. It is advised to configure multiple kube-dns/CoreDNS for a cluster. If there is only one kube-dns/CoreDNS in a cluster, the entire cluster will not run properly once the kube-dns/CoreDNS is down.

---

**Common Issue (Error from server Forbidden)**

When you use kubectl to create or query Kubernetes resources, the following output is returned:

```
# kubectl get deploy Error from server (Forbidden): deployments.apps is forbidden: User
"0c97ac3cb280f4d91fa7c0096739e1f8" cannot list resource "deployments" in API group "apps" in the
namespace "default"
```

The cause is that the user does not have the permissions to operate the Kubernetes resources. For details about how to assign permissions, see [Namespace Permissions (Kubernetes RBAC-based)](#).

**Related Operations**

After connecting to the cluster, you can use Kubernetes to manage workloads. For details, see [kubectl Usage Guide](#).

### 3.5.2 Configuring High Availability of kube-dns/CoreDNS Using kubectl

**Scenario**

Use the Kubernetes command line tool (kubectl) to configure high availability of kube-dns/CoreDNS.

kube-dns/CoreDNS provides the Domain Name Service (DNS) for clusters. It is advised to configure multiple kube-dns/CoreDNS for a cluster. If there is only one kube-dns/CoreDNS in a cluster, the entire cluster will not run properly once the kube-dns/CoreDNS is down.

---

**Common Issue (Error from server Forbidden)**

When you use kubectl to create or query Kubernetes resources, the following output is returned:

```
# kubectl get deploy Error from server (Forbidden): deployments.apps is forbidden: User
"0c97ac3cb280f4d91fa7c0096739e1f8" cannot list resource "deployments" in API group "apps" in the
namespace "default"
```

The cause is that the user does not have the permissions to operate the Kubernetes resources. For details about how to assign permissions, see [Namespace Permissions (Kubernetes RBAC-based)](#).

**Related Operations**

After connecting to the cluster, you can use Kubernetes to manage workloads. For details, see [kubectl Usage Guide](#).

### 3.5.2 Configuring High Availability of kube-dns/CoreDNS Using kubectl

**Scenario**

Use the Kubernetes command line tool (kubectl) to configure high availability of kube-dns/CoreDNS.

kube-dns/CoreDNS provides the Domain Name Service (DNS) for clusters. It is advised to configure multiple kube-dns/CoreDNS for a cluster. If there is only one kube-dns/CoreDNS in a cluster, the entire cluster will not run properly once the kube-dns/CoreDNS is down.

---

**Common Issue (Error from server Forbidden)**

When you use kubectl to create or query Kubernetes resources, the following output is returned:

```
# kubectl get deploy Error from server (Forbidden): deployments.apps is forbidden: User
"0c97ac3cb280f4d91fa7c0096739e1f8" cannot list resource "deployments" in API group "apps" in the
namespace "default"
```

The cause is that the user does not have the permissions to operate the Kubernetes resources. For details about how to assign permissions, see [Namespace Permissions (Kubernetes RBAC-based)](#).

**Related Operations**

After connecting to the cluster, you can use Kubernetes to manage workloads. For details, see [kubectl Usage Guide](#).
Prerequisites

The cluster is accessible from a public network, or the cluster and the client are in the same VPC.

Procedure

**Step 1** Log in to the CCE console, choose **Resource Management > Clusters**, and click **Command Line Tool > Kubectl** under the cluster to be connected.

**Step 2** Set the API access mode for the cluster.

**Step 3** Configure kubectl.

After kubectl is successfully configured, you can use it to manually configure high availability of kube-dns/CoreDNS.

**Step 4** Log in to the client.

**Step 5** Edit the deployment configuration file of kube-dns/CoreDNS.

The following uses CoreDNS as an example:

```
kubectl edit deployment coredns -n kube-system
```

Change the value of **replicas** in the **spec** section in the deployment configuration file to the number of kube-dns/CoreDNS instances required.

Example:

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  annotations:
    deployment.kubernetes.io/revision: "1"
  creationTimestamp: 2019-02-11T09:36:04Z
  generation: 1
  labels:
    app: coredns
    kubernetes-app: coredns
    kubernetes.io/cluster-service: "true"
    kubernetes.io/name: CoreDNS
    release: cceaddon-coredns
  name: coredns
  namespace: kube-system
  resourceVersion: "1927"
  selfLink: /apis/apps/v1/namespaces/kube-system/deployments/coredns
  uid: 737b9296-2de0-11e9-b629-fa163e7fb882
spec:
  progressDeadlineSeconds: 600
  replicas: 2
  revisionHistoryLimit: 10
  selector:
    matchLabels:
      app: coredns
      kubernetes-app: coredns
  strategy:
    rollingUpdate:
      maxSurge: 10%
      maxUnavailable: 0
    type: RollingUpdate
  template:
    metadata:
```

3.5.3 Common kubectl Commands

Getting Started

get

The *get* command displays one or many resources of a cluster.

This command prints a table of the most important information about all resources, including cluster nodes, running pods, replication controllers, and services.

**NOTICE**

A cluster can have multiple namespaces. If no namespace is specified, this command will run with the `--namespace=default` flag.

Examples:

To list all pods with detailed information:

```
kubectl get po -o wide
```

To display pods in all namespaces:

```
kubectl get po --all-namespaces
```

To list labels of pods in all namespaces:

```
kubectl get po --show-labels
```

To list all namespaces of the node:

```
kubectl get namespace
```

**NOTE**

To list information of other nodes, run this command with the `-s` flag. To list a specified type of resources, add the resource type to this command, for example, `kubectl get rc`, `kubectl get svc`, `kubectl get nodes`, and `kubectl get deploy`.

To list a pod with a specified name in YAML output format:

```
kubectl get po <podname> -o yaml
```

To list a pod with a specified name in JSON output format:

```
kubectl get po <podname> -o json
kubectl get po rc-nginx-2-btv4j -o=custom-columns=LABELS:.metadata.labels.app
```

**NOTE**

`LABELS` indicates a comma separated list of user-defined column titles. `metadata.labels.app` indicates the data to be listed in either YAML or JSON output format.
The `create` command creates a cluster resource from a file or input.

If there is already a resource descriptor (a YAML or JSON file), you can create the resource from the file by running the following command:

```
kubectl create -f filename
```

**expose**

The `expose` command exposes a resource as a new Kubernetes service. Possible resources include a pod, replication controller, service, and deployment.

```
kubectl expose deployment deployname --port=81 --type=NodePort --target-port=80 --name=service-name
```

**NOTE**

The example command creates a service of NodePort type for the deployment with the name specified in `deployname`. The service will serve on port 81 specified in `-port` and connect to the containers on port 80 specified in `-target-port`. More specifically, the service is reachable at `<cluster-internal IP address>:<port>`, and containers are reachable at `<node IP address>:<target-port>`.

**run**

Examples:

To run a particular image in the cluster:

```
kubectl run deployname --image=nginx:latest
```

To run a particular image using a specified command:

```
kubectl run deployname --image=busybox --command -- ping baidu.com
```

**set**

The `set` command configures object resources.

Example:

To change the image of a deployment with the name specified in `deployname` to image 1.0:

```
kubectl set image deploy deployname containername=containername:1.0
```

**edit**

The `edit` command edits a resource from the default editor.

Examples:

To update a pod:

```
kubectl edit po po-nginx-btv4j
```

The example command yields the same effect as the following command:

```
kubectl get po po-nginx-btv4j -o yaml >> /tmp/nginx-tmp.yaml
vim /tmp/nginx-tmp.yaml
/*do some changes here */
kubectl replace -f /tmp/nginx-tmp.yaml
```

**explain**

The `explain` command views documents or reference documents.

Example:
To get documentation of pods or services:

```
kubectl explain pods,svc
```

**delete**

The **delete** command deletes resources by resource name or label.

Example:

To delete a pod with minimal delay:

```
kubectl delete po podname --now
kubectl delete -f nginx.yaml
kubectl delete deployment deployname
```

**Deployment Commands**

**rolling-update**

**rolling-update** is a very important command. It updates a running service with zero downtime. Pods are incrementally replaced by new ones. One pod is updated at a time. The old pod is deleted only after the new pod is up. New pods must be distinct from old pods by name, version, and label. Otherwise, an error message will be reported.

```
kubectl rolling-update poname -f newfilename
kubectl rolling-update poname -image=image:v2
```

If any problem occurs during the rolling update, run the command with the **rollback** flag to abort the rolling update and revert to the previous pod.

```
kubectl rolling-update poname -rollback
```

**rollout**

The **rollout** command manages the rollout of a resource.

Examples:

To check the rollout status of a particular deployment:

```
kubectl rollout status deployment/deployname
```

To view the rollout history of a particular deployment:

```
kubectl rollout history deployment/deployname
```

To roll back to the previous deployment: (by default, a resource is rolled back to the previous version)

```
kubectl rollout undo deployment/test-nginx
```

**scale**

The **scale** command sets a new size for a resource by adjusting the number of resource replicas.

```
kubectl scale deployment deployname --replicas=newnumber
```

**autoscale**

The **autoscale** command automatically chooses and sets the number of pods. This command specifies the range for the number of pod replicas maintained by a replication controller. If there are too many pods, the replication controller
terminates the extra pods. If there is too few, the replication controller starts more pods.

`kubectl autoscale deployment deployname --min=minnumber --max=maxnumber`

**Cluster Management Commands**

**cordon, drain, uncordon**

If a node to be upgraded is running many pods or is already down, perform the following steps to prepare the node for maintenance:

**Step 1** Run the `cordon` command to mark a node as unschedulable. This means that new pods will not be scheduled onto the node.

```
kubectl cordon nodename
```

`nodename` is the IP address of the node.

**Step 2** Run the `drain` command to smoothly migrate the running pods from the node to another node.

```
kubectl drain nodename --ignore-daemonsets --ignore-emptydir
```

`ignore-emptydir` ignores the pods that use emptyDirs.

**Step 3** Perform maintenance operations on the node, such as upgrading the kernel and upgrading Docker.

**Step 4** After node maintenance is completed, run the `uncordon` command to mark the node as schedulable.

```
kubectl uncordon nodename
```

----End

**cluster-info**

To display the add-ons running in the cluster:

```
kubectl cluster-info
```

To dump current cluster information to stdout:

```
kubectl cluster-info dump
```

**top**

The `top` command displays resource (CPU/memory/storage) usage. This command requires Heapster to be correctly configured and working on the server.

**taint**

The `taint` command updates the taints on one or more nodes.

**certificate**

The `certificate` command modifies the certificate resources.

**Fault Diagnosis and Debugging Commands**

**describe**

The `describe` command is similar to the `get` command. The difference is that the `describe` command shows details of a specific resource or group of resources,
whereas the `get` command lists one or more resources in a cluster. The `describe` command does not support the `-o` flag. For resources of the same type, resource details are printed out in the same format.

**NOTE**

If the information about a resource is queried, you can use the `get` command to obtain more detailed information. If you want to check the status of a specific resource, for example, to check if a pod is in the running state, run the `describe` command to show more detailed status information.

```
kubectl describe po <podname>
```

**logs**

The `logs` command prints logs for a container in a pod or specified resource to stdout. To display logs in the `tail -f` mode, run this command with the `-f` flag.

```
kubectl logs -f podname
```

**exec**

The kubectl `exec` command is similar to the Docker `exec` command and executes a command in a container. If there are multiple containers in a pod, use the `-c` flag to choose a container.

```
kubectl exec -it podname bash
kubectl exec -it podname -c containername bash
```

**port-forward**

The `port-forward` command forwards one or more local ports to a pod.

Example:

To listen on ports 5000 and 6000 locally, forwarding data to/from ports 5000 and 6000 in the pod:

```
kubectl port-forward podname 5000:6000
```

**proxy**

The `proxy` command creates a proxy server between localhost and the Kubernetes API server.

Example:

To enable the HTTP REST APIs on the master node:

```
kubectl proxy -accept-hosts=’.*’ -port=8001 -address=’0.0.0.0’
```

**cp**

The `cp` command copies files and directories to and from containers.

```
cp filename newfilename
```

**auth**

The `auth` command inspects authorization.

**attach**

The `attach` command is similar to the `logs -f` command and attaches to a process that is already running inside an existing container. To exit, run the `ctrl-c`
command. If a pod contains multiple containers, to view the output of a specific container, use the -c flag and containername following podname to specify a container.

```
kubectl attach podname -c containername
```

### Advanced Commands

#### replace

The replace command updates or replaces an existing resource by attributes including the number of replicas, labels, image versions, and ports. You can directly modify the original YAML file and then run the replace command.

```
kubectl replace -f filename
```

---

**NOTICE**

Resource names cannot be updated. After a pod label is updated, pods with the original label will fall out of the scope of the replication controller using the new label selector. The replication controller notices that some pods no longer match its new label selector and spun up a specified number of new pod replicas to replace original pods. By default, original pods with the original label are not deleted. In this case, if you run the get po command, you will find that the number of pods is doubled. The original pods are no longer controlled by the replication controller using the new label selector.

#### apply*

The apply command provides a more strict control on resource updating than patch and edit commands. The apply command applies a configuration to a resource and maintains a set of configuration files in source control. Whenever there is an update, the configuration file is pushed to the server, and then the kubectl apply command applies the latest configuration to the resource. The Kubernetes compares the new configuration file with the original one and updates only the changed configuration instead of the whole file. The configuration that is not contained in the -f flag will remain unchanged. Unlike the replace command which deletes the resource and creates a new one, the apply command directly updates the original resource. Similar to the git operation, the apply command adds an annotation to the resource to mark the current apply.

```
kubectl apply -f
```

#### patch

If you want to modify attributes of a running container without first deleting the container or using the replace command, the patch command is to the rescue. The patch command updates field(s) of a resource using strategic merge patch, a JSON merge patch, or a JSON patch. For example, to change a pod label from app=nginx1 to app=nginx2 while the pod is running, use the following command:

```
kubectl patch pod podname -p '{"metadata":{"labels":{"app":"nginx2"}}}'
```

#### convert*

The convert command converts configuration files between different API versions.
Configuration Commands

**label**
The *label* command update labels on a resource.

```
kubectl label pods my-pod new-label=newlabel
```

**annotate**
The *annotate* command update annotations on a resource.

```
kubectl annotate pods my-pod icon-url=http://......
```

**completion**
The *completion* command provides autocompletion for shell.

Other Commands

**api-versions**
The *api-versions* command prints the supported API versions.

```
kubectl api-versions
```

**api-resources**
The *api-resources* command prints the supported API resources.

```
kubectl api-resources
```

**config**
The *config* command modifies kubeconfig files. An example use case of this command is to configure authentication information in API calls.

**help**
The *help* command gets all command references.

**version**
The *version* command prints the client and server version information for the current context.

```
kubectl version
```

3.5.4 kubectl Usage Guide

Before running kubectl commands, you should have the kubectl development skills and understand the kubectl operations. For details, see Kubernetes API and kubectl CLI.

Go to the Kubernetes release page to download kubectl corresponding to the cluster version or a later version.

Cluster Connection

- Connecting to a Kubernetes cluster using kubectl
- Connecting to a cluster using the web-terminal add-on
CoreDNS HA

- Configuring high availability of kube-dns using kubectl

Workload Creation

- Creating a Deployment using kubectl
- Creating a StatefulSet using kubectl

Workload Affinity/Anti-affinity Scheduling

- Example YAML for workload-node affinity
- Example YAML for workload-node anti-affinity
- Example YAML for workload-workload affinity
- Example YAML for workload-workload anti-affinity
- Example YAML for workload-AZ affinity
- Example YAML for workload-AZ anti-affinity

Workload Access Mode Settings

- Implementing intra-cluster access using kubectl
- Implementing node access using kubectl
- Implementing Layer 4 load balancing using kubectl
- Implementing DNAT Gateway using kubectl
- Implementing Layer 7 load balancing using kubectl

Advanced Workload Settings

- Example YAML for setting the container lifecycle

Job Management

- Creating a job using kubectl
- Creating a cron job using kubectl

Configuration Center

- Creating a ConfigMap using kubectl
- Creating a secret using kubectl

Storage Management

- Creating an EVS disk using kubectl
- Attaching an EVS disk using kubectl
- Creating an OBS volume using kubectl
- Attaching an OBS volume using kubectl
- Creating an SFS volume using kubectl
- Attaching an SFS volume using kubectl
3.6 Setting Cluster Auto Scaling

Scenario

The Cluster Auto Scaling feature allows CCE to automatically scale out a cluster (adding worker nodes to a cluster) according to custom policies when workloads cannot be scheduled into the cluster due to insufficient cluster resources.

Notes and Constraints

- Among worker nodes, only pay-per-use VM nodes can be automatically added to clusters during cluster auto scaling.
- Currently, master nodes cannot be automatically added to or removed from clusters.
- If both auto scale-in and auto scale-out are required, use the autoscaler add-on. For details, see autoscaler.
- Clusters of v1.17 do not support auto scaling using AOM. You can use node pools for auto scaling. For details, see Node Pool Overview.

Automatic Cluster Scale-out

Step 1 Log in to the CCE console. Choose Resource Management > Clusters in the navigation pane. In the card view of the cluster to be scaled, choose More > Auto Scaling.

Step 2 Click the Scale-out Settings tab and then Edit. Set the maximum number of nodes, minimum number of nodes, cooldown period, and node configuration.

Figure 3-14 Cluster scale-out settings
Table 3-7: Scale-out settings

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cooldown Period</td>
<td>Interval between consecutive scale-out operations, in the unit of second. The cooldown period ensures that a scale-out operation is initiated only when previous scaling operation is finished and the system is running stably. The value ranges from 60 to 3600, in seconds. The default value is 900. If the cooling interval is less than 900 seconds (15 minutes), the auto scaling may not work well, because creating a node may take 2 to 10 minutes.</td>
</tr>
</tbody>
</table>
| Maximum Nodes      | Maximum number of nodes to which the cluster can scale out. 1 ≤ Maximum Nodes < cluster node quota  

**NOTE**  
The cluster node quota depends on the cluster size (maximum number of nodes that can be managed by a cluster) and the node quota of the account. The cluster node quota used here is the smaller of the two.                                                                                     |

| Node Configuration | If scale-out is required after the scale-out policy is executed, the system creates a node.  
1. Click **Set** and set the node parameters. For details about how to set the node parameters, see *Buying a Node*. Billing Mode can be set to **Pay-per-use** only.  
2. After the parameters are configured, click **Submit**. |

Step 3: After confirming the scale-out configuration and node parameters, click **OK**.

Step 4: Set the scale-out policy for the cluster. Click the **Scale-out Policies** tab and click **Add Policy**.
- **Policy Name**: Enter a policy name, for example, **policy01**.
- **Policy Type**: Currently, the following types of auto scale-out policies are supported:
  - **Metric-based policy**: Scale-out is performed based on the CPU or memory settings.

Table 3-8: Parameters for adding a metric-based policy

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>Metric</em></td>
<td>Select <strong>Allocated CPU</strong> or <strong>Allocated Memory</strong>.</td>
</tr>
<tr>
<td><em>Trigger Condition</em></td>
<td>Set a condition for triggering a scale-out policy, that is, when the average CPU or memory allocation value is greater than or less than a specified percentage.</td>
</tr>
</tbody>
</table>
### Parameter Description

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Monitoring Window</strong></td>
<td>Size of the data aggregation window. Select a value from the drop-down list. If you select <strong>15min</strong>, the selected metric is measured every 15 minutes.</td>
</tr>
<tr>
<td><strong>Threshold Crossings</strong></td>
<td>Number of consecutive times that the threshold is reached within the monitoring window. The calculation cycle is fixed at one minute. If you set this parameter to 3, the configured action will be triggered when the metrics meet the specified threshold for three consecutive times.</td>
</tr>
<tr>
<td><strong>Action</strong></td>
<td>Action executed after a policy is triggered.</td>
</tr>
</tbody>
</table>

- **Scheduled policy**: Scale-out is performed at a specified time.

**Table 3-9** Parameters for adding a scheduled policy

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Policy Type</strong></td>
<td>Set this parameter to <strong>Scheduled policy</strong>.</td>
</tr>
<tr>
<td><strong>Trigger Time</strong></td>
<td>Time at which the policy is triggered.</td>
</tr>
<tr>
<td><strong>Action</strong></td>
<td>Action executed after a policy is triggered.</td>
</tr>
</tbody>
</table>

- **Periodic policy**: Scale-out can be performed by day, week, or month.

**Table 3-10** Parameters for adding a periodic policy

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Policy Type</strong></td>
<td>Set the parameter to <strong>Periodic policy</strong>.</td>
</tr>
<tr>
<td><strong>Time Range</strong></td>
<td>Specify the time for triggering the policy.</td>
</tr>
<tr>
<td><strong>Action</strong></td>
<td>Action executed after a policy is triggered.</td>
</tr>
</tbody>
</table>

Step 5  Click OK.

After the auto scale-out is completed, choose **Resource Management > Nodes** in the navigation pane. On the node list, you can view the worker nodes added during cluster auto scaling.

----End

### 3.7 Upgrading a Cluster
3.7.1 Upgrade Overview

HUAWEI CLOUD CCE has passed the Certified Kubernetes Conformance Program and is a certified Kubernetes offering. To enable interoperability from one Kubernetes installation to the next, you must upgrade your Kubernetes clusters before the maintenance period ends.

After the latest Kubernetes version is available in CCE, CCE will describe the changes in this version. For details, see Kubernetes Release Notes.

You can use the CCE console to upgrade the Kubernetes version of a cluster.

An upgrade flag will be displayed on the cluster card view if there is a new version for the cluster to upgrade.

**How to check:**

Choose Resource Management > Clusters and check whether there is an upgrade flag in the upper right corner of the cluster card view. If yes, the cluster can be upgraded.

**Figure 3-15 Cluster with the upgrade flag**

![Cluster with the upgrade flag](image)

**Cluster Upgrade**

The following table describes the target version to which each cluster version can be upgraded, the supported upgrade modes, and upgrade impacts.
<table>
<thead>
<tr>
<th>Source Version</th>
<th>Target Version</th>
<th>Upgrade Modes</th>
<th>Impacts</th>
</tr>
</thead>
</table>
| v1.15          | v1.17.11       | Rolling upgrade | ● The FlexVolume (storage-driver) storage is no longer supported.  
                          ● Before upgrading a cluster, you need to switch from FlexVolume (storage-driver) to CSI (everest) for container storage. For details, see How Do I Change the Storage Class Used by a Cluster of v1.15 from FlexVolume to CSI Everest?. |
| v1.13          | v1.15.11       | Rolling upgrade | ● The proxy configuration item in the coredns add-on configuration is not supported and needs to be replaced with forward.  
                          ● The storage add-on is changed from storage-driver to everest. |
| v1.11          | 1.13.10        | Rolling upgrade | ● The cluster signature certificate mechanism is changed. As a result, the original cluster certificate becomes invalid. You need to obtain the certificate or kubeconfig file again after the cluster is upgraded.  
                          ● RBAC is enabled for clusters of Kubernetes v1.13 by default. Applications need to adapt to RBAC.  
                          ● After the cluster is upgraded from v1.9 to v1.13, kube-dns in the cluster will be replaced with CoreDNS. Before the upgrade, you need to back up the kube-dns configuration. After the upgrade, you need to reconfigure kube-dns in the coredns add-on. |
| v1.9.10        | Latest version that can be created on the console | Migration | You need to learn about the differences between versions. For details, see Cluster Upgrade Between Major Versions. |
| v1.9.7         |                |               |         |
| v1.9.2         |                |               |         |
| v1.7           |                |               |         |
Upgrade Modes

CCE provides the following upgrade modes based on the cluster version and deployment site. The upgrade processes are the same for master modes. The differences between these upgrade modes are described as follows:

Table 3-12 Differences between upgrade modes and their advantages and disadvantages

<table>
<thead>
<tr>
<th>Upgrade Mode</th>
<th>Method</th>
<th>Advantage</th>
<th>Disadvantage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rolling upgrade</td>
<td>Only the Kubernetes components and certain network components are upgraded on the node. The SchedulingDisabled label will be added to all existing nodes to ensure that the running applications are not affected. After the upgrade is complete, you need to manually create nodes and gradually release the old nodes, thereby migrating your applications to the new nodes. In this mode, you can control the upgrade process.</td>
<td>Services are not interrupted.</td>
<td>/</td>
</tr>
<tr>
<td>Replace upgrade</td>
<td>The latest worker node image is used to reset the node OS.</td>
<td>This is the fastest upgrade mode and requires few manual interventions.</td>
<td>Data or configurations on the node will be lost, and services will be interrupted for a period of time.</td>
</tr>
</tbody>
</table>
### Cluster Upgrade Between Major Versions

**Table 3-13** Instructions for upgrade between major versions

<table>
<thead>
<tr>
<th>Source Version</th>
<th>Target Version</th>
<th>Description</th>
</tr>
</thead>
</table>
| v1.17          | v1.19          | - Changelog from v1.17 to v1.19  
                 |                | Changelog from v1.18 to v1.19:  
                 |                | https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG/CHANGELOG-1.19.md  
                 |                | Changelog from v1.17 to v1.18:  
                 |                | https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG/CHANGELOG-1.18.md  
| v1.15          | v1.17          | You can upgrade clusters of v1.15 to v1.17 on the CCE console.  
                 |                | - Changelog from v1.15 to v1.17  
                 |                | Changelog from v1.16 to v1.17:  
                 |                | https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG/CHANGELOG-1.17.md  
                 |                | Changelog from v1.15 to v1.16:  
                 |                | https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG/CHANGELOG-1.16.md  
<pre><code>             |                | - For clusters created in CCE, Kubernetes v1.15.11 is a transitional version in which the FlexVolume plug-in (storage-driver) is compatible with the CSI plug-in (everest). Clusters of v1.17 and later versions do not support FlexVolume any more. You need to use the everest add-on. For details about CSI and FlexVolume, see Differences Between CSI and FlexVolume Plug-ins. |
</code></pre>
<table>
<thead>
<tr>
<th>Source Version</th>
<th>Target Version</th>
<th>Description</th>
</tr>
</thead>
</table>
| v1.13          | v1.15          | You can upgrade your clusters from v1.13 to v1.15 on the CCE console. For details, see [Upgrading a Cluster Using the Console](#). After a cluster is upgraded, it cannot be rolled back to the source version.  
  - Changelog from v1.13 to v1.15  
    Changelog from v1.14 to v1.15:  
    [https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG/CHANGELOG-1.15.md](https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG/CHANGELOG-1.15.md)  
    Changelog from v1.13 to v.1.14:  
  - After a cluster is upgraded from v1.13 to v1.15, the FlexVolume plug-in ([storage-driver](#)) is taken over by the CSI plug-in (everest v1.1.6 or later) for container storage. After the takeover, the original functions remain unchanged, however, you are advised not to create FlexVolume storage resources any more, which may not work properly in the cluster. For details about CSI and FlexVolume, see [Differences Between CSI and FlexVolume Plug-ins](#). |
| v1.11          | v1.13          | You can upgrade your clusters from v1.11 to v1.13 on the CCE console. For details, see [Upgrading a Cluster Using the Console](#). After a cluster is upgraded, it cannot be rolled back to the source version.  
  - Changelog from v1.11 to v1.13  
    Changelog from v1.12 to v1.13:  
    Changelog from v1.11 to v1.12:  
  - kube-dns of the cluster is replaced by core-dns. |
### Source Version | Target Version | Description
---|---|---
v1.9 | v1.11 | You can upgrade your clusters from v1.9 to v1.11 on the CCE console. For details, see [Upgrading a Cluster Using the Console](#).

After a cluster is upgraded, it cannot be rolled back to the source version.

- Changelog from v1.9 to v1.11
  - Changelog from v1.10 to v1.11:
    - [https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG/CHANGELOG-1.11.md](https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG/CHANGELOG-1.11.md)
  - Changelog from v1.9 to v.1.10:
    - [https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG/CHANGELOG-1.10.md](https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG/CHANGELOG-1.10.md)
- kube-dns of the cluster is replaced by core-dns.

v1.7 | v1.9 | You can upgrade your clusters from v1.7 to v1.9 on the CCE console. For details, see [Upgrading a Cluster Using the Console](#).

After a cluster is upgraded, it cannot be rolled back to the source version.

- Changelog from v1.7.3 to v1.9.7
  - Changelog from v1.8 to v1.9:
  - Changelog from v1.7 to v1.8:
- Clusters support GPUs.
- Auto scaling of cluster nodes is supported.
- Cluster upgrade from v1.7.3-r13 to 1.9.7-r1 (the network model of the source cluster version must be overlay12 container tunnel network).

### 3.7.2 Before You Start

Before the upgrade, you can check whether your cluster can be upgraded and which versions are available on the CCE console. For details, see [Upgrade Overview](#).
Precautions

- **Upgraded clusters cannot be rolled back. Therefore, perform the upgrade during off-peak hours to minimize the impact on your services.**
- Before upgrading a cluster, learn about the features and differences of each cluster version in Kubernetes Release Notes to avoid exceptions due to the use of an incompatible cluster version.
- Do not shut down or restart nodes during cluster upgrade. Otherwise, the upgrade fails.
- Before upgrading a cluster, disable auto scaling policies to prevent node scaling during the upgrade. Otherwise, the upgrade fails.
- If you locally modify the configuration of a cluster node, the cluster upgrade may fail or the configuration may be lost after the upgrade. Therefore, modify the configurations on the CCE console (cluster or node pool list page) so that they will be automatically inherited during the upgrade.
- During the cluster upgrade, the running workload services will not be interrupted, but access to the API server will be temporarily interrupted.
- Before upgrading the cluster, check whether the cluster is healthy. If the cluster is abnormal, you can try to rectify the fault. If the fault persists, submit a service ticket for assistance.
- To ensure data security, you are advised to back up data before upgrading the cluster. During the upgrade, you are not advised to perform any operations on the cluster.

Notes and Constraints

Currently, only CCE clusters can be upgraded. CCE clusters consisting of nodes created from private images and Kunpeng clusters cannot be upgraded.

Performing Pre-upgrade Check

Before upgrading a cluster, check the health status of the cluster and nodes and ensure that they are available.

**Method 1: Use the console.**

On the CCE console, click Resource Management in the navigation pane, and click Clusters and Nodes separately to check whether the cluster and nodes are normal.

**Method 2: Run kubectl commands.**

**Step 1** Connect to the cluster. For details, see Connecting to a Cluster Using kubectl.

**Step 2** Run the following command to verify that all cluster modules are in the Healthy state:

```shell
kubectl get cs
```

Information similar to the following is displayed:

<table>
<thead>
<tr>
<th>NAME</th>
<th>STATUS</th>
<th>MESSAGE</th>
<th>ERROR</th>
</tr>
</thead>
<tbody>
<tr>
<td>scheduler</td>
<td>Healthy</td>
<td>ok</td>
<td></td>
</tr>
<tr>
<td>controller-manager</td>
<td>Healthy</td>
<td>ok</td>
<td></td>
</tr>
<tr>
<td>etcd-0</td>
<td>Healthy</td>
<td>&quot;{&quot;health&quot;: &quot;true&quot;}&quot;</td>
<td></td>
</tr>
<tr>
<td>etcd-1</td>
<td>Healthy</td>
<td>&quot;{&quot;health&quot;: &quot;true&quot;}&quot;</td>
<td></td>
</tr>
<tr>
<td>etcd-2</td>
<td>Healthy</td>
<td>&quot;{&quot;health&quot;: &quot;true&quot;}&quot;</td>
<td></td>
</tr>
</tbody>
</table>
**NOTE**

In the command output, the value of **STATUS** must be **Healthy** for all items.

**Step 3** Run the following command to verify that all nodes are in the Ready state:

```bash
kubectl get nodes
```

**NOTE** All nodes must be in the **Ready** state.

<table>
<thead>
<tr>
<th>NAME</th>
<th>STATUS</th>
<th>ROLES</th>
<th>AGE</th>
<th>VERSION</th>
</tr>
</thead>
<tbody>
<tr>
<td>xxx.xxx.xx.xx</td>
<td>Ready</td>
<td>&lt;none&gt;</td>
<td>38d</td>
<td>v1.9.7-r1</td>
</tr>
<tr>
<td>xxx.xxx.xx.xx</td>
<td>Ready</td>
<td>&lt;none&gt;</td>
<td>38d</td>
<td>v1.9.7-r1</td>
</tr>
<tr>
<td>xxx.xxx.xx.xx</td>
<td>Ready</td>
<td>&lt;none&gt;</td>
<td>38d</td>
<td>v1.9.7-r1</td>
</tr>
</tbody>
</table>

----End

**Pre-upgrade Checklist**

Before upgrading a cluster, follow the pre-upgrade checklist to identify risks and problems in advance.

**Table 3-14 Cluster upgrade check items**

<table>
<thead>
<tr>
<th>Module</th>
<th>Item</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cluster</td>
<td>Check whether the node IP addresses (including EIPs) of the current cluster are used in other configurations or whitelists.</td>
</tr>
<tr>
<td></td>
<td>Perform the pre-upgrade check.</td>
</tr>
<tr>
<td>Workload</td>
<td>Record the number and status of workloads for comparison after the upgrade.</td>
</tr>
<tr>
<td></td>
<td>For the databases you use (such as Direct Connect, Redis, and MongoDB), you need to consider the changes in their whitelists, routes, or security group policies in advance.</td>
</tr>
<tr>
<td>Storage</td>
<td>Record the storage status to check whether storage resources are lost after the upgrade.</td>
</tr>
<tr>
<td>Networking</td>
<td>Check and back up the load balancing services and ingresses.</td>
</tr>
<tr>
<td></td>
<td>If Direct Connect is used, check whether the upgrade causes changes in the IP addresses of nodes or pods where services are deployed. To handle changes, you need to enable routes on Direct Connect in advance.</td>
</tr>
<tr>
<td>Add-on</td>
<td>When Kubernetes 1.9 is upgraded to 1.11, the kube-dns of the cluster is uninstalled and replaced with CoreDNS. Back up the DNS address configured in kube-dns so that you can use it in CoreDNS when the domain name resolution is abnormal.</td>
</tr>
</tbody>
</table>
### Private configurations:
Check whether data plane passwords, certificates, and environment variables are configured for nodes or containers in the cluster before the upgrade. If a container is restarted (for example, the node is abnormal and the pod is rescheduled), the configurations will be lost and your service will be abnormal.

Check and back up kernel parameters or system configurations.

### 3.7.3 Upgrading a Cluster Using the Console

**Scenario**

You can upgrade your clusters to a newer Kubernetes version on the CCE console.

Before the upgrade, learn about the target version to which each CCE cluster can be upgraded in what ways, and the upgrade impacts. For details, see [Upgrade Overview](#) and [Before You Start](#).

**Precautions**

- If the coredns add-on needs to be upgraded during the cluster upgrade, ensure that the number of nodes is greater than or equal to the number of coredns instances and all coredns instances are running. Otherwise, the upgrade will fail. Before upgrading a cluster of v1.11 or v1.13, you need to upgrade the coredns add-on to the latest version available for the cluster.

- When a cluster of v1.11 or earlier is upgraded to v1.13, the impacts on the cluster are as follows:
  - All cluster nodes will be restarted as their OSs are upgraded, which affects application running.
  - The cluster signature certificate mechanism is changed. As a result, the original cluster certificate becomes invalid. You need to obtain the certificate or kubecfg file again after the cluster is upgraded.

- During the upgrade from one release of v1.13 to a later release of v1.13, applications in the cluster are interrupted for a short period of time only during the upgrade of network components.

- During the upgrade from Kubernetes 1.9 to 1.11, the kube-dns of the cluster will be uninstalled and replaced with CoreDNS, which may cause loss of the cascading DNS configuration in the kube-dns or temporary interruption of the DNS service. Back up the DNS address configured in the kube-dns so you can configure the domain name in the CoreDNS again when domain name resolution is abnormal.

**Procedure**

**Step 1** Log in to the CCE console. In the navigation pane, choose Resource Management > Clusters. In the cluster list, check the cluster version.
Step 2 Click **More** for the cluster you want to upgrade, and select **Upgrade** from the drop-down menu.

**Figure 3-16 Upgrading a cluster**

![Upgrading a cluster](image)

**NOTE**

- If your cluster version is up-to-date, the **Upgrade** button is grayed out.
- If the cluster status is **Unavailable**, the upgrade flag in the upper right corner of the cluster card view will be grayed out. Check the cluster status by referring to **Before You Start**.

Step 3 In the displayed **Pre-upgrade Check** dialog box, click **Check Now**.

**Figure 3-17 Pre-upgrade check**

![Pre-upgrade check](image)

Step 4 The pre-upgrade check starts. While the pre-upgrade check is in progress, the cluster status will change to **Pre-checking** and new nodes/applications will not be able to be deployed on the cluster. However, existing nodes and applications will not be affected. It takes 3 to 5 minutes to complete the pre-upgrade check.
Figure 3-18 Pre-upgrade check in process

A set of checks will be performed before upgrading the cluster. While the pre-upgrade check is in progress, the cluster status will change to Pre-checking and new nodes/applications will not be able to be deployed on the cluster. However, existing nodes and applications will not be affected. It takes 3 to 5 minutes to complete the pre-upgrade check.

Step 5 When the status of the pre-upgrade check is Completed, click Upgrade.

Figure 3-19 Pre-upgrade check completed

Step 6 On the cluster upgrade page, review or configure basic information by referring to Table 3-15.
### Table 3-15 Basic information

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cluster Name</td>
<td>Review the name of the cluster to be upgraded.</td>
</tr>
<tr>
<td>Current Version</td>
<td>Review the version of the cluster to be upgraded.</td>
</tr>
<tr>
<td>Target Version</td>
<td>Review the target version after the upgrade.</td>
</tr>
<tr>
<td>Node Upgrade Policy</td>
<td><strong>Replace</strong> (replace upgrade): Worker nodes will be reset. Their OSs will be reinstalled, and data on the system and data disks will be cleared. Exercise caution when performing this operation.</td>
</tr>
<tr>
<td></td>
<td><strong>NOTE</strong></td>
</tr>
<tr>
<td></td>
<td>● The lifecycle management function of the nodes and workloads in the cluster is unavailable.</td>
</tr>
<tr>
<td></td>
<td>● APIs cannot be called temporarily.</td>
</tr>
<tr>
<td></td>
<td>● Running workloads will be interrupted because nodes are reset during the upgrade.</td>
</tr>
<tr>
<td></td>
<td>● Data in the system and data disks on the worker nodes will be cleared. Back up important data before resetting the nodes.</td>
</tr>
<tr>
<td></td>
<td>● Data disks without LVM mounted to worker nodes need to be mounted again after the upgrade, and data on the disks will not be lost during the upgrade.</td>
</tr>
<tr>
<td></td>
<td>● The EVS disk quota must be greater than 0.</td>
</tr>
<tr>
<td></td>
<td>● The container IP addresses change, but the communication between containers is not affected.</td>
</tr>
<tr>
<td></td>
<td>● Custom labels on the worker nodes will be cleared.</td>
</tr>
<tr>
<td></td>
<td>● It takes about 12 minutes to complete the cluster upgrade.</td>
</tr>
<tr>
<td>Login Mode</td>
<td><strong>Gray</strong> (rolling upgrade): Worker nodes are upgraded in rolling mode in a node pool. This mode applies to scenarios where all nodes in a cluster are created from a node pool.</td>
</tr>
<tr>
<td></td>
<td><strong>NOTE</strong></td>
</tr>
<tr>
<td></td>
<td>● The lifecycle management function of the nodes and workloads in the cluster is unavailable.</td>
</tr>
<tr>
<td></td>
<td>● APIs cannot be called temporarily.</td>
</tr>
<tr>
<td></td>
<td>● Running workloads are not interrupted.</td>
</tr>
<tr>
<td></td>
<td>● It takes about 12 minutes to complete the cluster upgrade.</td>
</tr>
<tr>
<td></td>
<td>You can use a password or key pair.</td>
</tr>
<tr>
<td></td>
<td>● If the login mode is <strong>Password</strong>: The default username is <strong>root</strong>. Enter the password for logging in to the node and confirm the password.</td>
</tr>
<tr>
<td></td>
<td>Remember the password as you will need it when you log in to the node.</td>
</tr>
<tr>
<td></td>
<td>● <strong>Key pair</strong>: Select the key pair used to log in to the node. You can select a shared key.</td>
</tr>
<tr>
<td></td>
<td>A key pair is used for identity authentication when you remotely log in to a node. If no key pair is available, click <strong>Create a key pair</strong>.</td>
</tr>
<tr>
<td></td>
<td>For details on how to create a key pair, see <a href="#">Creating a Key Pair</a>.</td>
</tr>
</tbody>
</table>
### Parameter Description

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node Upgrade Priority</td>
<td>You can select the nodes to be upgraded first.</td>
</tr>
</tbody>
</table>

#### Step 7
Click **Next**. In the dialog box displayed, click **OK**.

The message displayed varies depending on the node upgrade policy you selected.

- **Replace**: After the upgrade, the cluster uses OSs of a later version. During the upgrade, nodes are restarted and the OSs are upgraded, which interrupt services.
- **Gray**: You need to reset the nodes (and remove the labels that make the nodes unschedulable for pods) or create nodes to complete the rolling upgrade.

#### Step 8
Upgrade add-ons. If an add-on needs to be upgraded, a red dot is displayed. Click the **Upgrade** button in the lower left corner of the add-on card view. After the upgrade is complete, click **Upgrade** in the lower right corner of the page.

**NOTE**

- Master nodes will be upgraded first, and then the worker nodes will be upgraded concurrently. If there are a large number of worker nodes, they will be upgraded in different batches.
- Select a proper time window for the upgrade to reduce impacts on services.
- Clicking **OK** will start the upgrade immediately, and the upgrade cannot be canceled. Do not shut down or restart nodes during the upgrade.

#### Step 9
In the displayed **Upgrade** dialog box, read the information and click **OK**. Note that the cluster cannot be rolled back after the upgrade.

![Figure 3-20](image)

**Figure 3-20** Confirming cluster upgrade

#### Step 10
Back to the cluster list, you can see that the cluster status is **Upgrading**. Wait until the upgrade is completed.

After the upgrade is successful, you can view the cluster status and version on the cluster list or cluster details page.
3.7.4 Upgrading a Cluster (from v1.15 to v1.19)

Scenario

On the CCE console, you can upgrade clusters from v1.15 and v1.17 to v1.19 to use new features. Before the upgrade, learn about the target version to which each CCE cluster can be upgraded in what ways, and the upgrade impacts. For details, see Upgrade Overview and Before You Start.

Description

- An in-place upgrade updates the Kubernetes components on cluster nodes, without changing their OS version.
- Data plane nodes are upgraded in batches. By default, they are prioritized based on their CPU, memory, and PodDisruptionBudgets (PDBs). You can also set the priorities according to your service requirements.

Precautions

- During the cluster upgrade, the system will automatically upgrade add-ons to a version compatible with the target cluster version. Do not uninstall or reinstall add-ons during the cluster upgrade.
- Before the upgrade, ensure that all add-ons are running. If an add-on fails to be upgraded, rectify the fault and try again.
- During the upgrade, CCE checks the add-on running status. Some add-ons (such as coredns) require at least two nodes to run normally. In this case, at least two nodes must be available for the upgrade.
- If an upgrade failure message is displayed during the cluster upgrade, rectify the fault as prompted and try again. If upgrade attempts fail again, submit a service ticket for assistance.
Procedure

This section describes how to upgrade a CCE cluster from v1.15 and v1.17 to v1.19. For other versions, see Upgrading a Cluster Using the Console.

Step 1 Log in to the CCE console. In the navigation pane, choose Resource Management > Clusters. In the cluster list, view the cluster version. In this example, the cluster version is v1.17.17-r0.

Step 2 Click More for the cluster you want to upgrade, and select Upgrade from the drop-down menu.

Figure 3-22 Upgrading a cluster

- If your cluster version is up-to-date, the Upgrade button is grayed out.
- If the cluster status is Unavailable, the upgrade flag in the upper right corner of the cluster card view will be grayed out. Check the cluster status by referring to Before You Start.

Step 3 Check the version information, last update/upgrade time, available upgrade version, and upgrade history of the current cluster.

The cluster upgrade goes through pre-upgrade check, add-on upgrade/uninstallation, master node upgrade, worker node upgrade, and post-upgrade processing.

Figure 3-23 Cluster upgrade page
Step 4  Click **Upgrade** on the right. Set the upgrade parameters.

- **Available Versions**: Select v1.19 in this example.
- **Add-on Upgrade Configuration**: Add-ons that have been installed in your cluster are listed. During the cluster upgrade, the system automatically upgrades the add-ons to be compatible with the target cluster version. You can click **Set** to re-define the add-on parameters.

  **NOTE**

  If a red dot is displayed on the right of an add-on, the add-on is incompatible with the target cluster version. During the upgrade, the add-on will be uninstalled and then re-installed. Ensure that the add-on parameters are correctly configured.

- **Node Upgrade Configuration**: Before setting the node upgrade priority, you need to select a node pool. Nodes and node pools will be upgraded according to the priorities you specify. You can set the maximum number of nodes to be upgraded in a batch, or set priorities for nodes to be upgraded. If you do not set this parameter, the system will determine the nodes to upgrade in batches based on specific conditions.
  - **Add Upgrade Priority**: Add upgrade priorities for node pools.
  - **Add Node Priority**: After adding a node pool priority, you can set the upgrade sequence of nodes in the node pool. The system upgrades nodes in the sequence you specify. If you skip this setting, the system upgrades nodes based on the default policy.

Figure 3-24 Configuring upgrade parameters

---

Step 5  Read the upgrade instructions carefully, and select **I have read the upgrade instructions**. Click **Upgrade**.

Figure 3-25 Final step before upgrade

---

Step 6  After you click **Upgrade**, the cluster upgrade starts. You can view the upgrade process in the lower part of the page.
During the upgrade, you can click **Suspend** on the right to suspend the cluster upgrade. To continue the upgrade, click **Continue**.

**Figure 3-26** Cluster upgrade in process

Step 7 When the upgrade progress reaches 100%, the cluster is upgraded. The version information will be properly displayed, and no upgrade is required.

**Figure 3-27** Upgrade completed

Step 8 After the upgrade is complete, verify the cluster Kubernetes version on the **Clusters** page.
3.7.5 Migrating Services Across Clusters of Different Versions

Application Scenarios

This section describes how to migrate services from a cluster of an earlier version to a cluster of a later version in CCE.

This operation is applicable when a cross-version cluster upgrade is required (for example, upgrade from $v1.7.*$ or $v1.9.*$ to $1.17.*$) and new clusters can be created for service migration.

Prerequisites

Table 3-16 Checklist before migration

<table>
<thead>
<tr>
<th>Category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cluster</td>
<td>NodeIP-related: Check whether node IP addresses (including EIPs) of the cluster before the migration have been used in other configurations or whitelists.</td>
</tr>
<tr>
<td>Workloads</td>
<td>Record the number of workloads for post-migration check.</td>
</tr>
</tbody>
</table>
| Storage    | 1. Check whether the storage in use is provisioned by the public cloud or by the customer.  
                          2. Change the automatically created storage to the existing storage in the new cluster.                                               |
| Network    | 1. Pay special attention to the ELB and ingress.  
                          2. Clusters of an earlier version support only the classic load balancer. To migrate services to a new cluster, you need to change load balancer type to shared load balancer. Then, the corresponding ELB service will be re-established. |
### Procedure

**Step 1** Create a CCE cluster.

Create a cluster with the same specifications and configurations as the cluster of the earlier version. For details, see [Buying a CCE Cluster](#).

**Step 2** Add a node.

Add nodes with the same specifications and manual configuration items. For details, see [Buying a Node](#).

**Step 3** Create a storage volume in the new cluster.

Use an existing storage volume to create a PVC in the new cluster. The PVC name remains unchanged. For details, see [Creating a PV from an Existing EVS Disk](#).

**NOTE**

Storage switching supports only OBS buckets, SFS file systems, and shared EVS disks. If a non-shared EVS disk is used, you need to suspend the workloads in the old cluster to switch the storage resources. As a result, services will be interrupted.

**Step 4** Create a workload in the new cluster.

The workload name and specifications remain unchanged. For details about how to create a workload, see [Creating a Deployment](#) or [Creating a StatefulSet](#). For details about how to attach a storage volume to the workload, see [Creating a Pod Mounted with an EVS Volume](#).

**Step 5** Create a Service in the new cluster.

The Service name and specifications remain unchanged. For details about how to create a Service, see [Services](#).

**Step 6** Commission services.

After all resources are created, commission the containerized services. If the commissioning is successful, migrate the services to the new cluster.

**Step 7** Delete or unsubscribe from the old cluster.

After the new cluster is fully stable, unbind the associated storage volume. For details, see [Unbinding an EVS Disk](#). Then, unsubscribe from or delete the old cluster. For details about how to delete a cluster, see [Deleting a Cluster (Pay-per-Use)](#).

---End
3.7.6 CCE Kubernetes Release Notes

HUAWEI CLOUD has passed the Certified Kubernetes Conformance Program and CCE is a certified Kubernetes offering. To enable interoperability from one Kubernetes installation to the next, you must upgrade your Kubernetes clusters before the maintenance period ends.

After the latest Kubernetes version is released, CCE will provide you the changes in this version. For details, see Kubernetes Release Notes.

3.8 Managing a Cluster

3.8.1 Deleting a Cluster (Pay-per-Use)

Scenario

This section describes how to delete a cluster billed on a pay-per-use basis.

**NOTICE**

- Exercise caution when deleting a cluster because this operation will destroy the nodes in the cluster and the running services.
- You can unsubscribe from a yearly/monthly-billed cluster. Deletion without unsubscripton will not cut your bills. For details, see Deleting, Unsubscribing From, or Releasing a Yearly/Monthly-Billed Cluster.

Procedure

**Step 1** Log in to the CCE console. In the navigation pane, choose Resource Management > Clusters.

**Step 2** Choose More > Delete.

**Figure 3-29 Deleting a cluster**
Step 3  Delete the cluster.

**NOTE**

- Deleting a cluster will also delete nodes (except for nodes that are billed on a yearly/monthly basis and nodes accepted into the cluster), running workloads, and services in the cluster.
- The delete operation takes 1 to 3 minutes to complete.
- Enter `DELETE` into the text box below to confirm that the delete operation will continue.
- If a cluster whose status is Unavailable is deleted, some storage resources of the cluster may need to be manually deleted.
- If the cluster version is v1.13.10 or earlier, do not manually change the listener name and backend server name on the ELB console. Otherwise, residual resources will exist when you delete the cluster.

Figure 3-30 Deleting a cluster

![Delete Cluster](image)

**Step 4**  Click Yes to start deleting the cluster.

---End

3.8.2 Deleting, Unsubscribing From, or Releasing a Yearly/Monthly-Billed Cluster

You can delete, unsubscribe from, or release a yearly/monthly-billed cluster.
Precautions

- Unsubscribing from a cluster involves the renewed resources and the resources that are being used. After the unsubscription, these resources become unavailable.
- Cluster resources include master node resources and IaaS resources used by worker nodes. For details, see Pricing Details.
- If an order contains resources in a primary-secondary relationship, you need to unsubscribe from the resources separately.
- For details about unsubscription rules, see Unsubscription Allowed.

Unsubscribing From a Cluster

This section describes how to unsubscribe from a yearly/monthly-billed cluster that has not expired.

**NOTICE**

- Unsubscribing from a cluster will destroy all nodes and the running services in the cluster. Before the unsubscription, back up or migrate service data, which cannot be recovered after the unsubscription is complete as related cloud resources are deleted.
- You can directly delete pay-per-use clusters. For details, see Deleting a Cluster (Pay-per-Use).

Step 1 Log in to the CCE console. In the navigation pane, choose Resource Management > Clusters.

Step 2 Choose More > Unsubscribe in the card view of the cluster you want to unsubscribe from.

**Figure 3-31 Unsubscribing from a cluster**

Step 3 On the Unsubscribe from In-Use Resources page, unsubscribe from the resources as prompted.
● Unsubscribing from a cluster will destroy nodes (except those accepted into the cluster), running workloads, and services in the cluster.

● The unsubscribe takes 1 to 3 minutes to complete.

● For details about unsubscribe rules and handling fees, see Unsubscriptions.
  
  - If you are unsubscribing from a resource that is being used, review the resource information and refund information carefully. The resource cannot be restored after unsubscribe. You can unsubscribe from a renewal period that has not yet taken effect on renewed resources on the Unsubscribe from Renewal Period page.
  
  - Unsubscribing from a resource associated with other yearly/monthly-billed resources may affect the normal use of those resources.

  Unsubscribing from a resource associated with other pay-per-use resources will not affect the normal use of those resources. They will be billed normally.

  - If your operation is not a five-day unconditional unsubscription, you will be charged for the handling fee and the used amount. Used cash coupons and discount coupons will not be refunded.

Figure 3-32 Unsubscribing from cluster resources

Step 4 Select the check box "Data will be deleted and cannot be recovered. I confirm I have backed up or I no longer need the data". Click Confirm.

Step 5 In the dialog box displayed, read the instructions and review the information about the resources to be unsubscribed from. Click Yes to unsubscribe from the cluster.
Figure 3-33 Unsubscription review

![Unsubscription review](image)

**NOTE**

- Ensure that you have backed up or no longer need the data on the resources. Unsubscribed resources will be deleted and the data cannot be recovered.
- If you paid your order using a third-party payment platform, the refund will be added to your HUAWEI CLOUD cash account.

----End

**Releasing a Cluster**

This section describes how to release a *yearly/monthly-billed* cluster that has expired without renewal.

**NOTICE**

- Releasing a cluster will destroy all nodes and the running services in the cluster. Before the release, back up or migrate service data, which cannot be recovered after the release is complete as related cloud resources are deleted.
- You can directly delete pay-per-use clusters. For details, see [Deleting a Cluster (Pay-per-Use)](Deleting a Cluster (Pay-per-Use)).

**Step 1** Log in to the CCE console. In the navigation pane, choose Resource Management > Clusters.

**Step 2** Choose More > Release in the card view of the cluster to be released.

**Step 3** Release the cluster as prompted.
**3 Clusters**

### 3.8.3 Renewing a Yearly/Monthly-Billed Cluster

You can renew a yearly/monthly-billed cluster.

**Procedure**

This section describes how to renew a **yearly/monthly-billed** CCE cluster.

---

**NOTICE**

A yearly/monthly-billed cluster will be deleted if it is not renewed after expiration, and all nodes and the running services in the cluster will be destroyed. CCE strongly recommends that you renew the cluster before it expires or enable **auto renewal**.

---

**Step 1** Log in to the CCE console. In the navigation pane, choose **Resource Management > Clusters**.

**Step 2** Choose **More > Renew** in the card view of the cluster to be renewed.

**Step 3** On the displayed page, renew the service as prompted.

**NOTE**

- If the selected resource (highlighted) is associated with other resources, you can decide whether you want to perform the operation on all these resources at the same time.
- If you change resource specifications before the renewal period starts, you can unsubscribe from the resource, but not the renewal period.
- Renewed resources are not eligible of a 5-day unconditional unsubscription.
Step 4 Click Pay. On the page displayed, review the order amount, select a payment method, and click Pay.

Figure 3-36 Paying for the renewal

Step 5 After the payment is complete, you can go back to the Orders or Renewals page to view and manage your order.

End

3.8.4 Hibernating and Waking Up a Cluster (Pay-per-Use)

Scenario

If you do not need to use a cluster temporarily, you are advised to hibernate the cluster to save cluster management costs.
After a cluster is hibernated, resources such as workloads cannot be created or managed in the cluster.

A hibernated cluster can be quickly woken up and used normally.

Notes and Constraints

Yearly/Monthly-billed clusters cannot be hibernated.

Hibernating a Cluster

Step 1 Log in to the CCE console. In the navigation pane, choose Resource Management > Clusters.

Step 2 Choose More > Hibernate for the target cluster.

Step 3 In the dialog box displayed, check the precautions and click Yes. Wait until the cluster is hibernated.

Figure 3-37 Hibernating a cluster

![Hibernate]

Are you sure you want to hibernate the cluster dasdas?
1. When a cluster is in hibernation, resources such as workloads cannot be created or managed in it.
2. When a pay-per-use cluster is in hibernation, billing is suspended for resources on the master node. The other nodes, the bound elastic IP address, and bandwidth continue to be billed according to their respective billing modes.

[ ] Stop all the nodes in the cluster.

Yes No

**NOTICE**

- After a pay-per-use cluster is hibernated, the billing of master node resources will stop.
- After a cluster is hibernated, resources, such as worker nodes (ECSs), bound EIPs, and bandwidth, are still billed based on their own billing modes. To shut down nodes, select **Stop all nodes in the cluster** in the dialog box or see **Stopping a Node**.

Step 4 When the cluster status changes from Hibernating to Hibernation, the cluster is hibernated.
Waking Up a Cluster

Step 1 Log in to the CCE console. In the navigation pane, choose Resource Management > Clusters.

Step 2 Choose More > Wake.

Step 3 In the dialog box displayed, click Yes and wait until the cluster is woken up.

Figure 3-38 Waking up a cluster

⚠️ Wake

Are you sure you want to wake the cluster dasdas?
After the cluster wakes up, billing will resume for the resources on the master node.

Step 4 When the cluster status changes from Waking to Available, the cluster is woken up.

🚨 NOTE
After the cluster is woken up, billing will be resumed for the resources on the master node.

3.8.5 Configuring Kubernetes Parameters

Scenario
CCE clusters allow you to manage Kubernetes parameters, through which you can let core components work under your very requirements.

Notes and Constraints
This function is supported only in clusters of v1.15 and later. It is not displayed for versions earlier than v1.15.

Procedure

Step 1 Log in to the CCE console. In the navigation pane, choose Resource Management > Clusters.

Step 2 Choose More > Configuration.
Figure 3-39 Configuration

Step 3 On the Configuration page on the right, change the values of the following Kubernetes parameters:

<table>
<thead>
<tr>
<th>Component</th>
<th>Parameter</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>kube-apiserver</td>
<td>default-not-ready-toleration-seconds</td>
<td>notReady tolerance time, in seconds. NoExecute that is added by default to every pod that does not already have such a toleration.</td>
<td>Default: 300</td>
</tr>
<tr>
<td></td>
<td>default-unreachable-toleration-seconds</td>
<td>unreachable tolerance time, in seconds. NoExecute that is added by default to every pod that does not already have such a toleration.</td>
<td>Default: 300</td>
</tr>
</tbody>
</table>
|                   | max-mutating-requests-inflight                      | Maximum number of mutating requests in a specified period. When the value of this parameter is exceeded, the server rejects requests.  
The value 0 indicates no limitation.                                                                 | Default: 1000 |
|                   | max-requests-inflight                               | Maximum number of non-mutating requests in a specified period. When the value of this parameter is exceeded, the server rejects requests.  
The value 0 indicates no limitation.                                                                 | Default: 2000 |
<table>
<thead>
<tr>
<th>Component</th>
<th>Parameter</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>service-node-port-range</td>
<td>Range of node port numbers.</td>
<td>Default: 30000-32767 Options: Min ≥ 30000 Max ≤ 65535</td>
</tr>
<tr>
<td>kube-controller-manager</td>
<td>concurrent-deployment-syncs</td>
<td>Number of Deployments that are allowed to synchronize concurrently.</td>
<td>Default: 5</td>
</tr>
<tr>
<td></td>
<td>concurrent-endpoint-syncs</td>
<td>Number of endpoints that are allowed to synchronize concurrently.</td>
<td>Default: 5</td>
</tr>
<tr>
<td></td>
<td>concurrent-gc-syncs</td>
<td>Number of garbage collector workers that are allowed to synchronize concurrently.</td>
<td>Default: 20</td>
</tr>
<tr>
<td></td>
<td>concurrent-job-syncs</td>
<td>Number of jobs that can be synchronized at the same time.</td>
<td>Default: 5</td>
</tr>
<tr>
<td></td>
<td>concurrent-namespace-syncs</td>
<td>Number of namespaces that are allowed to synchronize concurrently.</td>
<td>Default: 10</td>
</tr>
<tr>
<td></td>
<td>concurrent-replicaset-syncs</td>
<td>Number of ReplicaSets that are allowed to synchronize concurrently.</td>
<td>Default: 5</td>
</tr>
<tr>
<td></td>
<td>concurrent-resource-quota-syncs</td>
<td>Number of resource quotas that are allowed to synchronize concurrently.</td>
<td>Default: 5</td>
</tr>
<tr>
<td></td>
<td>concurrent-service-syncs</td>
<td>Number of Services that are allowed to synchronize concurrently.</td>
<td>Default: 10</td>
</tr>
<tr>
<td></td>
<td>concurrent-serviceaccount-token-syncs</td>
<td>Number of service account tokens that are allowed to synchronize concurrently.</td>
<td>Default: 5</td>
</tr>
<tr>
<td></td>
<td>concurrent-ttl-after-finished-syncs</td>
<td>Number of TTL-after-finished controller workers that are allowed to synchronize concurrently.</td>
<td>Default: 5</td>
</tr>
</tbody>
</table>
### Table

<table>
<thead>
<tr>
<th>Component</th>
<th>Parameter</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>concurrent_rc_syncs</td>
<td>Number of replication controllers that are allowed to synchronize concurrently.</td>
<td>Default: 5</td>
</tr>
<tr>
<td></td>
<td>horizontal-pod-autoscaler-sync-period</td>
<td>How often HPA audits metrics in a cluster.</td>
<td>Default: 15 seconds</td>
</tr>
<tr>
<td></td>
<td>kube-api-qps</td>
<td>Query per second (QPS) to use while talking with kube-apiserver.</td>
<td>Default: 100</td>
</tr>
<tr>
<td></td>
<td>kube-api-burst</td>
<td>Burst to use while talking with kube-apiserver.</td>
<td>Default: 100</td>
</tr>
<tr>
<td>kube-scheduler</td>
<td>kube-api-qps</td>
<td>Query per second (QPS) to use while talking with kube-apiserver.</td>
<td>Default: 100</td>
</tr>
<tr>
<td></td>
<td>kube-api-burst</td>
<td>Burst to use while talking with kube-apiserver.</td>
<td>Default: 100</td>
</tr>
</tbody>
</table>

**Step 4** Click OK.

---End

### References

- kube-apiserver
- kube-controller-manager
- kube-scheduler

### 3.9 Obtaining a Cluster Certificate

#### Scenario

Before accessing cluster resources through open-source Kubernetes APIs, obtain the cluster's certificate.

#### Procedure

**Step 1** Log in to the CCE console. In the navigation pane, choose **Resource Management** > **Clusters**.

**Step 2** In the card view of the target cluster, choose **More** > **Download X.509 Certificate**.
Step 3  In the **Download X.509 Certificate** dialog box displayed, download the X.509 certificate of the cluster as prompted.

**Figure 3-41 Downloading a certificate**

- The downloaded certificate contains three files: **client.key**, **client.crt**, and **ca.crt**. Keep these files secure.
- Certificates are not required for mutual access between containers in a cluster.

---End

**3.10 Monitoring a Cluster**

**Scenario**

CCE is seamlessly integrated with the Cloud Eye service on HUAWEI CLOUD. You can learn the resource usage and monitoring metrics of clusters' master nodes in real time and receive alarms immediately upon detecting an anomaly.
Procedure

**Step 1** Log in to the CCE console. In the navigation pane, choose **Resource Management > Clusters**.

**Step 2** In the displayed cluster list page, click the name of the cluster to be monitored.

**Step 3** Click on the right of **More** on the **Cluster Details** page to view the cluster resource monitoring details.

**Figure 3-42 Cluster Details page**

![Cluster Details page](image)

Cluster monitoring page:

![Cluster monitoring page](image)

**Step 4** Click **More** on the right of the master node name on the **Cluster Details** page to view the basic monitoring data that the AOM console displays for the master node. The data includes CPU usage, memory usage, and disk usage.
3.11 Controlling Cluster Permissions

Scenario

This section describes how to control permissions on resources in a cluster, for example, allow user A to read and write application data in a namespace, and user B to only read resource data in a cluster.

Procedure

Step 1 If you need to perform permission control on the cluster, select Enhanced authentication for Authentication Mode during cluster creation, upload your own CA certificate, client certificate, and client certificate private key (for details about how to create a certificate, see Certificates), and select I have confirmed that the uploaded certificates are valid. For details, see Table 3-3.
**CAUTION**

- Upload a file **smaller than 1 MB**. The CA certificate and client certificate can be in `.crt` or `.cer` format. The private key of the client certificate can only be uploaded **unencrypted**.
- The validity period of the client certificate must be longer than five years.
- The uploaded CA certificate is used for both the authentication proxy and the kube-apiserver aggregation layer configuration. **If the certificate is invalid, the cluster cannot be created.**

**Step 2** Create a role using kubectl.

The following example shows how to create a role and allow the role to read all pods in the default namespace. For details about the parameters, see the [official Kubernetes documentation](https://kubernetes.io/docs/reference/access-config/role/).

```yaml
kind: Role
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  namespace: default
  name: pod-reader
rules:
  - apiGroups: [""
    resources: ["pods"
    verbs: ["get", "watch", "list"]
```

**Step 3** Bind the role to a user by using kubectl.

In the following example, the RoleBinding assigns the role of **pod-reader** in the default namespace to user **jane**. This policy allows user **jane** to read all pods in the default namespace. For details about the parameters, see the [official Kubernetes documentation](https://kubernetes.io/docs/reference/access-config/role/).

```yaml
kind: RoleBinding
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: read-pods
  namespace: default
subjects:
  - kind: User
    name: jane  #User name
    apiGroup: rbac.authorization.k8s.io
roleRef:
  kind: Role
  name: pod-reader  #Name of the role that is created
  apiGroup: rbac.authorization.k8s.io
```

**Step 4** After a role is created and bound to a user, call a Kubernetes API by initiating an API request message where headers carry user information and the certificate uploaded during cluster creation. For example, to call the pod query API, run the following command:

```
```

If **200** is returned, user **jane** is authorized to read pods in the cluster's default namespace. If **403** is returned, user **jane** is not authorized to read pods in the cluster's default namespace.
To prevent the command execution failure, upload the certificate to the `/root` directory in advance.

The parameter descriptions are as follows:

- **X-Remote-User: jane**: The request header is fixed at `X-Remote-User`, and `jane` is the username.
- **tls-ca.crt**: CA root certificate uploaded during cluster creation.
- **tls.crt**: client certificate that matches the CA root certificate uploaded during cluster creation.
- **tls.key**: client key corresponding to the CA root certificate uploaded during cluster creation.
- **192.168.23.5:5443**: address for connecting to the cluster. To obtain the address, perform the following steps:

  Log in to the CCE console. In the navigation pane, choose **Resource Management > Clusters**. Click the name of the cluster to be connected and obtain the IP address and port number from **Internal API Server Address** on the cluster details page.

![Figure 3-45 Obtaining the access address](image)

**NOTE**

- CCE allows a HUAWEI CLOUD account and its IAM users to download the config file (`kubeconfig.json`) separately. The config file downloaded by the IAM users is valid only for 30 days, whereas the one downloaded by the HUAWEI CLOUD account is valid permanently. This file is used for user cluster authentication. If the file is leaked, your clusters may be attacked. When the file is leaked, you can replace the authentication credential by updating the certificate.
- The Kubernetes permissions assigned by the config file downloaded by the IAM users are the same as those assigned to the IAM users on the CCE console.
In addition, the **X-Remote-Group** header field, that is, the user group name, is supported. During role binding, a role can be bound to a group and carry user group information when you access the cluster.

----End
4.1 Node Overview

Introduction

A container cluster consists of a set of worker machines, called nodes, that run containerized applications. A node can be a virtual machine (VM) or a physical machine (PM), depending on your service requirements. The components on a node include kubelet, container runtime, and kube-proxy.

NOTE

A Kubernetes cluster consists of master nodes and node nodes. The nodes described in this section refer to worker nodes, the computing nodes of a cluster that run containerized applications.

CCE uses high-performance Elastic Cloud Servers (ECSs) or Bare Metal Servers (BMSs) as nodes to build highly available Kubernetes clusters.

Notes

- To ensure node stability, a certain amount of CCE node resources will be reserved for Kubernetes components (such as kubelet, kube-proxy, and docker) based on the node specifications. Therefore, the total number of node resources and the amount of allocatable node resources for your cluster are different. The larger the node specifications, the more the containers deployed on the node. Therefore, more node resources need to be reserved to run Kubernetes components.

- The node networking (such as the VM networking and container networking) is taken over by CCE. You are not allowed to add NICs or change routes. If you modify the networking configuration, the availability of CCE may be affected.

- If you want to modify the specifications of a purchased node, stop the node and perform the operations described in General Operations for Modifying Specifications. You can also purchase a new node and delete the old one.
Node Lifecycle

A lifecycle indicates the node statuses recorded from the time when the node is created through the time when the node is deleted or released.

<table>
<thead>
<tr>
<th>Status</th>
<th>Status Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Available</td>
<td>Stable state</td>
<td>The node is running properly and is connected to the cluster. Nodes in this state can provide services.</td>
</tr>
<tr>
<td>Unavailable</td>
<td>Stable state</td>
<td>The node is not running properly. A node in this state cannot provide services. Contact the administrator or perform the operations described in Resetting a Node.</td>
</tr>
<tr>
<td>Creating</td>
<td>Intermediate state</td>
<td>The node has been created but is not running.</td>
</tr>
<tr>
<td>Installing</td>
<td>Intermediate state</td>
<td>The Kubernetes software is being installed on the node.</td>
</tr>
<tr>
<td>Deleting</td>
<td>Intermediate state</td>
<td>The node is being deleted. If node stays stably in the Deleting state, an exception occurs. In this case, contact the administrator to handle the exception.</td>
</tr>
<tr>
<td>Stopped</td>
<td>Stable state</td>
<td>The node is stopped properly. A node in this state cannot provide services. You can start the node on the ECS console.</td>
</tr>
<tr>
<td>Error</td>
<td>Stable state</td>
<td>The node is abnormal. A node in this state cannot provide services. Contact the administrator or perform the operations described in Resetting a Node.</td>
</tr>
</tbody>
</table>

Mapping between Node OSs and Container Engines

<table>
<thead>
<tr>
<th>OS</th>
<th>Kernel Version</th>
<th>Container Engine</th>
<th>Container Storage Rootfs</th>
</tr>
</thead>
<tbody>
<tr>
<td>CentOS 7.x</td>
<td>3.x</td>
<td>Docker</td>
<td>Devicemapper</td>
</tr>
<tr>
<td>EulerOS 2.5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EulerOS 2.3</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table 4-3 Node OSs and container engines in CCE Kunpeng clusters

<table>
<thead>
<tr>
<th>OS</th>
<th>Kernel Version</th>
<th>Container Engine</th>
<th>Container Storage Rootfs</th>
</tr>
</thead>
<tbody>
<tr>
<td>EulerOS 2.8</td>
<td>4.x</td>
<td>Docker</td>
<td>Overlayfs</td>
</tr>
<tr>
<td>Ubuntu 18.04</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

4.2 Buying a Node

Scenario
A node is a virtual or physical machine that provides computing resources. Sufficient nodes must be available in your project to ensure that operations, such as creating workloads, can be performed.

Prerequisites

- At least one cluster is available. For details on how to create a cluster, see Buying a CCE Cluster.
- A key pair has been created. The key pair will be used for identity authentication upon remote node login.
  If you use a password to log in to a node, skip this step. For details, see Creating a Key Pair.

Notes and Constraints

- During the node creation, software packages are downloaded from OBS using the domain name. Therefore, you need to use a private DNS server to resolve the OBS domain name. Otherwise, the node fails to be created. Therefore, the subnet where the node resides must be configured with a private DNS server address so that the purchased node can use the private DNS server. When you create a subnet, the private DNS server is used by default. If you have changed the subnet DNS, ensure that the DNS server in the subnet can resolve the OBS domain name. Otherwise, you need to use the private DNS server.
- Only KVM nodes can be created. Non-KVM nodes cannot be used after being created.
- Once a node is purchased, its AZ cannot be changed.
- Nodes purchased in the pay-per-use billing mode will be deleted after you delete them on the Resource Management > Nodes page on the CCE console. Yearly/monthly-billed nodes in a cluster cannot be deleted on the
CCE console. You can choose Billing Center > My Orders in the upper right corner of the page to unsubscribe from the nodes.

- CCE supports GPUs through an add-on named gpu-beta. You need to install this add-on to use GPU-enabled nodes in your cluster.
- If the cluster network model is container tunnel network, the cluster cannot contain VM nodes, and the cluster version must be v1.13.10 or later. (Clusters using this network model can manage only one type of nodes.)
- If the network model is VPC network, the cluster version must be v1.11.7 or later. (Clusters using this network model can manage VM nodes and BMS nodes at the same time).

Procedure

**Step 1** Log in to the CCE console. Use either of the following methods to add a node:

- In the navigation pane, choose Resource Management > Nodes. Select the cluster to which the node will belong and click Buy Node on the upper part of the node list page.

**Figure 4-1** Method 1 for opening the Buy Node page

- In the navigation pane, choose Resource Management > Clusters. In the card view of the cluster to which you will add nodes, click Buy Node.

**Figure 4-2** Method 2 for opening the Buy Node page
Step 2  Set Billing mode to Pay-per-use or Yearly/Monthly. This section uses the pay-per-use billing mode as an example.

Step 3  Select a region and an AZ.
- **Current Region**: geographic location of the nodes to be created.
- **AZ**: Set this parameter based on the site requirements. An AZ is a physical region where resources use independent power supply and networks. AZs are physically isolated but interconnected through an internal network.

You are advised to deploy worker nodes in different AZs after the cluster is created to make your workloads more reliable. When creating a cluster, you can deploy nodes only in one AZ.

**Figure 4-3** Worker nodes in different AZs

Step 4  Configure node parameters.
- **Node Type**
  - **VM node**: A VM node will be created in the cluster.
  - **Bare-metal node**: Nodes running on bare metal servers (called BMS nodes) cannot be created along with a cluster. You can add BMS nodes to a cluster only after the cluster is created.

**NOTE**

BMS nodes can be created only after the CCE cluster is created and meets the following conditions:

- All IP addresses are not IPv6.
- The cluster version is v1.11.7 or later (with VPC network used), or v1.13.10 or later (with tunnel network used).
- BMS nodes in the cluster are billed on a yearly/monthly basis.

For details on how to buy BMS nodes, see **Buying a Node**.

- **Node Name**: Enter a node name. A node name contains 1 to 56 characters starting with a lowercase letter and not ending with a hyphen (-). Only lowercase letters, digits, and hyphens (-) are allowed.

If you change the node name on the ECS console after the node is created, be sure to synchronize the new node name from ECS to CCE. For details, see **Synchronizing Node Data**.

- **Specifications**: Select node specifications that best fit your business needs.
  - **General-purpose**: provides a balance of computing, memory, and network resources. It is a good choice for many applications, such as web servers, workload development, workload testing, and small-scale databases.
- **Memory-optimized**: provides higher memory capacity than general-purpose nodes and is suitable for relational databases, NoSQL, and other workloads that are both memory-intensive and data-intensive.

- **General computing-basic**: provides a balance of computing, memory, and network resources and uses the vCPU credit mechanism to ensure baseline computing performance. Nodes of this type are suitable for applications requiring burstable high performance, such as light-load web servers, enterprise R&D and testing environments, and low- and medium-performance databases.

- **GPU-accelerated**: provides powerful floating-point computing and is suitable for real-time, highly concurrent massive computing. Graphical processing units (GPUs) of P series are suitable for deep learning, scientific computing, and CAE. GPUs of G series are suitable for 3D animation rendering and CAD. **GPU-accelerated nodes can be added only to clusters of v1.11 or later.**

- **High-performance computing**: provides stable and ultra-high computing performance and is suitable for scientific computing and workloads that demand ultra-high computing power and throughput.

- **General computing-plus**: provides stable performance and exclusive resources to enterprise-class workloads with high and stable computing performance.

- **Disk-intensive**: supports local disk storage and provides high network performance. It is designed for workloads requiring high throughput and data switching, such as big data workloads.

- **Ultra-high I/O**: delivers ultra-low SSD access latency and ultra-high IOPS performance. This type of specifications is ideal for high-performance relational databases, NoSQL databases (such as Cassandra and MongoDB), and Elasticsearch.

**Figure 4-4 Selecting node specifications**

To ensure node stability, CCE automatically reserves some resources to run necessary system components. For details, see [Formula for Calculating the Reserved Resources of a Node].

- **OS**: Select an OS for the node to be created. In certain regions, only OSs are displayed and options **Public image** and **Private image** are unavailable.
  - **Public image**: Select an OS for the node.
A public image is a standard, widely used image. It contains an OS and preinstalled public applications and is available to all users. For more information, see Overview.

- **Private image** (OBT): A private image contains an OS or service data, preinstalled public applications, and the owner’s private applications. It is available only to the user who created it. **Private images are supported only for clusters of v1.15 or later.**

  If no private image is available, create one by following the instructions provided in **Using a Private Image to Build a Worker Node Image (OBT).**

- **Shared image**: A shared image is a private image shared by another user. For details, see **Overview of Sharing Images.**

Reinstalling the OS or modifying OS configurations could make the node unavailable. Exercise caution when performing these operations. For details, see **High-Risk Operations and Solutions.**

- **System Disk**: Set the system disk space of the worker node. The value ranges from 40GB to 1024 GB. The default value is 40GB.

  By default, system disks support High I/O (SAS) and Ultra-high I/O (SSD) EVS disks. For details, see **EVS Disk Overview.**

  **Encryption**: Data disk encryption safeguards your data. Snapshots generated from encrypted disks and disks created using these snapshots automatically inherit the encryption function. **This function is available only in certain regions.**

  - **Encryption** is not selected by default.

  - After you select **Encryption**, you can select an existing key in the displayed **Encryption Setting** dialog box. If no key is available, click the link next to the drop-down box to create a key. After the key is created, click the refresh icon.

- **Data Disk**: Set the data disk space of the worker node. The value ranges from 100 GB to 32,768 GB. The default value is 100 GB. The EVS disk types provided for the data disk are the same as those for the system disk.

  - **LVM**: If this option is selected, CCE data disks are managed by the Logical Volume Manager (LVM). On this condition, you can adjust the disk space allocation for different resources. This option is selected for the first disk by default and cannot be unselected. You can choose to enable or disable LVM for new data disks.

    ▪ This option is selected by default, indicating that LVM management is enabled.

    ▪ You can deselect the check box to disable LVM management.

---

**CAUTION**

If the data disk is uninstalled or damaged, the Docker service becomes abnormal and the node becomes unavailable. You are advised not to delete the data disk.
CAUTION

○ Disk space of the data disks managed by LVM will be allocated according to the ratio you set.

○ When creating a node in a cluster of v1.13.10 or later, if LVM is not selected for a data disk, follow instructions in Adding a Second Data Disk to a Node in a CCE Cluster to fill in the pre-installation script and format the data disk. Otherwise, the data disk will still be managed by LVM.

○ When creating a node in a cluster earlier than v1.13.10, you must format the data disks that are not managed by LVM. Otherwise, either these data disks or the first data disk will be managed by LVM.

- Encryption: Data disk encryption provides powerful protection for your data. Snapshots generated from encrypted disks and disks created using these snapshots automatically inherit the encryption function. This function is supported only for clusters of v1.13.10 or later in certain regions, and is not displayed for clusters of v1.13.10 or earlier.
  ▪ Encryption is not selected by default.
  ▪ After you select Encryption, you can select an existing key in the displayed Encryption Setting dialog box. If no key is available, click the link next to the drop-down box to create a key. After the key is created, click the refresh icon.

- Add Data Disk: Currently, a maximum of two data disks can be attached to a node. After the node is created, you can go to the ECS console to attach more data disks. This function is available only to clusters of certain versions.

- Data disk space allocation: Click Change Configuration to specify the resource ratio for Kubernetes Space and User Space. Disk space of the data disks managed by LVM will be allocated according to the ratio you set. This function is available only to clusters of certain versions.
  ▪ Kubernetes Space: You can specify the ratio of the data disk space for storing Docker and kubelet resources. Docker resources include the Docker working directory, Docker images, and image metadata. kubelet resources include pod configuration files, secrets, and emptyDirs.
    The Docker space size is determined by your service requirements. For details, see Docker Disk Space.
  ▪ User Space: You can set the ratio of the disk space that is not allocated to Kubernetes resources and the path to which the user space is mounted.
**NOTE**

Note that the mount path cannot be /, /home/paas, /var/paas, /var/lib, /var/script, /var/log, /mnt/paas, or /opt/cloud, and cannot conflict with the system directories (such as bin, lib, home, root, boot, dev, etc, lost+found, mnt, proc, sbin, srv, tmp, var, media, opt, selinux, sys, and usr). Otherwise, the system or node installation will fail.

If the cluster version is v1.13.10-r0 or later and the node type is Disk-intensive or Ultra-high I/O, the following options are displayed for data disks:

- **EVS**: Parameters are the same as those when the node type is not Disk-intensive or Ultra-high I/O. For details, see Data Disk above.
- **Local disk**: Local disks may break down and do not ensure data reliability. It is recommended that you store service data in EVS disks, which are more reliable than local disks.

Local disk parameters are as follows:

- **Disk Mode**: If the node type is disk-intensive, the supported disk mode is HDD. If the node type is ultra-high I/O, the supported disk mode is SSD.

- **Read/Write Mode**: When multiple local disks exist, you can set the read/write mode. The serial and sequential modes are supported. **Sequential** indicates that data is read and written in linear mode. When a disk is used up, the next disk is used. **Serial** indicates that data is read and written in striping mode, allowing multiple local disks to be read and written at the same time.

- **Kubernetes Space**: You can specify the ratio of the data disk space for storing Docker and kubelet resources. Docker resources include the Docker working directory, Docker images, and image metadata. kubelet resources include pod configuration files, secrets, and emptyDirs.

- **User Space**: You can set the ratio of the disk space that is not allocated to Kubernetes resources and the path to which the user space is mounted.

**NOTICE**

- The ratio of disk space allocated to the Kubernetes space and user space must be equal to 100% in total. You can click ⚡ to refresh the data after you have modified the ratio.

- By default, disks run in the direct-lvm mode. If data disks are removed, the loop-lvm mode will be used and this will impair system stability.
**Figure 4-5 Setting a local disk**

- **VPC**: A VPC where the current cluster is located. This parameter cannot be changed and is displayed only for clusters of v1.13.10-r0 or later.

- **Subnet**: A subnet improves network security by providing exclusive network resources that are isolated from other networks. You can select any subnet in the cluster VPC. Cluster nodes can belong to different subnets.

During the node creation, software packages are downloaded from OBS using the domain name. Therefore, you need to use a private DNS server to resolve the OBS domain name. Otherwise, the node fails to be created. Therefore, the subnet where the node resides must be configured with a **private DNS server address** so that the purchased node can use the private DNS server. When you create a subnet, the private DNS server is used by default. If you have changed the subnet DNS, ensure that the DNS server in the subnet can resolve the OBS domain name. Otherwise, you need to use the private DNS server.

When a node is added to an existing cluster, if an extended CIDR block is added to the VPC corresponding to the subnet and the subnet is an extended CIDR block, you need to add the following three security group rules to the master node security group (the group name is in the format of **Cluster name-cc-e-control-Random number**). These rules ensure that the nodes added to the cluster are available. (This step is not required if an extended CIDR block has been added to the VPC during cluster creation.)
Step 5  **EIP**: an independent public IP address. If the nodes to be created require public network access, select **Automatically assign** or **Use existing**. **This parameter is not displayed when IPv6 is enabled for the cluster.**

An EIP bound to the node allows public network access. EIP bandwidth can be modified at any time. An ECS without a bound EIP cannot access the Internet or be accessed by public networks. For details, see [EIP Overview](#).

- **Do not use**: A node without an EIP cannot be accessed from public networks. It can be used only as a cloud server for deploying services or clusters on a private network.
- **Automatically assign**: An EIP with specified configurations is automatically assigned to each node. If the number of EIPs is less than the number of nodes, the EIPs are randomly bound to the nodes. Configure the EIP specifications, billing factor, bandwidth type, and bandwidth size as required. When creating an ECS, ensure that the elastic IP address quota is sufficient.
- **Use existing**: Existing EIPs are assigned to the nodes to be created.

**NOTE**

By default, VPC's SNAT feature is disabled for CCE. If SNAT is enabled, you do not need to use EIPs to access public networks. For details about SNAT, see [Custom Policies](#).

Step 6  **Shared Bandwidth**: Select **Do not use** or **Use existing**. **This parameter is displayed only when IPv6 is enabled for the cluster.**

An EIP bound to the node allows public network access. EIP bandwidth can be modified at any time. An ECS without a bound EIP cannot access the Internet or be accessed by public networks.

Step 7  **Login Mode**: You can use a password or key pair.

- **Password**: The default username is `root`. Enter the password for logging in to the node and confirm the password.
  
  Be sure to remember the password as you will need it when you log in to the node.

- **Key pair**: Select the key pair used to log in to the node. You can select a shared key.

  A key pair is used for identity authentication when you remotely log in to a node. If no key pair is available, click **Create a key pair**. For details on how to create a key pair, see [Creating a Key Pair](#).

**NOTICE**

When creating a node using a key pair, IAM users can select only the key pairs created by their own, regardless of whether these users are in the same group. For example, user B cannot use the key pair created by user A to create a node, and the key pair is not displayed in the drop-down list on the CCE console.
Step 8  Advanced ECS Settings (optional): Click \(^\wedge\) to show advanced ECS settings.

- **ECS Group**: An ECS group logically groups ECSs. The ECSs in the same ECS group comply with the same policy associated with the ECS group.
  - **Anti-affinity**: ECSs in an ECS group are deployed on different physical hosts to improve service reliability.
  - **Fault domain**: ECSs in an ECS group are deployed in multiple failure domains so that a failure in one failure domain will not affect the ECSs in other failure domains, thereby improving service reliability. This option is displayed only when the environment supports failure domains. This option is not supported if a worker node is deployed in a random AZ.

Select an existing ECS group, or click **Create ECS Group** to create a new one. After the ECS group is created, click the refresh button.

- **Resource Tags**: By adding tags to resources, you can classify resources.

  You can create predefined tags in Tag Management Service (TMS). Predefined tags are visible to all service resources that support the tagging function. You can use predefined tags to improve tag creation and migration efficiency.

  CCE will automatically create the "CCE-Dynamic-Provisioning-Node=node id" tag. A maximum of 5 tags can be added.

- **Agency**: An agency is created by a tenant administrator on the IAM console. By creating an agency, you can share your cloud server resources with another account, or entrust a more professional person or team to manage your resources. For details on how to create an agency, see **Cloud Service Delegation**. To authorize an ECS or BMS to call cloud services, select **Cloud service** as the agency type, click **Select**, and then select **ECS BMS**.

- **Pre-installation Script**: Enter a maximum of 1,000 characters.

  The script will be executed before Kubernetes software is installed. Note that if the script is incorrect, Kubernetes software may not be installed successfully. The script is usually used to format data disks.

- **Post-installation Script**: Enter a maximum of 1,000 characters.

  The script will be executed after Kubernetes software is installed and will not affect the installation. The script is usually used to modify Docker parameters.

- **Subnet IP Address**: Select **Automatically assign IP address** (recommended) or **Manually assigning IP addresses**.

Step 9  Advanced Kubernetes Settings: (Optional) Click \(^\wedge\) to show advanced ECS settings.

- **Max Pods**: maximum number of pods that can be created on a node, including the system's default pods. If the cluster uses the VPC network model, the maximum value is determined by the number of IP addresses that can be allocated to containers on each node.

  This limit prevents the node from being overloaded by managing too many pods.
• **Maximum Data Space per Container**: maximum data space that can be used by a container. The value ranges from 10 GB to 80 GB. If the value of this field is larger than the data disk space allocated to Docker resources, the latter will override the value specified here. Typically, 90% of the data disk space is allocated to Docker resources. This parameter is displayed only for clusters of v1.13.10-r0 and later.

**Step 10 Nodes**: The value cannot exceed the management scale you select when configuring cluster parameters. Set this parameter based on service requirements and the remaining quota displayed on the page. Click ![ ] to view the factors that affect the number of nodes to be added (depending on the factor with the minimum value). To apply for more quotas, click **Increase quota**.

**Step 11 Validity Duration**: Set this parameter if you select the **yearly/monthly** billing mode.

**Step 12** Click **Next: Confirm**. After confirming that the configuration is correct, click **Submit**.

If the billing mode is **Yearly/Monthly**, click **Pay Now** after confirming the configuration, and pay as prompted.

The node list page is displayed. If the node status is **Available**, the node is added successfully. It takes about 6 to 10 minutes to create a node.

---

**NOTE**

- If the console indicates insufficient EIP quota during node creation, increase the quota by following the instructions provided in **How Do I Troubleshoot Insufficient EIPs When a Node Is Added?**
- A cloud server is automatically created during node creation. If the cloud server fails to be created, a rollback process starts and is charged according to the pricing principles of cloud servers. If there is a rollback fee, go to **Billing Center** to unsubscribe from the cloud server.
- Do not delete the security groups and related rules automatically configured during cluster creation. Otherwise, the cluster will exhibit unexpected behavior.

**Step 13** Click **Back to Node List**. The node has been created successfully if it changes to the **Available** state.

**Figure 4-7 Node list**
The allocatable resources are calculated based on the resource request value (Request), which indicates the upper limit of resources that can be requested by pods on this node, but does not indicate the actual available resources of the node.

The calculation formula is as follows:

- Allocatable CPUs = Total CPUs – Requested CPUs of all pods – Reserved CPUs for other resources
- Allocatable memory = Total memory – Requested memory of all pods – Reserved memory for other resources

### 4.3 Accepting ECSs/BMSs as Nodes into a Cluster

**Scenario**

In CCE, you can add a new node or add existing nodes (ECSs/BMS) into your cluster. These nodes can be billed in yearly/monthly or pay-per-use mode.

This section describes how to accept existing ECSs as nodes into a cluster on the CCE console.

**NOTICE**

- While an ECS is being accepted into a cluster, the operating system of the ECS will be reset to the standard OS image provided by CCE to ensure node stability. The CCE console prompts you to select the operating system and the login mode during the reset.
- The system disk and data disk of an ECS will be formatted while the ECS is being accepted into a cluster. Ensure that information in the disks has been backed up.
- While an ECS is being accepted into a cluster, do not perform any operation on the ECS through the ECS console.

**Notes and Constraints**

- The cluster version must be v1.13 or later.
- Heterogeneous nodes, such as Kunpeng, Ascend-accelerated, and HECS nodes, cannot be accepted and managed.
- If IPv6 is enabled for a cluster, only nodes in a subnet with IPv6 enabled can be accepted and managed. If IPv6 is not enabled for the cluster, only nodes in a subnet without IPv6 enabled can be accepted.
- If the password or key has been set when a VM node is created, the VM node can be accepted into a cluster 10 minutes after it is available.

**Prerequisites**

An ECS that meets the following conditions can be accepted:
- The node has been purchased and is in the running state, and is not used by other clusters.
- The node to be accepted and the CCE cluster must be in the same VPC. (If the cluster version is earlier than v1.13.10, the node to be accepted and the CCE cluster must be in the same subnet.)
- Only one data disk is attached to the node. The data disk capacity is greater than or equal to 100 GB. If the node has more than one data disk, detach other data disks first.
- The node has 2-core or higher CPU, 4 GB or larger memory, and only one NIC.

Procedure

Step 1 Log in to the CCE console. In the navigation pane, choose Resource Management > Clusters. Click More for the target cluster, and select Add to Cluster.

Step 2 (Alternative) Log in to the CCE console. In the navigation pane, choose Resource Management > Nodes. On the page displayed, select the cluster to which the node belongs, and click Add to Cluster in the upper right corner.

Step 3 View the ECSs/BMSs that meet the requirements. If the nodes cannot be added, a red icon is displayed and the number of unqualified nodes is provided. You can move the cursor to view the cause.

Step 4 Select the ECS to be accepted into the cluster, and click Next.

Step 5 On the Node Configuration page, select a node login mode, set Advanced Settings to On, set Max Pods and Disk Allocation, and click Create Now.

**Figure 4-8 Node settings**

Step 6 In the dialog box displayed, confirm the information and click OK.
Figure 4-9 Confirming node information

![Confirm](image)

Verify that the following information is correct:
- Cluster Name: 
- Node to Be Added: 
- OS after added: EulerOS 2.5
- Login Mode: Password

A message is displayed, indicating that the request for accepting the existing node into the selected cluster is successfully submitted.

---End

4.4 Removing a Node

Scenario

Removing a node from a cluster in CCE will re-install the node OS and clear CCE components on the node.

Removing a node will not delete the server (ECS or BMS) corresponding to the node. You are advised to remove nodes at off-peak hours to avoid impacts on your services.

Notes and Constraints

- Nodes can be removed only when the cluster is in the Available or Unavailable state.
- A CCE node can be removed only when it is in the Active, Abnormal, or Error state.
- A CCE node in the Active state can have its OS re-installed and CCE components cleared after it is removed.
- If the OS fails to be re-installed after the node is removed, manually re-install the OS. After the re-installation, log in to the node and run the clearance script to clear CCE components. For details, see Handling Failed OS Reinstallation.

Precautions

- Removing a node will lead to pod migration, which may affect services. Perform this operation during off-peak hours.
- Unexpected risks may occur during the operation. Back up data in advance.
- While the node is being deleted, the backend will set the node to the unschedulable state.
After you remove the node and re-install the OS, the original LVM partitions will be cleared and the data managed by LVM will be cleared. Therefore, back up data in advance.

**Procedure**

**Step 1** Log in to the CCE console. In the navigation pane, choose **Resource Management > Nodes**. In the same row as the target node, choose **More > Remove**.

![Figure 4-10 Removing a node](image)

You can also select multiple nodes and remove them at a time.

![Figure 4-11 Removing multiple nodes at a time](image)

**Step 2** In the dialog box displayed, enter **REMOVE**, configure the login information required for re-installing the OS, and click **Yes**. Wait until the node is removed.

After the node is removed, workload pods on the node are automatically migrated to other available nodes.

### Handling Failed OS Reinstallation

You can perform the following steps to re-install the OS and clear the CCE components on the node if previous attempts fail:

**Step 1** Log in to the management console of the server and re-install the OS. For details, see **Changing the OS**.

**Step 2** Log in to the server and run the following commands to clear the CCE components and LVM data:

Write the following scripts to the **clean.sh** file:

```
lsblk
vgs --noheadings | awk '{print $1}' | xargs vgremove -f
```
Run the following command:

```
bash clean.sh
```

#### 4.5 Logging In to a Node

**Notes and Constraints**

- If you use SSH to log in to a node (an ECS), ensure that the ECS already has an EIP (a public IP address).
- Only login to a running ECS is allowed.
- Only the user root can log in to a Linux server.

**Login Modes**

You can log in to an ECS in either of the following modes:

- **Management console (VNC)**
  
  If an ECS has no EIP, log in to the ECS console and click **Remote Login** in the same row as the ECS.
  
  For details, see **Login Using VNC**.

- **SSH**
  
  This mode applies only to ECSs running Linux. Usually, you can use a remote login tool, such as PuTTY, Xshell, and SecureCRT, to log in to your ECS. If none of the remote login tools can be used, log in to the ECS console and click **Remote Login** in the same row as the ECS to view the connection status and running status of the ECS.
NOTE

- You can use either an SSH key or SSH password for login. For details, see Login Using an SSH Key and Login Using an SSH Password.
- When you use the Windows OS to log in to a Linux node, set Auto-login username to root.
- The CCE console does not support node OS upgrade. Do not upgrade the node OS using the yum update command. Otherwise, the container networking components will be unavailable. For details on how to manually restore the container network, see What Can I Do If the Container Network Becomes Unavailable After yum update Is Used to Upgrade the OS?

Table 4-4 Linux ECS login modes

<table>
<thead>
<tr>
<th>EIP Binding</th>
<th>On-Premises OS</th>
<th>Connection Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes</td>
<td>Windows</td>
<td>Use a remote login tool, such as PuTTY or XShell.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SSH password authentication: Login Using an SSH Password</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SSH key authentication: Login Using an SSH Key</td>
</tr>
<tr>
<td>Yes</td>
<td>Linux</td>
<td>Run commands.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SSH password authentication: Login Using an SSH Password</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SSH key authentication: Login Using an SSH Key</td>
</tr>
<tr>
<td>Yes/No</td>
<td>Windows/ Linux</td>
<td>Remote login using the management console: Login Using VNC</td>
</tr>
</tbody>
</table>

4.6 Monitoring a Node

Scenario

CCE is seamlessly integrated with Cloud Eye, which allows you to view the resource usage of each worker node in a cluster in real time. You can use Cloud Eye to monitor your nodes in real time. Automatic alarm reporting and notification help you better understand the performance of your nodes.

Cloud Eye is a multi-dimensional monitoring platform for resources such as ECSs and bandwidth. With Cloud Eye, you can view the resource usage and service running status in the cloud, and respond to exceptions in a timely manner to ensure smooth running of services.

HUWEI CLOUD Cloud Eye can monitor node running. You can obtain the monitoring metrics of each node on the CCE console or the Cloud Eye console.

Monitored data requires a period of time for transmission and display. The node status displayed on the Cloud Eye page is the status obtained 5 to 10 minutes
before. You can view the monitored data of a newly created node after 5 to 10 minutes.

**Procedure**

**Step 1** Log in to the CCE console. In the navigation pane, choose Resource Management > Nodes.

**Step 2** Select a cluster in the upper right corner of the Nodes page.

**Step 3** In the same row of the node to be monitored, click Monitoring.

**Figure 4-12 Monitoring a node**

**Step 4** In the node monitoring window displayed, you can view the basic monitoring metrics, including the CPU/memory/disk usage and upstream/downstream bandwidth at different time points.

**Figure 4-13 Metrics**

**Table 4-5 Major monitoring metrics**

<table>
<thead>
<tr>
<th>Metric</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU Usage</td>
<td>Measures the CPU usage of the physical server accommodating the monitored ECS, which is not accurate as that obtained on the monitored ECS.</td>
</tr>
<tr>
<td>Memory Usage</td>
<td>Memory usage of the node.</td>
</tr>
<tr>
<td>Metric</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Disk Usage</td>
<td>Disk usage of the node. The monitored object is the disk with the highest disk usage and the data does not represent the average usage of all disks. The remaining disk capacity displayed is that of the node with the largest remaining disk capacity, not the sum of the remaining capacities of all disks.</td>
</tr>
<tr>
<td>CPU/Memory/Disk Usage Monitoring</td>
<td>You can view the curves of the CPU usage, memory usage, and disk usage at different time points within one hour.</td>
</tr>
<tr>
<td>Downstream Rate</td>
<td>Speed at which data is downloaded to the node. The unit is KB/s.</td>
</tr>
<tr>
<td>Upstream Rate</td>
<td>Speed at which data is uploaded from the node. The unit is KB/s.</td>
</tr>
<tr>
<td>Network Monitoring</td>
<td>You can view the curves of the downstream rate and upstream rate at different time points within one hour.</td>
</tr>
</tbody>
</table>

----End

**Related Operations**

- Viewing Server Monitoring Metrics
- Monitoring Metrics
- Setting Alarm Rules

**4.7 Managing Node Labels**

You can add different labels to nodes and define different attributes for labels. By using these node labels, you can quickly understand the characteristics of each node.

**Node Label Usage Scenario**

Node labels are mainly used in the following scenarios:

- Node management: Node labels are used to classify nodes.
- Affinity and anti-affinity between a workload and node:
  - Some workloads require a large CPU, some require a large memory, some require a large I/O, and other workloads may be affected. In this case, you are advised to add different labels to nodes. When deploying a workload, you can select nodes with specified labels for affinity deployment to ensure the normal operation of the system. Otherwise, node anti-affinity deployment can be used.
  - A system can be divided into multiple modules. Each module consists of multiple microservices. To ensure the efficiency of subsequent O&M, you can add a module label to each node so that each module can be
deployed on the corresponding node, does not interfere with other modules, and can be easily developed and maintained on its node.

Inherent Label of a Node

After a node is created, some fixed labels exist and cannot be deleted. For details about these labels, see Table 4-6.

Table 4-6 Inherent label of a node

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>New: topology.kubernetes.io/region</td>
<td>Indicates the region where the node is located. ap-southeast-3 indicates the AP-Singapore region.</td>
</tr>
<tr>
<td>Old: failure-domain.beta.kubernetes.io/region</td>
<td></td>
</tr>
<tr>
<td>New: node.kubernetes.io/baremetal</td>
<td>Indicates whether the node is a bare metal node. false indicates that the node is not a bare metal node.</td>
</tr>
<tr>
<td>Old: failure-domain.beta.kubernetes.io/is-baremetal</td>
<td></td>
</tr>
<tr>
<td>node.kubernetes.io/container-engine</td>
<td>Indicates the container engine. Example: docker or containerd</td>
</tr>
<tr>
<td>topology.kubernetes.io/zone</td>
<td>Indicates the AZ where the node is located. ap-southeast-3a indicates the AP-Singapore1 region.</td>
</tr>
<tr>
<td>node.kubernetes.io/subnetid</td>
<td>Indicates a subnet ID.</td>
</tr>
<tr>
<td>os.architecture</td>
<td>Indicates the processor architecture of a node. For example, amd64 indicates a AMD64-bit processor.</td>
</tr>
<tr>
<td>os.name</td>
<td>Indicates the operating system name of a node. For example, EulerOS_2.0_SP5 indicates that EulerOS 2.5 is used.</td>
</tr>
<tr>
<td>os.version</td>
<td>Indicates the kernel version of a node. For example, 3.10.0-327.62.59.83.h96.x86_64.</td>
</tr>
</tbody>
</table>

Adding a Node Label

Step 1 Log in to the CCE console. In the navigation pane, choose Resource Management > Nodes.
Step 2 In the same row as the node for which you will add labels, choose **Operation > More > Manage Labels**.

Step 3 In the dialog box displayed, click **Add Label** below the label list, enter the key and value of the label to be added, and click **OK**.

As shown in the figure, the key is **deploy_qa** and the value is **true**, indicating that the node is used to deploy the QA (test) environment.

**Figure 4-14 Adding a label**

![Manage Labels dialog box](image)

Step 4 After the label is added, click **Manage Labels**. Then, you will see the label that you have added.

**Figure 4-15 Label added successfully**

![Label added successfully](image)

---End

**Deleting a Node Label**

Only labels added by users can be deleted. Labels that are fixed on the node cannot be deleted.
Step 1 Log in to the CCE console. In the navigation pane, choose Resource Management > Nodes.

Step 2 In the same row as the node for which you will delete labels, choose Operation > More > Manage Labels.

Step 3 Click Delete, and then click OK to delete the label. Label updated successfully is displayed.

Figure 4-16 Label updated successfully

Searching for a Node by Label

Step 1 Log in to the CCE console. In the navigation pane, choose Resource Management > Nodes.

Step 2 In the upper right corner of the node list, click Search by Label.

Step 3 Enter a Kubernetes label to find the target node.

4.8 Synchronizing Node Data

Scenario

Each node in a cluster is a cloud server or physical machine. After a cluster node is created, you can change the cloud server name or specifications as required.

Some information about CCE nodes is maintained independently from the ECS console. After you change the name, EIP, billing mode, or specifications of an ECS on the ECS console, you need to synchronize the ECS information to the corresponding node on the CCE console. After the synchronization, information on both consoles is consistent.

Common ECS information to be modified:

- ECS (node) name: Changing an ECS Name
- Billing mode: Changing Pay-per-Use to Yearly/Monthly and Changing Yearly/Monthly to Pay-per-Use
- Node specifications: General Operations for Modifying Specifications
Procedure

Step 1  Log in to the CCE console. In the navigation pane, choose Resource Management > Nodes.

Step 2  In the same row as the node whose data will be synchronized, choose More > Sync Node data.

NOTE

Alternatively, click the node name, and click Sync Node Data in the upper right corner of the node details page.

Figure 4-17 Synchronizing node data

After the synchronization is complete, the "Sync success" message is displayed in the upper right corner.

Figure 4-18 Successful data synchronization

4.9 Resetting a Node

Scenario

When a node in a CCE cluster is reset, services running on the node will also be deleted. Exercise caution when performing this operation.

Notes and Constraints

- For CCE clusters, the version must be v1.13 or later to support node resetting.
- For Kunpeng clusters, the version must be v1.15 or later to support node resetting.

Notes

- Only worker nodes can be reset. If the node is still unavailable after the resetting, delete the node and purchase a new one.
- Resetting a node will reinstall the node OS and interrupt workload services running on the node. Therefore, perform this operation during off-peak hours.
- Data in the system disk and Docker data disks will be cleared. Back up important data before resetting the node.
- When an extra data disk is mounted to a node, data in this disk will be cleared if the disk has not been unmounted before the node reset. To prevent data loss, back up data in advance and mount the data disk again after the node reset is complete.
- The IP addresses of the workload pods on the node will change, but the container network access is not affected.
- There is remaining EVS disk quota.
- While the node is being deleted, the backend will set the node to the unschedulable state.

Procedure

**Step 1** Log in to the CCE console. In the navigation pane, choose **Resource Management** > **Nodes**. In the same row as the node you will reset, choose **More > Reset**.

*Figure 4-19* Resetting a node

**Step 2** In the dialog box displayed, enter **RESET** and reconfigure the password or key pair for login.
Figure 4-20 Node resetting dialog box

Step 3 Click Yes and wait until the node is reset.

After the node is reset, pods on it are automatically migrated to other available nodes.

---End

4.10 Deleting a Node

Scenario

When a node in a CCE cluster is deleted, services running on the node will also be deleted. Exercise caution when performing this operation.

Notes and Constraints

- After a CCE cluster is deleted, the ECS nodes in the cluster are also deleted.

  - NOTICE

For clusters of v1.17.11 or later, after a VM is unsubscribed from or deleted on the ECS console, the corresponding node in the CCE cluster is automatically deleted.
Notes

- Deleting a node will lead to pod migration, which may affect services. Perform this operation during off-peak hours.
- Unexpected risks may occur during the operation. Back up related data in advance.
- During the operation, the backend will set the node to the unschedulable state.
- Only worker nodes can be deleted.

Procedure

**Step 1** Log in to the CCE console. In the navigation pane, choose Resource Management > Nodes. In the same row as the node you will delete, choose More > Delete.

![Deleting a node](image)

**Step 2** In the Delete Node dialog box, enter DELETE and click Yes.

![Confirming the deletion](image)
### 4.11 Stopping a Node

**Scenario**

After a node in the cluster is stopped, services on the node are also stopped. Before stopping a node, ensure that discontinuity of the services on the node will not result in adverse impacts.

After a node is stopped, it is no longer billed.

**Notes and Constraints**

- Deleting a node will lead to pod migration, which may affect services. Therefore, delete nodes during off-peak hours.
- Unexpected risks may occur during node deletion. Back up related data in advance.
- While the node is being deleted, the backend will set the node to the unschedulable state.
- Only worker nodes can be stopped.

**Procedure**

**Step 1** Log in to the CCE console. In the navigation pane, choose Resource Management > Nodes.

**Step 2** In the node list, click the name of the node to be stopped.

**Step 3** On the node details page displayed, click the node name.
Step 4 In the upper right corner of the ECS details page, click **Stop**. In the **Stop ECS** dialog box, click **Yes**.

---End

### 4.12 Performing Rolling Upgrade for Nodes

**Scenario**

You can migrate workloads to a node of a new version without deleting the original node. **Figure 4-26** shows the migration process.
**Figure 4-26 Workload migration**

![Flowchart for workload migration](image)

**Notes and Constraints**
- The original node and the target node to which the workload is to be migrated must be in the same cluster.
- The cluster must be of v1.13.10 or later.
- The default node pool DefaultPool does not support this configuration.

**Scenario 1: The Original Node Is in DefaultPool**

**Step 1** Create a node.
1. Log in to the CCE console. In the navigation pane, choose **Resource Management > Node Pools**.
2. Select the cluster to which the original node belongs.
3. Click **Create Node Pool**, set the following parameters, and modify other parameters as required. For details about the parameters, see **Creating a Node Pool**.
   a. **Name**: Enter the name of the new node pool, for example, `nodepool-demo`.
   b. **Nodes**: In this example, add one node.
c. **Specifications**: Select node specifications that best suit your needs.

d. **OS**: Select the operating system (OS) of the nodes to be created.

e. **Login Mode**: You can use a password or key pair.

- If the login mode is **Password**: The default username is **root**. Enter the password for logging in to the node and confirm the password. Remember the password as you will need it when you log in to the node.

- If the login mode is **Key pair**, select a key pair for logging in to the node and select the check box to acknowledge that you have obtained the key file and that without this file you will not be able to log in to the node.

  A key pair is used for identity authentication when you remotely log in to a node. If no key pair is available, click **Create a key pair**. For details on how to create a key pair, see *[Creating a Key Pair]*.

4. Click **Next: Confirm**. Confirm the node pool configuration and click **Submit**. Go back to the node pool list. In the node list, you can view that the new node pool has been created and is in the Normal state.

**Step 2** Click the name of the node pool. The IP address of the new node is displayed in the node list.

**Step 3** Install and configure kubectl.

1. In the navigation pane of the CCE console, choose **Resource Management > Clusters**, and click **Command Line Tool > Kubectl** under the cluster where the original node is located.

2. On the **Kubectl** tab page of the cluster details page, connect to the cluster as prompted.

**Step 4** Migrate the workload.

1. Add a taint to the node where the workload needs to be migrated out.

   ```
kubectl taint node [node] key=value:[effect]
   ```

   In the preceding command, `[node]` indicates the IP address of the node where the workload to be migrated is located. The value of `[effect]` can be **NoSchedule**, **PreferNoSchedule**, or **NoExecute**. In this example, set this parameter to **NoSchedule**.

   - **NoSchedule**: Pods that do not tolerate this taint are not scheduled on the node; existing pods are not evicted from the node.

   - **PreferNoSchedule**: Kubernetes tries to avoid scheduling pods that do not tolerate this taint onto the node.

   - **NoExecute**: A pod is evicted from the node if it is already running on the node, and is not scheduled onto the node if it is not yet running on the node.

   **NOTE**

   To reset a taint, run the **kubectl taint node [node] key:[effect]-** command to remove the taint.

2. Safely evicts the workload on the node.
kubectl drain [node]
In the preceding command, [node] indicates the IP address of the node where
the workload to be migrated is located.

3. In the navigation pane of the CCE console, choose Workloads >
Deployments. In the workload list, the status of the workload to be migrated
changes from Running to Unready. If the workload status changes to
Running again, the migration is successful.

**NOTE**
During workload migration, if node affinity is configured for the workload, the workload
keeps displaying a message indicating that the workload is not ready. In this case, click the
workload name to go to the workload details page. On the Scheduling Policies tab page,
delete the affinity configuration of the original node and click Add Simple Scheduling
Policy to configure the affinity and anti-affinity policies of the new node. For details, see
Simple Scheduling Policies.

After the workload is successfully migrated, you can view that the workload is
migrated to the node created in Step 1 on the Pods tab page of the workload
details page.

**Step 5** Delete the original node.

After the workload is successfully migrated and is running properly, choose
Resource Management > Nodes to delete the original node.

---End

**Scenario 2: The Original Node Is Not in DefaultPool**

**Step 1** Copy the node pool and add nodes to it.

1. Log in to the CCE console. In the navigation pane, choose Resource
Management > Node Pools.
2. Select the cluster to which the original node belongs.
   In the node pool list, locate the node pool to which the original node belongs.
3. Click More > Copy next to the node pool name. On the Create Node Pool
   page, set the following parameters and modify other parameters as required.
   For details about the parameters, see Creating a Node Pool.
   - Name: Enter the name of the new node pool, for example, nodepool-
demo.
   - Nodes: In this example, add one node.
   - Specifications: Select node specifications that best suit your needs.
   - OS: Select the operating system (OS) of the nodes to be created.
   - Login Mode: You can use a password or key pair.
     - If the login mode is Password: The default username is root. Enter
       the password for logging in to the node and confirm the password.
       Remember the password as you will need it when you log in to the
       node.
     - If the login mode is Key pair, select a key pair for logging in to the
       node and select the check box to acknowledge that you have
obtained the key file and that without this file you will not be able to log in to the node.

A key pair is used for identity authentication when you remotely log in to a node. If no key pair is available, click **Create a key pair**. For details on how to create a key pair, see **Creating a Key Pair**.

4. Click **Next: Confirm**. Confirm the node pool configuration and click **Submit**.

   Go back to the node pool list. In the node list, you can view that the new node pool has been created and is in the Normal state.

**Step 2** Click the name of the node pool. The IP address of the new node is displayed in the node list.

**Step 3** Migrate the workload.

1. Click **Edit** on the right of nodepool-demo and set **Taints**.
2. Click **Add Taint**, set **Key** and **Value**, and set **Effect** to **NoExecute**. The value options of **Effect** include **NoSchedule**, **PreferNoSchedule**, or **NoExecute**.
   - **NoSchedule**: Pods that do not tolerate this taint are not scheduled on the node; existing pods are not evicted from the node.
   - **PreferNoSchedule**: Kubernetes tries to avoid scheduling pods that do not tolerate this taint onto the node.
   - **NoExecute**: A pod is evicted from the node if it is already running on the node, and is not scheduled onto the node if it is not yet running on the node.

   **NOTE**

   If you need to reset the taint, enter the new values or click **Delete**.

3. Click **Save**.

4. In the navigation pane of the CCE console, choose **Workloads > Deployments**. In the workload list, the status of the workload to be migrated changes from **Running** to **Unready**. If the workload status changes to **Running** again, the migration is successful.

   **NOTE**

   During workload migration, if node affinity is configured for the workload, the workload keeps displaying a message indicating that the workload is not ready. In this case, click the workload name to go to the workload details page. On the **Scheduling Policies** tab page, delete the affinity configuration of the original node and click **Add Simple Scheduling Policy** to configure the affinity and anti-affinity policies of the new node. For details, see **Simple Scheduling Policies**.

   After the workload is successfully migrated, you can view that the workload is migrated to the node created in **Step 1** on the **Pods** tab page of the workload details page.

**Step 4** Delete the original node.

After the workload is successfully migrated and is running properly, choose **Resource Management > Node Pools** to delete the original node.

----End
4.13 Upgrading the Node OS Kernel

Scenario

CCE clusters depend on the system kernel version. When you upgrade a cluster, the OSs of the nodes in the cluster will be automatically upgraded by default. Manually upgrading the OS kernel of cluster nodes can be risky, and you are not advised to perform this operation.

It is strongly recommended that you back up data on the node before the upgrade. If a node/cluster becomes unavailable or the network becomes abnormal after the upgrade, you can reset the node. For details, see Resetting a Node.

Notes

- CentOS 7.6 VMs can be upgraded to 3.10.0-1127.18.2.el7.x86_64. BMS nodes can be upgraded to 3.10.0-514.44.5.10.h275.x86_64.
- EulerOS 2.2 supports an upgrade to the kernel version 3.10.0-327.62.59.83.h162.x86_64.
- EulerOS 2.5 supports an upgrade to the kernel version 3.10.0-862.14.1.0.h197.eulerovs2r7.x86_64.
- The node vulnerable to Linux Kernel SACK vulnerabilities must have an EIP, and must be restarted after the kernel is upgraded.
- The following errors may be encountered during the upgrade. However, they do not affect system functions and can be ignored.

![Error messages](image)

CentOS Upgrade

CentOS Linux kernels earlier than 3.10.0-1127.el7 can be upgraded in clusters of v1.15.6-r1, v1.15.11-r1, and v1.17.9-r0.

NOTICE

- If the gpu-beta add-on is installed in the cluster, ensure that the add-on has been upgraded to v1.1.13 or later before installing the kernel upgrade patch. For details, see gpu-beta.
- The kernel cannot be rolled back after upgrade.
- Before upgrading the node kernel, back up key data on the node.
- During the kernel upgrade, the node needs to be restarted. You are advised to evict the workloads on the node before the upgrade.

For existing nodes
Step 1 Log in to the worker node as user root.

Step 2 Run the following command to install the patch:

```
cd /root; curl http://{obs_package_bucket}/cluster-versions/CCE-Kernel-Patch-1127.18.2.1.tgz -1 -O; tar -zxf CCE-Kernel-Patch-1127.18.2.1.tgz; bash update.sh; script_path=`cat /etc/crontab | grep 'run/update.sh' | awk -F " " '{print $8}'; if [ $script_path != "x" ];then sed -i '/grub2-mkconfig/d' $script_path; fi
```

`obs_package_bucket` is the address resolved from the domain name of the OBS bucket.

Step 3 Restart the OS after the patch is installed.

----End

For newly created nodes

Step 1 Log in to the CCE console. In the navigation pane, choose Resource Management > Clusters. In the card view of the cluster to which you will add nodes, click Buy Node.

Step 2 Enter the following command in the Post-installation Script text box:

```
cd /root; curl http://{obs_package_bucket}/cluster-versions/CCE-Kernel-Patch-1127.18.2.1.tgz -1 -O; tar -zxf CCE-Kernel-Patch-1127.18.2.1.tgz; bash update.sh; script_path=`cat /etc/crontab | grep 'run/update.sh' | awk -F " " '{print $8}'; if [ $script_path != "x" ];then sed -i '/grub2-mkconfig/d' $script_path; fi
```

`obs_package_bucket` is the address resolved from the domain name of the OBS bucket.

Step 3 After the node is created, log in to the node and restart the OS for the kernel patch to take effect. Before restarting the node, you are advised to evict the workloads on the node to avoid service interruption.

----End

For existing BMS nodes

Step 1 Log in to the node as user root and ensure that the current BMS kernel version is h275.

Run the following command on the node:

```
uname -r
```

```
[root@instance-c1414ocyv-ovs -]# uname -r
3.10.0-514.44.5.10.h275.x86_64
```

Step 2 Run the following command to install the patch:

```
mkdir -p /root/upgrade_ovs; cd /root/upgrade_ovs; wget https://{obs_package_bucket}/package/canal-agent/canal-agent-20.6.0.B005.sp1.tgz; tar zxf canal-agent-20.6.0.B005.sp1.tgz; tar zxf canal-agent/package/openvswitch-20.6.0.B003-x86_64.tar.gz; bash openswitch/can_ovs.sh; bash openswitch/can_ovs.sh uninstall; bash openswitch/can_ovs.sh install;
```

`obs_package_bucket` is the address resolved from the domain name of the OBS bucket.

Step 3 After the command is executed, run the modinfo openvswitch command to check whether the current Open vSwitch version is 3.10.0-514.44.5.10.h142.x86_64.
EulerOS Upgrade

**Step 1** Connect to the master node and evict the pods on the affected worker node vulnerable to Linux Kernel SACK vulnerabilities. The pods will then be rebuilt on other nodes to meet the replica requirements. For details about how to connect to the master node of the cluster, see Connecting to a Cluster Using kubectl.

Run the following command:

```bash
kubectl drain <node name> --ignore-daemonsets
```

**Step 2** Run the following commands as user root to update the kernel and restart the system. Ensure that the node has an EIP.

- For EulerOS 2.2, run the following commands:
  ```bash
  bash /var/paas/kubernetes/canal/openvswitch/can_ovs.sh uninstall
  yum update kernel -y
  reboot
  ```

- For EulerOS 2.5, run the following commands:
  ```bash
  bash /var/paas/kubernetes/canal/openvswitch/can_ovs.sh uninstall
  wget http://obs.cn-east-2.myhuaweicloud.com/cce-east/cce-openvswitch/kernel-3.10.0-862.14.1.0.h197.eulerosv2r7.x86_64.rpm
  rpm -ivh kernel-3.10.0-862.14.1.0.h197.eulerosv2r7.x86_64.rpm
  reboot
  ```

**Step 3** Run the following command as user root to upgrade the CCE components and adapt to the new kernel:

- For EulerOS 2.2, run the following commands:
  ```bash
  bash /var/paas/kubernetes/canal/openvswitch/can_ovs.sh install
  su paas; monit restart ovsdb-server ovs-vswitchd
  ```

- For EulerOS 2.5, run the following scripts:
  ```bash
  function upgrade_ovs()
  {
    mv /var/paas/kubernetes/canal/openvswitch /var/paas/kubernetes/canal/openvswitch.bak
    tar zxvf openvswitch-1.0.RC10.SPC100.B050.tar.gz -C /var/paas/kubernetes/canal/
    bash /var/paas/kubernetes/canal/openvswitch/can_ovs.sh install
    systemctl restart ovsdb-server ovs-vswitchd
  }
  upgrade_ovs
  ```

**Note**

You can obtain the environment address from Table 4-7.
Step 4  Mark the node as schedulable.

Run the following command:

```
kubectl uncordon <node name>
```

----End

Appendix

Table 4-7 OBS environment address information for each region

<table>
<thead>
<tr>
<th>Region</th>
<th>Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>CN East-Shanghai2</td>
<td>cce-east.obs.cn-east-2.myhuaweicloud.com</td>
</tr>
<tr>
<td>CN North-Beijing1</td>
<td>cce-north.obs.cn-north-1.myhuaweicloud.com</td>
</tr>
<tr>
<td>CN North-Beijing4</td>
<td>cce-north-4.obs.cn-north-4.myhuaweicloud.com</td>
</tr>
<tr>
<td>CN-Hong Kong</td>
<td>cce-ap-southeast.obs.ap-southeast-1.myhuaweicloud.com</td>
</tr>
<tr>
<td>CN South-Guangzhou</td>
<td>cce-south.obs.cn-south-1.myhuaweicloud.com</td>
</tr>
<tr>
<td>AP-Bangkok</td>
<td>cce-ap-southeast-2.obs.ap-southeast-2.myhuaweicloud.com</td>
</tr>
<tr>
<td>CN Southwest-Guiyang1</td>
<td>cce-static.cn-southwest-2.obs.cn-southwest-2.myhuaweicloud.com</td>
</tr>
<tr>
<td>CN Northeast-Dalian</td>
<td>cce-statics.cn-northeast-1.obs.cn-northeast-1.myhuaweicloud.com</td>
</tr>
<tr>
<td>CN South-Shenzhen</td>
<td>cce-south-2.obs.cn-south-2.myhuaweicloud.com</td>
</tr>
<tr>
<td>LA-Buenos Aires1</td>
<td>brazil-cce-static.obs.sa-argentina-1.myhuaweicloud.com</td>
</tr>
<tr>
<td>LA-Sao Paulo1</td>
<td>brazil-cce-static.obs.sa-brazil-1.myhuaweicloud.com</td>
</tr>
<tr>
<td>AF-Johannesburg</td>
<td>cce-statics.af-south-1.obs.af-south-1.myhuaweicloud.com</td>
</tr>
<tr>
<td>AP-Singapore</td>
<td>cce-statics.ap-southeast-3.obs.ap-southeast-3.myhuaweicloud.com</td>
</tr>
<tr>
<td>Russia-Moscow2</td>
<td>obs.ru-northwest-2.myhuaweicloud.com</td>
</tr>
<tr>
<td>LA-Santiago</td>
<td>obs.la-south-2.myhuaweicloud.com</td>
</tr>
<tr>
<td>CN East-Shanghai1</td>
<td>cce-statics.cn-east-3.obs.cn-east-3.myhuaweicloud.com</td>
</tr>
<tr>
<td>LA-Lima1</td>
<td>obs.myhuaweicloud.com</td>
</tr>
<tr>
<td>LA-Santiago2</td>
<td>brazil-cce-static.obs.sa-brazil-1.myhuaweicloud.com</td>
</tr>
</tbody>
</table>
4.14 Formula for Calculating the Reserved Resources of a Node

Some of the resources on the node need to run some necessary Kubernetes system components and resources to make the node as part of your cluster. Therefore, the total number of node resources and the number of assignable node resources in Kubernetes are different. The larger the node specifications, the more the containers deployed on the node. Therefore, Kubernetes needs to reserve more resources.

To ensure node stability, a certain amount of CCE node resources will be reserved for Kubernetes components (such as kubelet, kube-proxy, and docker) based on the node specifications.

CCE calculates the resources that can be allocated to user nodes as follows:

\[
\text{Allocatable resources} = \text{Total amount} - \text{Reserved amount} - \text{Eviction threshold}
\]

**NOTE**
For details about the eviction mechanism and thresholds in Kubernetes, see *What Should I Do If a Pod Fails to Be Evicted?*

**Rules for Reserving Node Memory**

You can use the following formula calculate how much memory you should reserve for running containers on a node:

\[
\text{Total reserved amount} = \text{Reserved memory for system components} + \text{Reserved memory for kubelet to manage pods}
\]

**Table 4-8** Reservation rules for system components

<table>
<thead>
<tr>
<th>Total Memory (TM)</th>
<th>Reserved Memory for System Components</th>
</tr>
</thead>
<tbody>
<tr>
<td>TM ≤ 8 GB</td>
<td>0 MB</td>
</tr>
<tr>
<td>8 GB &lt; TM ≤ 16 GB</td>
<td>([(\text{TM} - 8 \text{ GB}) \times 1024 \times 10%]) MB</td>
</tr>
<tr>
<td>16 GB &lt; TM ≤ 128 GB</td>
<td>([8 \text{ GB} \times 1024 \times 10% + (\text{TM} - 16 \text{ GB}) \times 1024 \times 6%]) MB</td>
</tr>
<tr>
<td>TM &gt; 128 GB</td>
<td>((8 \text{ GB} \times 1024 \times 10% + 112 \text{ GB} \times 1024 \times 6% + (\text{TM} - 128 \text{ GB}) \times 1024 \times 2%)) MB</td>
</tr>
</tbody>
</table>

**Table 4-9** Reservation rules for kubelet

<table>
<thead>
<tr>
<th>Total Memory (TM)</th>
<th>Number of Pods</th>
<th>Reserved Memory for kubelet</th>
</tr>
</thead>
<tbody>
<tr>
<td>TM ≤ 2 GB</td>
<td>-</td>
<td>TM x 25%</td>
</tr>
</tbody>
</table>
### Rules for Reserving Node CPU

**Table 4-10** Node CPU reservation rules

<table>
<thead>
<tr>
<th>Total CPU Cores (Total)</th>
<th>Reserved CPU Cores</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total ≤ 1 core</td>
<td>Total x 6%</td>
</tr>
<tr>
<td>1 core &lt; Total ≤ 2 cores</td>
<td>1 core x 6% + (Total – 1 core) x 1%</td>
</tr>
<tr>
<td>2 cores &lt; Total ≤ 4 cores</td>
<td>1 core x 6% + 1 core x 1% + (Total – 2 cores) x 0.5%</td>
</tr>
<tr>
<td>Total &gt; 4 cores</td>
<td>1 core x 6% + 1 core x 1% + 2 cores x 0.5% + (Total – 4 cores) x 0.25%</td>
</tr>
</tbody>
</table>

---

**NOTICE**

CCE reserves an extra 100 MiB for kubelet eviction.
4.15 Creating a Linux LVM Disk Partition for Docker

Scenario

This section describes how to check whether there are available raw disks and Linux LVM disk partitions and how to create Linux LVM disk partitions.

Prerequisites

To improve the system stability, attach a data disk to Docker and use the direct-lvm mode.

Procedure

Step 1
Check whether available raw disks exist on the current node.

1. Log in to the target node as the root user.
2. Check the raw disk device.
   
   lsblk -l | grep disk
   
   If the following information is displayed, the raw disks named xvda and xvdb exist on the node.
   
   xvda 202:0 0 40G 0 disk
   xvdb 202:16 0 100G 0 disk

3. Check whether the raw disk is in use.
   
   lsblk /dev/<devicename>
   
   devicename indicates the raw disk name, for example, xvda and xvdb in the previous step.
   
   Run the lsblk /dev/xvda and lsblk /dev/xvdb commands. If the following information is displayed, xvda has been partitioned and used while xvdb is available. If no raw disk is available, bind an EVS disk to the node. It is advised that the disk space be no less than 80 GB.

   NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
   xvda   202:0 0 40G 0 disk
   └─xvda1 202:1 0 100M 0 part /boot
   └─xvda2 202:2 0 39.9G 0 part /
   NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
   xvdb 202:16 0 100G 0 disk

Step 2
Check whether there are partitions available. Currently, only Linux LVM partitions are supported.

1. Log in to the target node as the root user.
2. Check the partition whose system type is Linux LVM.
   
   sfdisk -l 2>/dev/null | grep "Linux LVM"
   
   If the following information is displayed, two Linux LVM partitions, /dev/nvme0n1p1 and /dev/nvme0n1p2, exist in the system.

   /dev/nvme0n1p1  1 204800 204800 209715200 8e Linux LVM
   /dev/nvme0n1p2  204801 409600 204800 209715200 8e Linux LVM

3. Check whether the partition is in use.
   
   lsblk <partdevice>
<partdevice> is the Linux LVM partition found in the previous step.

In this example, run the `lsblk/dev/nvme0n1p1` and `lsblk/dev/nvme0n1p2` commands. If the following information is displayed, partition `nvme0n1p` is in use while `nvme0n1p2` is available.

<table>
<thead>
<tr>
<th>NAME</th>
<th>MAJ:MIN</th>
<th>RM</th>
<th>SIZE</th>
<th>RO</th>
<th>TYPE</th>
<th>MOUNTPOINT</th>
</tr>
</thead>
<tbody>
<tr>
<td>nvme0n1p1</td>
<td>259:3</td>
<td>0</td>
<td>200G</td>
<td>0</td>
<td>part</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>lvm</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NAME</td>
<td>MAJ:MIN</td>
<td>RM</td>
<td>SIZE</td>
<td>RO</td>
<td>TYPE</td>
<td>MOUNTPOINT</td>
</tr>
<tr>
<td>-----------------------</td>
<td>---------</td>
<td>----</td>
<td>--------</td>
<td>----</td>
<td>----------</td>
<td>-------------</td>
</tr>
<tr>
<td>nvme0n1p2</td>
<td>259:1</td>
<td>0</td>
<td>100G</td>
<td>0</td>
<td>part</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
| If no AZ is available, perform **Step 3** to create a partition for Docker.

**Step 3** Create a Linux LVM disk partition for Docker.

1. Run the following command to create a disk partition. `devicename` indicates the available raw disk name, for example, `xvdb` in **Step 1**.

   ```bash
disk /dev/devicename
   ```

2. Enter `n` to create a new partition. Enter `p` to display the primary partition number. Enter `4` to indicate the fourth primary partition.

   ```
   Figure 4-27 Creating a partition
   ```

3. Configure the start and last sectors as follows for example:
   `Start sector (1048578048-4294967295, 1048578048 by default):`
   `1048578048`
   `Last sector, +sector or size {K, M, or G} (1048578048-4294967294, 4294967294 by default): +100G`
   This configuration indicates that partition 4 has been set to the Linux type and the size is 100 GiB.

4. Enter `t` to change the partition system type. Enter the hex code `8e` when prompted to change the system type to Linux LVM.

   ```
   Command (enter help): t
   Partition type
   p primary (0 primary, 0 extended, 4 free)
   e extended (container for logical partitions)
   Select (default p): p
   Partition number (1-4, default 1): 4
   ```

   ```
   This configuration changes the type of the partition Linux to **Linux LVM**.
   ```

5. Enter `w` to save the modification.

   ```
   Command (enter help): w
   The partition table has been altered!
   ```

6. Run the `partprobe` command to refresh the disk partition.

    ----End

---

### 4.16 Docker Disk Space

When creating a node, you need to configure data disks for the node.
The space of a data disk is divided into Kubernetes space and user space. The Kubernetes space consists of the following two parts:

- **Docker space (90% by default):** stores Docker working directories, Docker image data, and image metadata.
- **kubelet space (10% by default):** stores pod configuration files, secrets, and mounted storage such as emptyDir volumes.

The Docker space size affects image download and container startup and running. This section describes how the Docker space is used so that you can configure the Docker space accordingly.

**Docker Space Description**

By default, a data disk, 100 GB for example, is divided as follows (depending on the container storage Rootfs):

- **Rootfs (Device Mapper):**
  - The `/var/lib/docker` directory is used as the Docker working directory and occupies 20% of the Docker space by default. (Space size of the `/var/lib/docker` directory = Data disk space x 90% x 20%)
  - The thin pool is used to store Docker image data, image metadata, and container data, and occupies 80% of the Docker space by default. (Thin pool space = Data disk space x 90% x 80%)

- **Rootfs (OverlayFS):** No separate thinpool. The entire Docker space is in the `/var/lib/docker` directory.
Using rootfs for container storage in CCE

- CCE cluster: EulerOS 2.5 and CentOS 7.6 nodes use Device Mapper, and Ubuntu 18.09 nodes use OverlayFS.

### Docker Space and Containers

The number of pods and the space configured for each container determine whether the Docker space of a node is sufficient.

\[
\text{Docker space} > \text{Number of containers} \times \text{Available data space for a single container (basesize)}
\]

**When device mapper is used**, although you can limit the size of the /home directory of a single container (to 10 GB by default), all containers on the node still share the thin pool of the node for storage. They are not completely isolated. When the sum of the thin pool space used by certain containers reaches the upper limit, other containers cannot run properly.

In addition, after a file is deleted in the /home directory of the container, the thin pool space occupied by the file is not released immediately. Therefore, even if basesize is set to 10 GB, the thin pool space occupied by files keeps increasing until 10 GB when files are created in the container. The space released after file deletion will be reused but after a while. If the number of containers on the node multiplied by basesize is greater than the thin pool space size of the node, there is a possibility that the thin pool space has been used up.

### Garbage Collection Policies for Container Images

When the Docker space is insufficient, image garbage collection is triggered.

The policy for garbage collecting images takes two factors into consideration: **HighThresholdPercent** and **LowThresholdPercent**. Disk usage above the high threshold (default: 85%) will trigger garbage collection. The garbage collection will delete least recently used images until the low threshold (default: 80%) has been met.

### Docker Space Configuration Suggestions

- The Docker space should be greater than the total disk space used by containers. Formula: \( \text{Docker space} > \text{Number of containers} \times \text{Available data space for a single container (basesize)} \)

- You are advised to create and delete files of containerized services in local storage volumes (such as emptyDir and hostPath volumes) or cloud storage directories mounted to the containers. In this way, the thin pool space is not occupied. emptyDir volumes occupy the kubelet space. Therefore, properly plan the size of the kubelet space. For details about local storage and cloud storage operations, see [Storage (CSI)].
Docker uses the OverlayFS storage mode. This mode is used in Ubuntu 18.04 nodes in CCE clusters by default. You can deploy services on these nodes to prevent that the disk space occupied by files created or deleted in containers is not released immediately.

**Common Issues**

**How Do I Expand the Storage Capacity of a Container?**

**Expanding the Disk Capacity of a Node in a CCE Cluster**

### 4.17 Adding a Second Data Disk to a Node in a CCE Cluster

You can use the **pre-installation script** feature to configure CCE cluster nodes (ECSSs). For details, see **Buying a Hybrid Cluster - Advanced Kubernetes Settings**.

**NOTE**

- When creating a node in a cluster of v1.13.10 or later, if a data disk is not managed by LVM, follow the instructions in this section to format the data disk before adding the disk. Otherwise, the data disk will still be managed by LVM.
- When creating a node in a cluster of v1.13.10 or earlier, if a data disk is not managed by LVM, format the data disk. Otherwise, either this data disk or the first data disk will be managed by LVM, which is not as expected.

Before using this feature, write a script that can format data disks and save it to your OBS bucket. This script must be executed by user **root**.

**Input Parameters**

1. Set the script name to **formatdisk.sh**, save the script to your OBS bucket, and obtain the address of the script in OBS. For details, see **Accessing an Object Using Its URL**.
2. You need to specify the size of the Docker data disk (the data disk managed by LVM is called the Docker data disk). The size of the Docker disk must be different from that of the second disk. For example, the Docker data disk is 100 GB and the new disk is 110 GB.
3. Set the mount path of the second data disk, for example, `/data/code`.

Run the following command in the pre-installation script to format the disk:

```
cd /tmp; curl -k -X GET OBS bucket address/formatdisk.sh -1 -O; fdisk -l; sleep 60; bash -x formatdisk.sh 100 /data/code; fdisk -l
```

Example script (**formatdisk.sh**):

```
dockerdisksize=$1
mountdir=$2
systemdisksize=40
i=0
while [ $systemdisksize -gt $i ]; do
    echo $i;
    if ! blkid -o KNAME,TYPE | grep disk | grep -v nvme | awk '{print $1}' | awk '{ print "$(/dev/"$1)" }' | wc -l -ge 3 ]; then
        break
    fi
    i+=10
done
```

```
```

Cloud Container Engine
User Guide

4 Nodes

Issue 01 (2021-09-08) Copyright © Huawei Technologies Co., Ltd. 145
```bash
else
    sleep 5
fi;
i=$[i+1]
done
all_devices=$(lsblk -o KNAME,TYPE | grep disk | grep -v nvme | awk '{print $1}' | awk '{ print "/dev/"$1}')
for device in $all_devices[@]; do
    isRawDisk=$(lsblk -n $device 2>/dev/null | grep disk | wc -l)
    if [[ $isRawDisk > 0 ]]; then
        # is it partitioned ?
        match=$(lsblk -n $device 2>/dev/null | grep -v disk | wc -l)
        if [[ $match > 0 ]]; then
            # already parted
            [[ -n "$(DOCKER_BLOCK_DEVICES)" ]] && echo "Raw disk $device has been partioned, will skip this device"
            continue
        fi
    else
        isPart=$(lsblk -n $device 2>/dev/null | grep part | wc -l)
        if [[ $isPart -ne 1 ]]; then
            # not parted
            [[ -n "$(DOCKER_BLOCK_DEVICES)" ]] && echo "Disk $device has not been partitioned, will skip this device"
            continue
        fi
        # is used ?
        match=$(lsblk -n -o MOUNTPOINT $device 2>/dev/null)
        if [[ -n $isMount ]]; then
            # already used
            [[ -n "$(DOCKER_BLOCK_DEVICES)" ]] && echo "Disk $device has been used, will skip this device"
            continue
        fi
        isLvm=$(sfdisk -lqL 2>>/dev/null | grep $device | grep "8e.*Linux LVM")
        if [[ -n $isLvm ]]; then
            # part system type is not Linux LVM
            [[ -n "$(DOCKER_BLOCK_DEVICES)" ]] && echo "Disk $device system type is not Linux LVM, will skip this device"
            continue
        fi
        block_devices_size=$(lsblk -n -o SIZE $device 2>/dev/null | awk '{ print $1}')
        if [[ $block_devices_size"x" != "$dockerdisksizeGx" ]] && [[ $block_devices_size"x" != "$systemdisksizeGx" ]]; then
            echo "n
            p
            1
            w"
            fdisk $device
            mkfs -t ext4 $device1
            mkdir -p $mountdir
            uuid=$(blkid $device1 | awk '{print $2}')
            echo "$uuid") $mountdir ext4 noatime 0 0" | tee -a /etc/fstab >/dev/null
            mount $mountdir
        fi
    done
```

**NOTE**

If the preceding example cannot be executed, use the dos2unix tool to convert the format.
5 Node Pools

5.1 Node Pool Overview

Introduction

CCE introduces node pools to help you better manage nodes in Kubernetes clusters. A node pool contains one node or a group of nodes with identical configuration in a cluster.

You can create custom node pools on the CCE console. With node pools, you can quickly create, manage, and destroy nodes without affecting the cluster. All nodes in a custom node pool have identical parameters and node type. You cannot configure a single node in a node pool; any configuration changes affect all nodes in the node pool.

In CCE, the nodes configured during cluster creation are grouped into the default node pool named DefaultPool. This node pool cannot be edited, deleted, migrated, expanded, or auto scaled.

You can also use node pools for auto scaling.

- When a pod in a cluster cannot be scheduled due to insufficient resources, scale-out can be automatically triggered.
- When there is an idle node or a monitoring metric threshold is met, scale-in can be automatically triggered.

This section describes how node pools work in CCE and how to create and manage node pools.
Node Pool Architecture

Figure 5-1 Overall architecture of a node pool

Generally, all nodes in a node pool have the following same attributes:

- Node OS
- Startup parameters of Kubernetes components on a node
- User-defined startup script of a node
- **K8S Labels** and **Taints**

CCE provides the following extended attributes for node pools:

- Node pool OS
- Maximum number of pods on each node in a node pool

Applicable Scenarios

When a large-scale cluster is required, you are advised to use node pools to manage nodes.

The following table describes multiple scenarios of large-scale cluster management and the functions of node pools in each scenario.

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multiple heterogeneous nodes (with different models and configurations) in the cluster</td>
<td>Nodes can be grouped into different pools for management.</td>
</tr>
<tr>
<td>Frequent node scaling required in a cluster</td>
<td>Node pools support auto scaling to dynamically add or reduce nodes.</td>
</tr>
<tr>
<td>Complex application scheduling rules in a cluster</td>
<td>Node pool tags can be used to quickly specify service scheduling rules.</td>
</tr>
</tbody>
</table>
### Scenario
- Routine node maintenance

### Function
- Node pools allow you to easily upgrade Kubernetes and Docker versions.

## Functions and Precautions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Creating a node pool</td>
<td>Adding a new node pool. The yearly/monthly billing mode is not supported.</td>
<td>It is recommended that a cluster contain no more than 100 node pools.</td>
</tr>
<tr>
<td>Deleting a node pool</td>
<td>Deleting a node pool will delete nodes in the pool. Pods on these nodes will be automatically migrated to available nodes in other node pools.</td>
<td>If pods in the node pool have a specific node selector and none of the other nodes in the cluster satisfies the node selector, the pods will become unschedulable.</td>
</tr>
<tr>
<td>Enabling auto scaling for a node pool</td>
<td>After auto scaling is enabled, nodes will be automatically created or deleted in the node pool based on the cluster loads.</td>
<td>You are advised not to store important data on nodes in a node pool because after auto scaling, data cannot be restored as nodes may be deleted.</td>
</tr>
<tr>
<td>Enabling auto scaling for a node pool</td>
<td>After auto scaling is disabled, the number of nodes in a node pool will not automatically change with the cluster loads.</td>
<td>/</td>
</tr>
<tr>
<td>Adjusting the size of a node pool</td>
<td>The number of nodes in a node pool can be directly adjusted. If the number of nodes is reduced, nodes are randomly removed from the current node pool.</td>
<td>After auto scaling is enabled, you are not advised to manually adjust the node pool size.</td>
</tr>
<tr>
<td>Changing node pool configurations</td>
<td>You can modify the node pool name, node quantity, Kubernetes labels, taints, and resource tags.</td>
<td>The modified Kubernetes labels and taints will apply to all nodes in the node pool, which may cause pod re-scheduling. Therefore, exercise caution when performing this operation.</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
<td>Notes</td>
</tr>
<tr>
<td>---------------------------</td>
<td>------------------------------------------------------------------------------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>Adding an existing node</td>
<td>Nodes that do not belong to the cluster can be added to a node pool. The</td>
<td>Unless required, you are not advised to add existing nodes. You are</td>
</tr>
<tr>
<td>to a node pool</td>
<td>following requirements must be met:</td>
<td>advised to create a node pool.</td>
</tr>
<tr>
<td></td>
<td>● The node to be added and the CCE cluster are in the same VPC and subnet.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>● The node is not used by other clusters and has the same configurations</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(such as specifications and billing mode) as the node pool.</td>
<td></td>
</tr>
<tr>
<td>Removing a node from a</td>
<td>Nodes in a node pool can be migrated to the default node pool of the same</td>
<td>Nodes in the default node pool cannot be migrated to other node pools,</td>
</tr>
<tr>
<td>node pool</td>
<td>cluster.</td>
<td>and nodes in a user-created node pool cannot be migrated to other</td>
</tr>
<tr>
<td></td>
<td></td>
<td>user-created node pools.</td>
</tr>
<tr>
<td>Cloning a node pool</td>
<td>You can copy the configuration of an existing node pool to create a new node</td>
<td>/</td>
</tr>
<tr>
<td>Setting Kubernetes</td>
<td>You can configure core components with fine granularity.</td>
<td></td>
</tr>
<tr>
<td>parameters</td>
<td></td>
<td>● This function is supported only in clusters of v1.15 and later. It</td>
</tr>
<tr>
<td></td>
<td></td>
<td>is not displayed for versions earlier than v1.15</td>
</tr>
<tr>
<td></td>
<td></td>
<td>● The default node pool DefaultPool does not support this type of</td>
</tr>
<tr>
<td></td>
<td></td>
<td>configuration.</td>
</tr>
</tbody>
</table>

### Deploying a Workload in a Specified Node Pool

When creating a workload, you can constrain pods to run in a specified node pool.

For example, on the CCE console, you can set the affinity between the workload and the node on the **Scheduling Policies** tab page on the workload details page to forcibly deploy the workload to a specific node pool. In this way, the workload runs only on nodes in the node pool. If you need to better control where the workload is to be scheduled, you can use affinity or anti-affinity policies between workloads and nodes described in **Scheduling Policy Overview**.

For example, you can use container’s resource request as a nodeSelector so that workloads will run only on the nodes that meet the resource request.

If the workload definition file defines a container that requires four CPUs, the scheduler will not choose the nodes with two CPUs to run workloads.
Related Operations

You can log in to the CCE console and refer to the following sections to perform operations on node pools:

- Creating a Node Pool
- Managing a Node Pool
- Creating a Deployment
- Workload-Node Affinity

5.2 Creating a Node Pool

Scenario

This section describes how to create a node pool for a running CCE cluster and how to perform operations on the node pool. For details about how a node pool works, see Node Pool Overview.

Procedure

To create a node pool in a cluster, perform the following steps:

**Step 1** Log in to the CCE console. In the navigation pane, choose Resource Management > Node Pools.

**Step 2** In the upper right corner of the page, click Create Node Pool.

**Step 3** Set node pool parameters.

- **Billing Mode**
  
  Only pay-per-use is supported. In this billing mode, resources are billed by hour.
  
  After a node pool is created, the billing mode of resources in the created node pool cannot be changed to the yearly/monthly billing, while this change is allowed for resources in the default node pool. You can migrate resources from a self-built node pool to the default node pool and then change the billing mode to yearly/monthly billing. For details, see Migrating a Node.

- **Current Region**: geographic location of the node pool to be created.
  
  To minimize network latency and resource access time, select the region nearest to your node pool. Cloud resources are region-specific and cannot be used across regions over internal networks.

- **Name**: name of the new node pool. By default, the name is in the format of Cluster name-nodepool-Random number. You can also use a custom name.

- **Node Type**: Currently, only VM nodes are supported.

- **Nodes**: number of nodes to be purchased for this node pool. The value cannot exceed the maximum number of nodes that can be managed by the cluster. To increase the quota limit on nodes, click Increase quota.

- **Autoscaler**
  
  - By default, this parameter is disabled.
- After you enable autoscaler by clicking , nodes in the node pool will be automatically created or deleted based on cluster loads.

  ▪ **Maximum Nodes** and **Minimum Nodes**: You can set the maximum and minimum number of nodes to ensure that the number of nodes to be scaled is within a proper range.

  ▪ **Priority**: Set this parameter based on service requirements. A larger value indicates a higher priority. For example, if this parameter is set to 1 and 4 respectively for node pools A and B, B has a higher priority than A. If the priorities of multiple node pools are set to the same value, for example, 2, the node pools are not prioritized and the system performs scaling based on the minimum resource waste principle.

  □ NOTE

  CCE selects a node pool for auto scaling based on the following policies:

  1. CCE uses algorithms to determine whether a node pool meets the conditions to allow scheduling of a pod in pending state, including whether the node resources are greater than requested by the pod, and whether the nodeSelect, nodeAffinity, and taints meet the conditions. In addition, the node pools that fail to be scaled (due to insufficient resources or other reasons) and are still in the 15-minute cool-down interval are filtered.

  2. If multiple node pools meet the scaling requirements, the system checks the priority of each node pool and selects the node pool with the highest priority for scaling. The value ranges from 0 to 100 and the default priority is 0. The value 100 indicates the highest priority, and the value 0 indicates the lowest priority.

  3. If multiple node pools have the same priority or no priority is configured for them, the system selects the node pool that will consume the least resources based on the configured VM specification.

  4. If the VM specifications of multiple node pools are the same but the node pools are deployed in different AZs, the system randomly selects a node pool to trigger scaling.

  ▪ **Scale-In Cooling Interval**: Set this parameter in the unit of minute or hour. This parameter indicates the interval between the previous scale-out action and the next scale-in action.

  Scale-in cooling intervals can be configured in the node pool settings and the autoscaler add-on settings.

  **Scale-in cooling interval configured in a node pool**

  This interval indicates the period during which nodes added to the current node pool after a scale-out operation cannot be deleted. This interval takes effect at the node pool level.

  **Scale-in cooling interval configured in the autoscaler add-on**

  The interval after a scale-out indicates the period during which the entire cluster cannot be scaled in after the autoscaler add-on triggers scale-out (due to the unschedulable pods, metrics, and scaling policies). This interval takes effect at the cluster level.

  The interval after a node is deleted indicates the period during which the cluster cannot be scaled in after the autoscaler add-on triggers scale-in. This interval takes effect at the cluster level.
The interval after a failed scale-in indicates the period during which the cluster cannot be scaled in after the autoscaler add-on triggers scale-in. This interval takes effect at the cluster level.

**NOTE**

You are advised not to store important data on nodes in a node pool because after auto scaling, data cannot be restored as nodes may be deleted.

If **Autoscaler** is enabled, install the **autoscaler add-on** to use the auto scaling feature.

- **AZ**: An AZ is a physical region where resources use independent power supply and networks. AZs are physically isolated but interconnected through an internal network.

  Set an AZ based on your requirements. After a node pool is created, **AZ** cannot be modified. Exercise caution when selecting an AZ for the node pool.

- ** Specifications**: Select node specifications that best fit your business needs.
  - **General-purpose**: provides a balance of computing, memory, and network resources. It is a good choice for many applications, such as web servers, workload development, workload testing, and small-scale databases.
  - **Memory-optimized**: provides higher memory capacity than general-purpose nodes and is suitable for relational databases, NoSQL, and other workloads that are both memory-intensive and data-intensive.
  - **General computing-basic**: provides a balance of computing, memory, and network resources and uses the vCPU credit mechanism to ensure baseline computing performance. Nodes of this type are suitable for applications requiring burstable high performance, such as light-load web servers, enterprise R&D and testing environments, and low- and medium-performance databases.
  - **GPU-accelerated**: provides powerful floating-point computing and is suitable for real-time, highly concurrent massive computing. Graphical processing units (GPUs) of P series are suitable for deep learning, scientific computing, and CAE. GPUs of G series are suitable for 3D animation rendering and CAD. **GPU-accelerated nodes can be added only to clusters of v1.11 or later.**
  - **High-performance computing**: provides stable and ultra-high computing performance and is suitable for scientific computing and workloads that demand ultra-high computing power and throughput.
  - **General computing-plus**: provides stable performance and exclusive resources to enterprise-class workloads with high and stable computing performance.
  - **Disk-intensive**: supports **local disk storage** and provides high network performance. It is designed for workloads requiring high throughput and data switching, such as big data workloads.
  - **Ultra-high I/O**: delivers ultra-low SSD access latency and ultra-high IOPS performance. This type of specifications is ideal for high-performance relational databases, NoSQL databases (such as Cassandra and MongoDB), and Elasticsearch.
To ensure node stability, CCE automatically reserves some resources to run necessary system components. For details, see Formula for Calculating the Reserved Resources of a Node.

- **OS**: Select an OS for the node to be created. In certain regions, only OSs are displayed and options Public image and Private image are unavailable.
  - **Public image**: Select an OS for the node.
  - **Private image (OBT)**: If no private image is available, click Creating a Private Image to create one. This function is available only for clusters of v1.15 or later. For details, see Using a Private Image to Build a Worker Node Image?

### NOTICE

Reinstalling the OS or modifying OS configurations could make the node unavailable. Exercise caution when performing these operations. For details, see High-Risk Operations and Solutions.

- **VPC**: The value is the same as that of the cluster and cannot be changed. This parameter is displayed only for clusters of v1.13.10-r0 and later.
- **Subnet**: A subnet improves network security by providing exclusive network resources that are isolated from other networks.
  
  You can select any subnet in the cluster VPC. Cluster nodes can belong to different subnets.
  
  Ensure that the DNS server in the subnet can resolve the OBS domain name. Otherwise, nodes cannot be created. This parameter is displayed only for clusters of v1.13.10-r0 and later.

- **System Disk**: Set the system disk space of the worker node. The value ranges from 40GB to 1024 GB. The default value is 40GB.
  
  By default, system disks support High I/O (SAS) and Ultra-high I/O (SSD) EVS disks. For details, see EVS Disk Overview.
  
  Encryption: Data disk encryption safeguards your data. Snapshots generated from encrypted disks and disks created using these snapshots automatically inherit the encryption function. This function is available only in certain regions.
- **Encryption** is not selected by default.
- After you select **Encryption**, you can select an existing key in the displayed **Encryption Setting** dialog box. If no key is available, click the link next to the drop-down box to create a key. After the key is created, click the refresh icon.

- **Data Disk**: Set the data disk space of the worker node. The value ranges from 100 GB to 32,768 GB. The default value is 100 GB. The EVS disk types provided for the data disk are the same as those for the system disk.

<table>
<thead>
<tr>
<th>CAUTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>If the data disk is uninstalled or damaged, the Docker service becomes abnormal and the node becomes unavailable. You are advised not to delete the data disk.</td>
</tr>
</tbody>
</table>

- **LVM**: If this option is selected, CCE data disks are managed by the Logical Volume Manager (LVM). On this condition, you can adjust the disk space allocation for different resources. This option is selected for the first disk by default and cannot be unselected. You can choose to enable or disable LVM for new data disks.
  - This option is selected by default, indicating that LVM management is enabled.
  - You can deselect the check box to disable LVM management.

<table>
<thead>
<tr>
<th>CAUTION</th>
</tr>
</thead>
</table>
| ○ Disk space of the data disks managed by LVM will be allocated according to the ratio you set.  
○ When creating a node in a cluster of v1.13.10 or later, if LVM is not selected for a data disk, follow instructions in **Adding a Second Data Disk to a Node in a CCE Cluster** to fill in the pre-installation script and format the data disk. Otherwise, the data disk will still be managed by LVM.  
○ When creating a node in a cluster earlier than v1.13.10, you must format the data disks that are not managed by LVM. Otherwise, either these data disks or the first data disk will be managed by LVM. |

- **Encryption**: Data disk encryption provides powerful protection for your data. Snapshots generated from encrypted disks and disks created using these snapshots automatically inherit the encryption function.

<table>
<thead>
<tr>
<th>This function is supported only for clusters of v1.13.10 or later in certain regions, and is not displayed for clusters of v1.13.10 or earlier.</th>
</tr>
</thead>
</table>
| - **Encryption** is not selected by default.  
- After you select **Encryption**, you can select an existing key in the displayed **Encryption Setting** dialog box. If no key is available, click |
the link next to the drop-down box to create a key. After the key is created, click the refresh icon.

- **Add Data Disk**: Currently, a maximum of two data disks can be attached to a node. After the node is created, you can go to the ECS console to attach more data disks. This function is available only to clusters of certain versions.

- **Data disk space allocation**: Click to specify the resource ratio for Kubernetes Space and User Space. Disk space of the data disks managed by LVM will be allocated according to the ratio you set. This function is available only to clusters of certain versions.

  - **Kubernetes Space**: You can specify the ratio of the data disk space for storing Docker and kubectl resources. Docker resources include the Docker working directory, Docker images, and image metadata. Kubectl resources include pod configuration files, secrets, and emptyDirs.
  
  The Docker space size is determined by your service requirements. For details, see Docker Disk Space.

  - **User Space**: You can set the ratio of the disk space that is not allocated to Kubernetes resources and the path to which the user space is mounted.

    **NOTE**

    Note that the mount path cannot be /, /home/paas, /var/paas, /var/lib, /var/script, /var/log, /mnt/paas, or /opt/cloud, and cannot conflict with the system directories (such as bin, lib, home, root, boot, dev, etc, lost+found, mnt, proc, sbin, srv, tmp, var, media, opt, selinux, sys, and usr). Otherwise, the system or node installation will fail.

If the cluster version is v1.13.10-r0 or later and the node type is Disk-intensive or Ultra-high I/O, the following options are displayed for data disks:

- **EVS**: Parameters are the same as those when the node type is not Disk-intensive or Ultra-high I/O. For details, see Data Disk above.

- **Local disk**: Local disks may break down and do not ensure data reliability. It is recommended that you store service data in EVS disks, which are more reliable than local disks.

  Local disk parameters are as follows:

  - **Disk Mode**: If the node type is disk-intensive, the supported disk mode is HDD. If the node type is ultra-high I/O, the supported disk mode is SSD.

  - **Read/Write Mode**: When multiple local disks exist, you can set the read/write mode. The serial and sequential modes are supported. Sequential indicates that data is read and written in linear mode. When a disk is used up, the next disk is used. Serial indicates that data is read and written in striping mode, allowing multiple local disks to be read and written at the same time.

  - **Kubernetes Space**: You can specify the ratio of the data disk space for storing Docker and kubectl resources. Docker resources include
the Docker working directory, Docker images, and image metadata. Kubelet resources include pod configuration files, secrets, and emptyDirs.

- **User Space**: You can set the ratio of the disk space that is not allocated to Kubernetes resources and the path to which the user space is mounted.

**NOTICE**
- The ratio of disk space allocated to the Kubernetes space and user space must be equal to 100% in total. You can click to refresh the data after you have modified the ratio.
- By default, disks run in the direct-lvm mode. If data disks are removed, the loop-lvm mode will be used and this will impair system stability.

**Figure 5-3 Setting a local disk**

- **Login Mode**: You can use a password or key pair.
  - **Password**: The default username is root. Enter the password for logging in to the node and confirm the password. Be sure to remember the password as you will need it when you log in to the node.
  - **Key pair**: Select the key pair used to log in to the node. You can select a shared key.

A key pair is used for identity authentication when you remotely log in to a node. If no key pair is available, click **Create a key pair**. For details on how to create a key pair, see **Creating a Key Pair**.

**NOTICE**
When creating a node using a key pair, IAM users can select only the key pairs created by their own, regardless of whether these users are in the same group. For example, user B cannot use the key pair created by user A to create a node, and the key pair is not displayed in the drop-down list on the CCE console.
Step 4 Advanced ECS Settings (optional): Click to show advanced ECS settings.

- ECS Group: An ECS group logically groups ECSs. The ECSs in the same ECS group comply with the same policy associated with the ECS group.
  - Anti-affinity: ECSs in an ECS group are deployed on different physical hosts to improve service reliability.
  - Fault domain: ECSs in an ECS group are deployed in multiple failure domains so that a failure in one failure domain will not affect the ECSs in other failure domains, thereby improving service reliability. This option is displayed only when the environment supports failure domains. This option is not supported if a worker node is deployed in a random AZ.

Select an existing ECS group, or click Create ECS Group to create a new one. After the ECS group is created, click the refresh button.

- Resource Tags: By adding tags to resources, you can classify resources. You can create predefined tags in Tag Management Service (TMS). Predefined tags are visible to all service resources that support the tagging function. You can use predefined tags to improve tag creation and migration efficiency.

CCE will automatically create the "CCE-Dynamic-Provisioning-Node=node id" tag. A maximum of 5 tags can be added.

- Agency: An agency is created by a tenant administrator on the IAM console. By creating an agency, you can share your cloud server resources with another account, or entrust a more professional person or team to manage your resources. For details on how to create an agency, see Cloud Service Delegation. To authorize an ECS or BMS to call cloud services, select Cloud service as the agency type, click Select, and then select ECS BMS.

- Pre-installation Script: Enter a maximum of 1,000 characters.

The script will be executed before Kubernetes software is installed. Note that if the script is incorrect, Kubernetes software may not be installed successfully. The script is usually used to format data disks.

- Post-installation Script: Enter a maximum of 1,000 characters.

The script will be executed after Kubernetes software is installed and will not affect the installation. The script is usually used to modify Docker parameters.

- Subnet IP Address: Select Automatically assign IP address (recommended) or Manually assigning IP addresses.

Step 5 Advanced Kubernetes Settings (optional): Click to show advanced Kubernetes settings.

- Max Pods: maximum number of pods that can be created on a node, including the system's default pods. If the cluster uses the VPC network model, the maximum value is determined by the number of IP addresses that can be allocated to containers on each node. This limit prevents the node from being overloaded by managing too many pods.
**Taints:** This field is left blank by default. Taints allow nodes to repel a set of pods. You can add a maximum of 10 taints for each node. Each taint contains the following parameters:

- **Key:** A key must contain 1 to 63 characters starting with a letter or digit. Only letters, digits, hyphens (-), underscores (_), and periods (.) are allowed. A DNS subdomain name can be used as the prefix of a key.

- **Value:** A value must start with a letter or digit and can contain a maximum of 63 characters, including letters, digits, hyphens (-), underscores (_), and periods (.).

- **Effect:** Available options are **NoSchedule**, **PreferNoSchedule**, and **NoExecute**.

--- **NOTICE**

- If taints are used, you must configure tolerations in the YAML files of pods. Otherwise, scale-up may fail or pods cannot be scheduled onto the added nodes.

- After a node pool is created, you can click **Edit** to modify its configuration. The modification will be synchronized to all nodes in the node pool.

---

**K8S Labels:** Labels are key/value pairs that are attached to objects, such as pods. Labels are used to specify identifying attributes of objects that are meaningful and relevant to users, but do not directly imply semantics to the core system. For more information, see Labels and Selectors.

**Maximum Data Space per Container:** maximum data space that can be used by a container. The value ranges from 10 GB to 80 GB. If the value of this field is larger than the data disk space allocated to Docker resources, the latter will override the value specified here. Typically, 90% of the data disk space is allocated to Docker resources. This parameter is displayed only for clusters of v1.13.10-r0 and later.

---

**Step 6** (Optional) Click **Add Node Pool** on the left to add multiple node pools. You can view the available node pool quotas under the button.

**Step 7** Click **Next: Confirm** to confirm the configured service parameters, fees, and specifications.

**Step 8** Click **Submit**.

It takes about 6 to 10 minutes to create a node pool. You can click **Back to Node Pool List** to perform other operations on the node pool or click **Go to Node Pool Events** to view the node pool details. If the status of the node pool is Normal, the node pool is successfully created.

--- End

**Viewing Node Pools in a Cluster**

**Step 1** Log in to the CCE console. In the navigation pane, choose **Resource Management > Node Pools**.
Step 2 In the upper right corner of the node pool list, select a cluster. All node pools in the cluster will be displayed. You can view the node type, node specifications, autoscaler status, and OS of each node pool.

Figure 5-5 Viewing node pools in a cluster

- **NOTE**
  - A default node pool DefaultPool is automatically created in each cluster. The default node pool cannot be edited, deleted, or migrated. All nodes created during and after cluster creation are displayed in the default node pool.
  - To display a list of nodes in DefaultPool, click the Nodes subcard in the DefaultPool card.

Step 3 To filter node pools by autoscaler status, select the autoscaler status in the upper right corner of the node pool list.

Step 4 In the node pool list, click a node pool name. On the node pool details page, view the basic information, advanced ECS settings, advanced Kubernetes settings, and node list of the node pool.

Figure 5-6 Node pool details

---End

5.3 Managing a Node Pool

Notes and Constraints

The default node pool DefaultPool does not support the following management operations.
Configuring Kubernetes Parameters

CCE allows you to highly customize Kubernetes parameter settings on core components in a cluster. For more information, see kubelet.

This function is supported only in clusters of v1.15 and later. It is not displayed for clusters earlier than v1.15.

**Step 1** Log in to the CCE console. In the navigation pane, choose Resource Management > Node Pools.

**Step 2** In the upper right corner of the displayed page, select a cluster to filter node pools by cluster.

**Step 3** Click Configuration next to the node pool name.

**Step 4** On the Configuration page on the right, change the values of the following Kubernetes parameters:

<table>
<thead>
<tr>
<th>Component</th>
<th>Parameter</th>
<th>Description</th>
<th>Default Value</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>docker</td>
<td>native-umask</td>
<td><code>--exec-opt native.umask</code></td>
<td>normal</td>
<td>Cannot be changed.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>docker</td>
<td>base-size</td>
<td><code>--storage-opts dm.basesize</code></td>
<td>10G</td>
<td>Cannot be changed.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>insecure-registry</td>
<td></td>
<td>Address of an insecure image registry</td>
<td>false</td>
<td>Cannot be changed.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>limitcore</td>
<td></td>
<td>Limit on the number of cores</td>
<td>5368709</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>120</td>
<td></td>
</tr>
<tr>
<td></td>
<td>default-ulimit-nofile</td>
<td>Limit on the number of handles in a container</td>
<td>{soft}:</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>{hard}</td>
<td></td>
</tr>
<tr>
<td>kube-proxy</td>
<td>conntrack-min</td>
<td>sysctl -w net.nf_conntrack_max</td>
<td>131072</td>
<td>The values can be modified during</td>
</tr>
<tr>
<td>Component</td>
<td>Parameter</td>
<td>Description</td>
<td>Default Value</td>
<td>Remarks</td>
</tr>
<tr>
<td>-----------</td>
<td>-----------</td>
<td>-------------</td>
<td>---------------</td>
<td>---------</td>
</tr>
<tr>
<td>conntrack-tcp-timeout-close-wait</td>
<td>sysctl -w net.netfilter.nf_conntrack_tcp_timeout_close_wait</td>
<td></td>
<td>1h0m0s</td>
<td>the node pool lifecycle.</td>
</tr>
<tr>
<td>kubelet</td>
<td>cpu-manager-policy</td>
<td>`--cpu-manager-policy</td>
<td>none</td>
<td>The values can be modified during the node pool lifecycle.</td>
</tr>
<tr>
<td>kube-api-qps</td>
<td>Query per second (QPS) to use while talking with kube-apiserver.</td>
<td>100</td>
<td></td>
<td></td>
</tr>
<tr>
<td>kube-api-burst</td>
<td>Burst to use while talking with kube-apiserver.</td>
<td>100</td>
<td></td>
<td></td>
</tr>
<tr>
<td>max-pods</td>
<td>Maximum number of pods managed by kubelet.</td>
<td>110</td>
<td></td>
<td></td>
</tr>
<tr>
<td>pod-pids-limit</td>
<td>PID limit in Kubernetes</td>
<td>-1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>with-local-dns</td>
<td>Whether to use the local IP address as the ClusterDNS of the node.</td>
<td>false</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>allowed-unsafe-sysctls</td>
<td>Insecure system configuration allowed. Starting from v1.17.17, CCE enables pod security policies for kube-apiserver. You need to add corresponding configurations to <code>allowedUnsafeSysctls</code> of a pod security policy to make the policy take effect. (This configuration is not required for clusters earlier than v1.17.17.) For details, see Example of Enabling Unsafe Sysctls in Pod Security Policy.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Step 5** Click OK.

----End
Editing a Node Pool

**Step 1** Log in to the CCE console. In the navigation pane, choose **Resource Management > Node Pools.**

**Step 2** In the upper right corner of the displayed page, select a cluster to filter node pools by cluster.

**Step 3** Click **Edit** next to the name of the node pool you will edit. In the **Edit Node Pool** dialog box, edit the following parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Name of the node pool.</td>
</tr>
<tr>
<td>Nodes</td>
<td>Modify the number of nodes based on service requirements.</td>
</tr>
<tr>
<td>Autoscaler</td>
<td>By default, autoscaler is disabled. After you enable autoscaler by clicking , nodes in the node pool are automatically created or deleted based on service requirements.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Maximum Nodes</strong> and <strong>Minimum Nodes</strong>: You can set the maximum and minimum number of nodes to ensure that the number of nodes to be scaled is within a proper range.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Priority</strong>: A larger value indicates a higher priority. For example, if this parameter is set to 1 and 4 respectively for node pools A and B, B has a higher priority than A, and auto scaling is first triggered for B. If the priorities of multiple node pools are set to the same value, for example, 2, the node pools are not prioritized and the system performs scaling based on the minimum resource waste principle.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Scaling-In Cooling Interval</strong>: Set this parameter in the unit of minute. This field indicates the period during which the nodes added in the current node pool cannot be scaled in.</td>
</tr>
<tr>
<td></td>
<td>If the <strong>Autoscaler</strong> field is set to on, install the autoscaler add-on to use the autoscaler feature.</td>
</tr>
<tr>
<td>Edit Key pair</td>
<td>By default, editing key pair is disabled. After you enable this function by clicking , you can select an existing key pair or create a key pair. A key pair is used for identity authentication when you remotely log in to a node.</td>
</tr>
</tbody>
</table>

(Available only in the Singapore region)
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| Taints      | ● This field is left blank by default. Taints allow nodes to repel a set of pods. You can add a maximum of 10 taints for each node pool. Each taint contains the following parameters:  
  - **Key**: A key must contain 1 to 63 characters starting with a letter or digit. Only letters, digits, hyphens (-), underscores (_), and periods (.) are allowed. A DNS subdomain name can be used as the prefix of a key.  
  - **Value**: A value must start with a letter or digit and can contain a maximum of 63 characters, including letters, digits, hyphens (-), underscores (_), and periods (.).  
  - **Effect**: Available options are `NoSchedule`, `PreferNoSchedule`, and `NoExecute`.  
  **NOTICE** If taints are used, you must configure tolerations in the YAML files of pods. Otherwise, scale-up may fail or pods cannot be scheduled onto the added nodes. |
| K8S Labels  | K8S labels are key/value pairs that are attached to objects, such as pods. Labels are used to specify identifying attributes of objects that are meaningful and relevant to users, but do not directly imply semantics to the core system. For more information, see [Labels and Selectors](#). |
| Resource Tags | It is recommended that you use TMS's predefined tag function to add the same tag to different cloud resources. Predefined tags are visible to all service resources that support the tagging function. You can use predefined tags to improve tag creation and migration efficiency. Tag changes do not affect the node. |

**Step 4** After the configuration is complete, click **Save**.

In the node pool list, the node pool status becomes **Scaling**. After the status changes to **Completed**, the node pool parameters are modified successfully. The modified configuration will be synchronized to all nodes in the node pool.

---End

**Deleting a Node Pool**

Deleting a node pool will delete nodes in the pool. Pods on these nodes will be automatically migrated to available nodes in other node pools. If pods in the node pool have a specific node selector and none of the other nodes in the cluster satisfies the node selector, the pods will become unschedulable.
Step 1 Log in to the CCE console. In the navigation pane, choose Resource Management > Node Pools.

Step 2 In the upper right corner of the displayed page, select a cluster to filter node pools by cluster.

Step 3 Choose More > Delete next to a node pool name to delete the node pool.

Step 4 Read the precautions in the Delete Node Pool dialog box.

Step 5 Enter DELETE in the text box and click Yes to confirm that you want to continue the deletion.

--- End

Copying a Node Pool

You can copy the configuration of an existing node pool to create a new node pool on the CCE console.

Step 1 Log in to the CCE console. In the navigation pane, choose Resource Management > Node Pools.

Step 2 In the upper right corner of the displayed page, select a cluster to filter node pools by cluster.

Step 3 Choose More > Copy next to a node pool name to copy the node pool.

Step 4 The configuration of the selected node pool is replicated to the Create Node Pool page. You can edit the configuration as required and click Next: Confirm.

Step 5 On the Confirm page, confirm the node pool configuration and click Create Now. Then, a new node pool is created based on the edited configuration.

--- End

Migrating a Node

Nodes in a node pool can be migrated. Currently, nodes in a node pool can be migrated only to the default node pool (defaultpool) in the same cluster.

Step 1 Log in to the CCE console. In the navigation pane, choose Resource Management > Node Pools.

Step 2 In the upper right corner of the displayed page, select a cluster to filter node pools by cluster.

Step 3 Click More > Migrate next to the name of the node pool.

Step 4 In the dialog box displayed, select the destination node pool and the node to be migrated.

NOTE

After node migration, original resource tags, Kubernetes labels, and taints will be retained, and new Kubernetes labels and taints from the destination node pool will be added.
Step 5  Click OK.

----End
6 Workloads

6.1 Overview

A workload is an application running on Kubernetes. No matter how many components are there in your workload, you can run it in a group of Kubernetes pods. A workload is an abstract model of a group of pods in Kubernetes. Workloads classified in Kubernetes include Deployments, StatefulSets, DaemonSets, jobs, and cron jobs.

CCE provides Kubernetes-native container deployment and management and supports lifecycle management of container workloads, including creation, configuration, monitoring, auto scaling, upgrade, uninstall, service discovery, and load balancing.

Basic Concepts

- **Deployment**: Pods are completely independent of each other and functionally identical. They feature auto scaling and rolling upgrade. Typical examples include Nginx and WordPress. For details on how to create a Deployment, see Creating a Deployment.

- **StatefulSet**: Pods are not completely independent of each other. They have stable persistent storage, and feature orderly deployment and deletion. Typical examples include MySQL-HA and etcd. For details on how to create a StatefulSet, see Creating a StatefulSet.

- **DaemonSet**: A DaemonSet ensures that all or some nodes run a pod. It is applicable to pods running on every node. Typical examples include Ceph, Fluentd, and Prometheus Node Exporter. For details about how to create a DaemonSet, see Creating a DaemonSet.

- **Job**: A job is a one-time task that runs to completion. It can be executed immediately after being created. Before creating a workload, you can execute a job to upload an image to the image repository. For details about how to create a job, see Creating a Job.

- **Cron job**: A cron job runs periodically on a given schedule. You can perform time synchronization for all active nodes at a fixed time point. For details about how to create a cron job, see Creating a Cron Job.
Relationship Between Workloads and Containers

As shown in Figure 6-1, a workload controls one or more pods. A pod consists of one or more containers. Each container is created from a container image. Pods of Deployments are exactly the same.

Figure 6-1 Relationship between workloads and containers

Workload Lifecycle

Table 6-1 Status description

<table>
<thead>
<tr>
<th>Status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Running</td>
<td>All pods are running.</td>
</tr>
<tr>
<td>Unready</td>
<td>A container is abnormal, the number of pods is 0, or the workload is in pending state.</td>
</tr>
<tr>
<td>Upgrading/Rolling back</td>
<td>The workload is being upgraded or rolled back.</td>
</tr>
<tr>
<td>Available</td>
<td>For a multi-pod Deployment, some pods are abnormal but at least one pod is available.</td>
</tr>
<tr>
<td>Completed</td>
<td>The task is successfully executed. This status is available only for common tasks.</td>
</tr>
<tr>
<td>Stopped</td>
<td>The workload is stopped and the number of pods changes to 0. This status is available for workloads earlier than v1.13.</td>
</tr>
<tr>
<td>Deleting</td>
<td>The workload is being deleted.</td>
</tr>
<tr>
<td>Pausing</td>
<td>The workload is being paused.</td>
</tr>
</tbody>
</table>

Related Operations

The Kubernetes pod structure does not contain `ExtendPathMode`. Therefore, when you use client-go to call the API for creating a pod or Deployment, the
created pod does not contain **ExtendPathMode**. CCE provides a solution to ensure compatibility with the Kubernetes client-go. For details about the solution, see *How Can I Achieve Compatibility Between ExtendPathMode and Kubernetes client-go?*

### 6.2 Creating a Deployment

**Scenario**

Deployments are workloads (for example, Nginx) that do not store any data or status. You can create Deployments on the CCE console or by running kubectl commands.

**Prerequisites**

- Before creating a containerized workload, you must have an available cluster. For details on how to create a cluster, see *Buying a CCE Cluster*.
- To enable access to a workload from a public network, ensure that an elastic IP address (EIP) has been bound to or a load balancer has been configured for at least one node in the cluster.

 **NOTE**

When creating multiple workloads, ensure that each container has a unique port. Otherwise, workload creation will fail.

**Using the CCE Console**

CCE provides multiple methods for creating a workload. You can use any of the following methods:

- Use an image in **Open Source Images**. You do not need to upload any image before using it.
- Use an image that you have uploaded to SWR. For details, see **Image Management**.
- Use a **shared image** to create a workload. Specifically, other tenants share an image with you by using the **SWR service**.
- Use a YAML file to create a workload. You can click **Create YAML** on the right of the **Configure Advanced Settings** page when creating a Deployment. For details about YAML, see **Table 6-4**. After the YAML file is written, click **Create** to create a workload.

 **NOTE**

Settings in the YAML file are synchronized with those on the console. You can edit the YAML file on the console to create a workload. For example:

- If you enter a workload name on the console, the name will automatically appear in the YAML file.
- If you add an image on the console, the image will be automatically added to the YAML file.

When you click **Create YAML** on the right of the console, do not create multiple YAML files in the YAML definition pane displayed. You need to create them one by one. Otherwise, an error will be reported during the creation.
Step 1  Log in to the CCE console. In the navigation pane, choose Workloads > Deployments. On the page displayed, click Create Deployment. Set basic workload parameters as described in Table 6-2. The parameters marked with an asterisk (*) are mandatory.

<table>
<thead>
<tr>
<th>Table 6-2</th>
<th>Basic workload parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>* Workload Name</td>
<td>Name of the workload to be created. The name must be unique. Enter 4 to 63 characters starting with a letter and ending with a letter or digit. Only lowercase letters, digits, and hyphens (-) are allowed.</td>
</tr>
<tr>
<td>* Cluster Name</td>
<td>Cluster in which the workload resides.</td>
</tr>
<tr>
<td>* Namespace</td>
<td>In a single cluster, data in different namespaces is isolated from each other. This enables applications to share the services of the same cluster without interfering each other. If no namespace is set, the default namespace is used.</td>
</tr>
<tr>
<td>* Instances</td>
<td>Number of pods in the workload. A workload can have one or more pods. You can set the number of pods. The default value is 2 and can be set to 1. Each workload pod consists of the same containers. Configuring multiple pods for a workload ensures that the workload can still run properly even if a pod is faulty. If only one pod is used, a node or pod exception may cause service exceptions.</td>
</tr>
<tr>
<td>Time Zone Synchronization</td>
<td>If this parameter is enabled, the container and the node use the same time zone. NOTICE After time zone synchronization is enabled, disks of the hostPath type will be automatically added and listed in the Data Storage &gt; Local Volume area. Do not modify or delete the disks.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the workload.</td>
</tr>
</tbody>
</table>

Step 2  Click Next: Add Container.

1. Click Add Container and select the image to be deployed.
   - **My Images**: Create a workload using an image in the image repository you created.
   - **Open Source Images**: Create a workload using an image in an open source image registry.
   - **Third-Party Images**: Create a workload using an image from any third-party image repository (image repositories other than SWR and open source image registries). When you create a workload using a third-party image, ensure that the node where the workload is running can access public networks. For details on how to create a workload using a third-party image, see Using a Third-Party Image.
If your image repository does not require authentication, set Secret Authentication to No, enter an image pull address, and then click OK.

If your image repository must be authenticated (account and password), you need to create a secret and then use a third-party image. For details, see Using a Third-Party Image.

- **Shared Images**: Create a workload using an image shared by another tenant through the SWR service.

2. Configure basic image information.

A workload is an abstract model of a group of pods. One pod can encapsulate one or more containers. You can click Add Container in the upper right corner to add multiple container images and set them separately.

**Table 6-3 Image parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Image Name</td>
<td>Name of the image. You can click Change Image to update it.</td>
</tr>
<tr>
<td>*Image Version</td>
<td>Select the image tag to be deployed.</td>
</tr>
<tr>
<td>*Container Name</td>
<td>Name of the container. You can modify it.</td>
</tr>
<tr>
<td>Privileged Container</td>
<td>Programs in a privileged container have certain privileges. If Privileged Container is On, the container is granted superuser permissions. For example, privileged containers can manipulate network devices on the host machine and modify kernel parameters.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
| Container Resources | **CPU**  
  - **Request**: minimum number of CPU cores required by a container. The default value is 0.25 cores.  
  - **Limit**: maximum number of CPU cores available for a container. Do not leave Limit unspecified. Otherwise, intensive use of container resources will occur and your workload may exhibit unexpected behavior.  

**Memory**  
  - **Request**: minimum amount of memory required by a container. The default value is 512 MiB.  
  - **Limit**: maximum amount of memory available for a container. When memory usage exceeds the specified memory limit, the container will be terminated.  

For more information about Request and Limit, see Setting Container Specifications.  

**GPU**: configurable only when the cluster contains GPU nodes.  
It indicates the percentage of GPU resources reserved for a container. Select **Use** and set the percentage. For example, if this parameter is set to 10%, the container is allowed to use 10% of GPU resources. If you do not select Use or set this parameter to 0, no GPU resources can be used.  

**GPU/Graphics Card**: The workload's pods will be scheduled to the node with the specified GPU.  
If Any GPU type is selected, the container uses a random GPU in the node. If you select a specific GPU, the container uses this GPU accordingly.  

**Ascend 310 Quota**: number of Ascend 310 processors requested by a container. The value must be an integer.  
The settings of this parameter take effect only if an Ascend-accelerated node is selected and the huawei-npu add-on is installed. Ascend-accelerated nodes are available for open beta testing (OBT) now. This type of nodes, powered by HiSilicon Ascend 310 AI processors, are suitable for image recognition, video processing, AI inference, machine learning, and other scenarios that require high performance and low power consumption. Click here to apply for OBT now.  

3. **Lifecycle**: Commands for starting and running containers can be set.  
   - **Start Command**: executed when the workload is started. For details, see Setting Container Startup Commands.  
   - **Post-Start**: executed after the workload runs successfully. For more information, see Setting Container Lifecycle Parameters.
- **Pre-Stop**: executed to delete logs or temporary files before the workload ends. For more information, see Setting Container Lifecycle Parameters.

4. **Health Check**: CCE provides two types of probes: liveness probe and readiness probe. They are used to determine whether containers and user services are running properly. For more information, see Setting Health Check for a Container.
   - **Liveness Probe**: used to restart the unhealthy container.
   - **Readiness Probe**: used to change the container to the unready state when detecting that the container is unhealthy. In this way, service traffic will not be directed to the container.

5. **Environment Variables**: Environment variables can be added to a container. In general, environment variables are used to set parameters. On the Environment Variables tab page, click Add Environment Variable. Currently, three types of environment variables are supported:
   - **Added manually**: Set Variable Name and Variable Value/Reference.
   - **Added from Secret**: Set Variable Name and select the desired secret name and data. A secret must be created in advance. For details, see Creating a Secret.
   - **Added from ConfigMap**: Set Variable Name and select the desired ConfigMap name and data. A ConfigMap must be created in advance. For details, see Creating a ConfigMap.

   **NOTE**
   To edit an environment variable that has been set, click Edit. To delete an environment variable that has been set, click Delete.

6. **Data Storage**: Data storage can be mounted to containers for persistent storage and high disk I/O. Local volume and cloud storage are supported. For details, see Storage (CSI).

7. **Security Context**: Container permissions can be configured to protect CCE and other containers from being affected.
   Enter the user ID to set container permissions and prevent systems and other containers from being affected.

8. **Log Policies**: Log collection policies and log directory can be configured to collect container logs for unified management and analysis. For details, see Collecting Default Standard Output Container Logs and Collecting Container Logs from Mount Paths.

**Step 3** Click Next: Set Application Access. Then, click Add Service and set the workload access type.

If your workload will be reachable to other workloads or public networks, add a Service to define the workload access type.

The workload access type determines the network attributes of the workload. Workloads with different access types can provide different network capabilities. For details, see Overview.

**Step 4** Click Next: Configure Advanced Settings to configure advanced policies.

- **Upgrade Mode**: You can specify the upgrade mode of a Deployment, including Rolling upgrade and In-place upgrade.
- **Rolling upgrade**: Old pods are gradually replaced with new ones. During the upgrade, service traffic is evenly distributed to both pods to ensure service continuity.
  
  - **Maximum Number of Unavailable Pods**: maximum number of unavailable pods allowed in a rolling upgrade. If the number is equal to the total number of pods, services may be interrupted. Minimum number of alive pods = Total pods – Maximum number of unavailable pods

- **In-place upgrade**: Old pods are deleted before new pods are created. Services will be interrupted during an in-place upgrade.

- **Graceful Deletion**: A time window can be set for workload deletion and reserved for executing commands in the pre-stop phase in the lifecycle. If workload processes are not terminated after the time window elapses, the workload will be forcibly deleted.
  
  - **Graceful Time Window (s)**: Set a time window (0–9999s) for pre-stop commands to finish execution before a workload is deleted. The default value is 30s.

- **Scale Order**: Choose **Prioritize new pods** or **Prioritize old pods** based on service requirements. **Prioritize new pods** indicates that new pods will be first deleted when a scale-in is triggered.

- **Migration Policy**: When the node where a workload's pods are located is unavailable for the specified amount of time, the pods will be rescheduled to other available nodes.
  
  - **Migration Time Window (s)**: Set a time window for migration. The default value is 300s.

- **Scheduling Policies**: You can combine static global scheduling policies or dynamic runtime scheduling policies as required. For details, see [Scheduling Policy Overview](#).

- **Advanced Pod Settings**
  
  - **Pod Label**: The built-in **app** label is specified when the workload is created. It is used to set affinity and anti-affinity scheduling and cannot be modified. You can click **Add Label** to add labels.

  ![Advanced pod settings](#)

- **Client DNS Configuration**: A CCE cluster has a built-in DNS add-on (CoreDNS) to provide domain name resolution for workloads in the cluster. For details, see [Using Kubernetes In-Cluster DNS](#).
  
  - **DNS Policy**
    
    - **ClusterFirst**: The default DNS configuration overrides the **Nameserver** and **DNS Search Domain** configurations of the client.
- **None**: Only the Nameserver and DNS Search Domain configurations are used for domain name resolution.

- **Default**: The pod inherits the DNS configuration from the node on which the pod runs.
  - **Nameserver**: You can configure a domain name server for a user-defined domain name. The value is one or a group of DNS IP addresses, for example, 1.2.3.4.
  - **DNS Search Domain**: a search list for host-name lookup. When a domain name cannot be resolved, DNS queries will be attempted combining the domain name with each domain in the search list in turn until a match is found or all domains in the search list are tried.
  - **Timeout (s)**: amount of time the resolver will wait for a response from a remote name server before retrying the query on a different name server. Set it based on the site requirements.
  - **ndots**: threshold for the number of dots that must appear in a domain name before an initial absolute query will be made. If a domain name has 
  ndots or more than ndots dots, the name is a fully qualified domain name (FQDN) and will be tried first as an absolute name. If a domain name has less than ndots dots, the operating system will look up the name in a list of search domain names.

- **Custom Monitoring**: You can specify the metrics to be collected and the path and port from which the monitoring system will regularly collect the specified metrics. For details, see Working with Prometheus for Custom Monitoring.

- **APM Settings**: APM helps you quickly locate workload problems and identify performance bottlenecks to improve user experience. For details, see Configuring APM Settings for Performance Bottleneck Analysis.

**Step 5** After the preceding configurations are complete, click **Create**. On the page displayed, click **Return to Workload List** to view the workload status.

If the workload is in the **Running** state, it has been successfully created.

Workload status is not updated in real time. Click in the upper right corner or press F5 to refresh the page.

**Step 6** To access the workload in a browser, go to the workload list on the **Deployments** page. Copy the corresponding **External Access Address** and paste it into the address box in the browser.

**NOTE**

- External access addresses are available only if the Deployment access type is set to **NodePort** and an EIP is assigned to any node in the cluster, or if the Deployment access type is set to **LoadBalancer (ELB)**.

- If the workload list contains more than 500 records, the Kubernetes pagination mechanism will be used. Specifically, you can only go to the first page or the next page, but cannot go to the previous page. In addition, if resources are divided into discrete pages, the total number of resources displayed is the maximum number of resources that can be queried at a time, not the actual total number of resources.
Using kubectl

The following procedure uses Nginx as an example to describe how to create a workload using kubectl.

**Step 1** Use kubectl to connect to the cluster. For details, see [Connecting to a Cluster Using kubectl](#).

**Step 2** Create and edit the `nginx-deployment.yaml` file. `nginx-deployment.yaml` is an example file name. You can rename it as required.

```yaml
vi nginx-deployment.yaml
```

The following is an example YAML file. For more information about Deployments, see [Kubernetes documentation](#).

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx
spec:
  replicas: 1
  selector:
    matchLabels:
      app: nginx
  strategy:
    type: RollingUpdate
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
        - image: nginx  # If you use an image from an open-source image registry, enter the image name. If you use an image in My Images, obtain the image path from SWR.
          imagePullPolicy: Always
          name: nginx
          imagePullSecrets:
            - name: default-secret
```

For details about these parameters, see [Table 6-4](#).

### Table 6-4 Deployment YAML parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Mandatory/Optional</th>
</tr>
</thead>
<tbody>
<tr>
<td>apiVersion</td>
<td>API version.</td>
<td>Mandatory</td>
</tr>
<tr>
<td></td>
<td><strong>NOTE</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Set this parameter based on the cluster version.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• For clusters of v1.17 or later, the apiVersion format of Deployments is</td>
<td></td>
</tr>
<tr>
<td></td>
<td>apps/v1.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• For clusters of v1.15 or earlier, the apiVersion format of Deployments is</td>
<td></td>
</tr>
<tr>
<td></td>
<td>extensions/v1beta1</td>
<td></td>
</tr>
<tr>
<td>kind</td>
<td>Type of a created object.</td>
<td>Mandatory</td>
</tr>
<tr>
<td>metadata</td>
<td>Metadata of a resource object.</td>
<td>Mandatory</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Mandatory/Optional</td>
</tr>
<tr>
<td>-------------</td>
<td>------------------------------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td>name</td>
<td>Name of the Deployment.</td>
<td>Mandatory</td>
</tr>
<tr>
<td>Spec</td>
<td>Detailed description of the Deployment.</td>
<td>Mandatory</td>
</tr>
<tr>
<td>replicas</td>
<td>Number of pods.</td>
<td>Mandatory</td>
</tr>
<tr>
<td>selector</td>
<td>Determines container pods that can be managed by the Deployment.</td>
<td>Mandatory</td>
</tr>
<tr>
<td>strategy</td>
<td>Upgrade mode. Possible values:</td>
<td>Optional</td>
</tr>
<tr>
<td></td>
<td>● RollingUpdate</td>
<td></td>
</tr>
<tr>
<td></td>
<td>● ReplaceUpdate</td>
<td></td>
</tr>
<tr>
<td></td>
<td>By default, rolling update is used.</td>
<td></td>
</tr>
<tr>
<td>template</td>
<td>Detailed description of a created container pod.</td>
<td>Mandatory</td>
</tr>
<tr>
<td>metadata</td>
<td>Metadata.</td>
<td>Mandatory</td>
</tr>
<tr>
<td>labels</td>
<td>metadata.labels: Container labels.</td>
<td>Optional</td>
</tr>
<tr>
<td>spec:</td>
<td></td>
<td>Mandatory</td>
</tr>
<tr>
<td>containers</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>● image (mandatory): Name of a container image.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>● imagePullPolicy (optional): Policy for obtaining an image. The options include Always (attempting to download images each time), Never (only using local images), and IfNotPresent (using local images if they are available; downloading images if local images are unavailable). The default value is Always.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>● name (mandatory): Container name.</td>
<td></td>
</tr>
<tr>
<td>imagePullSecrets</td>
<td>Name of the secret used during image pulling. If a private image is used, this parameter is mandatory.</td>
<td>Optional</td>
</tr>
<tr>
<td></td>
<td>● To pull an image from the Software Repository for Container (SWR), set this parameter to default-secret.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>● To pull an image from a third-party image repository, set this parameter to the name of the created secret.</td>
<td></td>
</tr>
</tbody>
</table>
Step 3 Create a Deployment.

```
kubectl create -f nginx-deployment.yaml
```

If the following information is displayed, the Deployment is being created.

```
deployment "nginx" created
```

Step 4 Query the Deployment status.

```
kubectl get pods
```

If the following information is displayed, the Deployment is running.

<table>
<thead>
<tr>
<th>NAME</th>
<th>READY</th>
<th>STATUS</th>
<th>RESTARTS</th>
<th>AGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>icagent-m9dkt</td>
<td>0/0</td>
<td>Running</td>
<td>0</td>
<td>3d</td>
</tr>
<tr>
<td>nginx-1212400781-qv313</td>
<td>1/1</td>
<td>Running</td>
<td>0</td>
<td>3d</td>
</tr>
</tbody>
</table>

**Parameter description**

- **NAME**: pod name
- **READY**: number of pod replicas that have been deployed
- **STATUS**: status of the Deployment
- **RESTARTS**: restart times
- **AGE**: period the Deployment keeps running

Step 5 If the Deployment will be accessed through a ClusterIP or NodePort Service, add the corresponding Service. For details, see **Networking**.

----End

### 6.3 Creating a StatefulSet

**Scenario**

StatefulSets are a type of workloads whose data or status is stored while they are running. For example, MySQL is a StatefulSet because it needs to store new data.

A container can be migrated between different hosts, but data is not stored on the hosts. To store StatefulSet data persistently, attach HA storage volumes provided by CCE to the container.

**Prerequisites**

- Before creating a workload, you must have an available cluster. For details on how to create a cluster, see **Buying a CCE Cluster**.
- To enable access to a workload from a public network, ensure that an elastic IP address (EIP) has been bound to or a load balancer has been configured for at least one node in the cluster.

**NOTE**

When creating multiple workloads, ensure that each container has a unique port. Otherwise, workload creation will fail.
Using the CCE Console

CCE provides multiple methods for creating a workload. You can use any of the following methods:

1. Use an image in Open Source Images. You do not need to upload any image before using it.
2. Use an image that you have uploaded to SWR. For details, see Image Management.
3. Use a shared image to create a workload. Specifically, other tenants share an image with you by using the SWR service.
4. Use a YAML file to create a workload. You can click Create YAML on the right of the Create StatefulSet page. For details about YAML, see Using kubectl. After the YAML file is written, click Create to create a workload.

**NOTE**

Settings in the YAML file are synchronized with those on the console. You can edit the YAML file on the console to create a workload. For example:

- If you enter a workload name on the console, the name will automatically appear in the YAML file.
- If you add an image on the console, the image will be automatically added to the YAML file.

When you click Create YAML on the right of the console, do not create multiple YAML files in the YAML definition pane displayed. You need to create them one by one. Otherwise, an error will be reported during the creation.

**Step 1** Log in to the CCE console. In the navigation pane, choose Workloads > StatefulSets. On the displayed page, click Create StatefulSet. Set basic workload parameters as described in Table 6-5. The parameters marked with an asterisk (*) are mandatory.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>* Workload Name</td>
<td>Name of a workload, which must be unique. Enter 4 to 52 characters starting with a lowercase letter and ending with a letter or digit. Only lowercase letters, digits, and hyphens (-) are allowed.</td>
</tr>
<tr>
<td>* Cluster Name</td>
<td>Cluster to which the workload belongs.</td>
</tr>
<tr>
<td>* Namespace</td>
<td>In a single cluster, data in different namespaces is isolated from each other. This enables applications to share the services of the same cluster without interfering each other. If no namespace is set, the default namespace is used.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>Instances</strong></td>
<td>Number of pods in a workload. A workload can have one or more pods. The default value is 2. You can customize the value, for example, setting it to 1. Each workload pod consists of the same containers. You can configure multiple pods for a workload to ensure high reliability. For such a workload, if one pod is faulty, the workload can still run properly. If only one pod is used, a node or pod exception may cause service exceptions.</td>
</tr>
</tbody>
</table>
| **Time Zone Synchronization** | If this parameter is enabled, the container and the node use the same time zone.  
**NOTICE**  
After time zone synchronization is enabled, disks of the hostPath type will be automatically added and listed in the Data Storage > Local Volume area. Do not modify or delete the disks. |
| **Description**          | Description of the workload.                                                                                                                                                                               |

**Step 2**  
Click Next: Add Container.  
1. Click Add Container and select the image to be deployed.  
   - **My Images**: Create a workload using an image in the image repository you created.  
   - **Open Source Images**: Create a workload using an image in an open source image registry.  
   - **Third-Party Images**: Create a workload using an image from any third-party image repository (image repositories other than SWR and open source image registries). When you create a workload using a third-party image, ensure that the node where the workload is running can access public networks. For details on how to create a workload using a third-party image, see Using a Third-Party Image.  
     - If your image repository does not require authentication, set Secret Authentication to No, enter an image pull address, and then click OK.  
     - If your image repository must be authenticated (account and password), you need to create a secret and then use a third-party image. For details, see Using a Third-Party Image.  
   - **Shared Images**: Create a workload using an image shared by another tenant through the SWR service.  
2. Configure basic image information.  
   A workload is an abstract model of a group of pods. One pod can encapsulate one or more containers. You can click Add Container in the upper right corner to add multiple container images and set them separately.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Image Name</td>
<td>Name of the image. You can click Change Image to update it.</td>
</tr>
<tr>
<td>*Image Version</td>
<td>Select the image tag to be deployed.</td>
</tr>
<tr>
<td>*Container Name</td>
<td>Name of the container. You can modify it.</td>
</tr>
<tr>
<td>Privileged Container</td>
<td>Programs in a privileged container have certain privileges. If Privileged Container is On, the container is granted superuser permissions. For example, privileged containers can manipulate network devices on the host machine and modify kernel parameters.</td>
</tr>
</tbody>
</table>
### Container Resources

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CPU</strong></td>
<td>- <strong>Request</strong>: minimum number of CPU cores required by a container. The default value is 0.25 cores.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Limit</strong>: maximum number of CPU cores available for a container. Do not leave <strong>Limit</strong> unspecified. Otherwise, intensive use of container resources will occur and your workload may exhibit unexpected behavior.</td>
</tr>
<tr>
<td><strong>Memory</strong></td>
<td>- <strong>Request</strong>: minimum amount of memory required by a container. The default value is 512 MiB.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Limit</strong>: maximum amount of memory available for a container. When memory usage exceeds the specified memory limit, the container will be terminated.</td>
</tr>
<tr>
<td><strong>GPU</strong></td>
<td>configurable only when the cluster contains GPU nodes.</td>
</tr>
<tr>
<td></td>
<td>It indicates the percentage of GPU resources reserved for a container. Select <strong>Use</strong> and set the percentage. For example, if this parameter is set to 10%, the container is allowed to use 10% of GPU resources. If you do not select <strong>Use</strong> or set this parameter to 0, no GPU resources can be used.</td>
</tr>
<tr>
<td><strong>GPU/Graphics Card</strong></td>
<td>The workload's pods will be scheduled to the node with the specified GPU.</td>
</tr>
<tr>
<td></td>
<td>If <strong>Any GPU type</strong> is selected, the container uses a random GPU in the node. If you select a specific GPU, the container uses this GPU accordingly.</td>
</tr>
<tr>
<td><strong>Ascend 310 Quota</strong></td>
<td>number of Ascend 310 processors requested by a container. The value must be an integer.</td>
</tr>
<tr>
<td></td>
<td>The settings of this parameter take effect only if an Ascend-accelerated node is selected and the huawei-npu add-on is installed. Ascend-accelerated nodes are available for open beta testing (OBT) now. This type of nodes, powered by HiSilicon Ascend 310 AI processors, are suitable for image recognition, video processing, AI inference, machine learning, and other scenarios that require high performance and low power consumption. Click here to apply for OBT now.</td>
</tr>
</tbody>
</table>

3. **Lifecycle**: Commands for starting and running containers can be set.
   - **Start Command**: executed when the workload is started. For details, see [Setting Container Startup Commands](#).
   - **Post-Start**: executed after the workload runs successfully. For more information, see [Setting Container Lifecycle Parameters](#).
4. **Health Check**: CCE provides two types of probes: liveness probe and readiness probe. They are used to determine whether containers and user services are running properly. For more information, see Setting Health Check for a Container.
   - **Liveness Probe**: used to restart the unhealthy container.
   - **Readiness Probe**: used to change the container to the unready state when detecting that the container is unhealthy. In this way, service traffic will not be directed to the container.

5. **Environment Variables**: Environment variables can be added to a container. In general, environment variables are used to set parameters. On the Environment Variables tab page, click Add Environment Variable. Currently, three types of environment variables are supported:
   - **Added manually**: Set Variable Name and Variable Value/Reference.
   - **Added from Secret**: Set Variable Name and select the desired secret name and data. A secret must be created in advance. For details, see Creating a Secret.
   - **Added from ConfigMap**: Set Variable Name and select the desired ConfigMap name and data. A ConfigMap must be created in advance. For details, see Creating a ConfigMap.

   **NOTE**

   To edit an environment variable that has been set, click Edit. To delete an environment variable that has been set, click Delete.

6. **Data Storage**: Data storage can be mounted to containers for persistent storage and high disk I/O. Local disks and cloud storage volumes are supported. For details, see Storage (CSI).

   **NOTE**

   You can add data storage volumes only when creating a StatefulSet.

7. **Security Context**: Container permissions can be configured to protect CCE and other containers from being affected. Enter the user ID to set container permissions and prevent systems and other containers from being affected.

8. **Log Policies**: Log collection policies and log directory can be configured to collect container logs for unified management and analysis. For details, see Collecting Default Standard Output Container Logs and Collecting Container Logs from Mount Paths.

**Step 3** Click Next: Set Application Access and set Headless Service and workload access type.

Table 6-7 describes the parameters in the Headless Service area.
Table 6-7 Parameter description

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Service Name</td>
<td>Name of the Service corresponding to the workload for mutual access between pods. This Service is used for internal discovery of pods, and does not require an independent IP address or load balancing.</td>
</tr>
<tr>
<td>Port Name</td>
<td>Name of the container port. You are advised to enter a name that indicates the function of the port.</td>
</tr>
<tr>
<td>Container Port</td>
<td>Listening port inside the container.</td>
</tr>
</tbody>
</table>

Click **Add Service** and set the workload access type.

If your workload will be reachable to other workloads or public networks, add a Service to define the workload access type.

The workload access type determines the network attributes of the workload. Workloads with different access types can provide different network capabilities. For details, see **Overview**.

**Step 4** Click **Next: Configure Advanced Settings**.

- **Upgrade Policy**: Only **Rolling upgrade** is supported.
  
  During a rolling upgrade, old pods are gradually replaced with new ones, and service traffic is evenly distributed to both pods to ensure service continuity.

- **Pod Management Policy**: There are two types of policies: ordered and parallel.
  
  **Ordered**: The StatefulSet will deploy, delete, or scale pods in order and one by one (the StatefulSet waits until each pod is ready before continuing). This is the default policy.

  **Parallel**: The StatefulSet will create pods in parallel to match the desired scale without waiting, and will delete all pods at once.

- **Graceful Deletion**: A time window can be set for workload deletion and reserved for executing commands in the pre-stop phase in the lifecycle. If workload processes are not terminated after the time window elapses, the workload will be forcibly deleted.
  
  - **Graceful Time Window (s)**: Set a time window (0–9999s) for pre-stop commands to finish execution before a workload is deleted. The default value is 30s.

  - **Scale Order**: Choose **Prioritize new pods** or **Prioritize old pods** based on service requirements. **Prioritize new pods** indicates that new pods will be first deleted when a scale-in is triggered.

- **Scheduling Policies**: You can combine static global scheduling policies or dynamic runtime scheduling policies as required. For details, see **Scheduling Policy Overview**.

- **Advanced Pod Settings**
Pod Label: The built-in app label is specified when the workload is created. It is used to set affinity and anti-affinity scheduling and cannot be modified. You can click Add Label to add labels.

Figure 6-3 Advanced pod settings

Advanced Pod Settings

- Client DNS Configuration: A CCE cluster has a built-in DNS add-on (CoreDNS) to provide domain name resolution for workloads in the cluster. For details, see Using Kubernetes In-Cluster DNS.

- DNS Policy
  - ClusterFirst: The default DNS configuration overrides the Nameserver and DNS Search Domain configurations of the client.
  - None: Only the Nameserver and DNS Search Domain configurations are used for domain name resolution.
  - Default: The pod inherits the DNS configuration from the node on which the pod runs.

- Nameserver: You can configure a domain name server for a user-defined domain name. The value is one or a group of DNS IP addresses, for example, 1.2.3.4.

- DNS Search Domain: a search list for host-name lookup. When a domain name cannot be resolved, DNS queries will be attempted combining the domain name with each domain in the search list in turn until a match is found or all domains in the search list are tried.

- Timeout (s): amount of time the resolver will wait for a response from a remote name server before retrying the query on a different name server. Set it based on the site requirements.

- ndots: threshold for the number of dots that must appear in a domain name before an initial absolute query will be made. If a domain name has ndots or more than ndots dots, the name is a fully qualified domain name (FQDN) and will be tried first as an absolute name. If a domain name has less than ndots dots, the operating system will look up the name in a list of search domain names.

- Custom Monitoring: You can specify the metrics to be collected and the path and port from which the monitoring system will regularly collect the specified metrics. For details, see Working with Prometheus for Custom Monitoring.

- APM Settings: APM helps you quickly locate workload problems and identify performance bottlenecks to improve user experience. For details, see Configuring APM Settings for Performance Bottleneck Analysis.
Step 5  Click **Create** and then **Back to StatefulSet List**. If the workload is in the **Running** state, it has been successfully created. If the workload status is not updated, click in the upper right corner or press **F5** to refresh the page.

**NOTE**

- When a node is unavailable, pods become **Unready**. In this case, you need to manually delete the pods of the StatefulSet so that the pods can be migrated to a normal node.
- If the workload list contains more than 500 records, the Kubernetes pagination mechanism will be used. Specifically, you can only go to the first page or the next page, but cannot go to the previous page. In addition, if resources are divided into discrete pages, the total number of resources displayed is the maximum number of resources that can be queried at a time, not the actual total number of resources.

----End

Using kubectl

The following procedure uses an etcd workload as an example to describe how to create a workload using kubectl.

Step 1  Use kubectl to connect to the cluster. For details, see **Connecting to a Cluster Using kubectl**.

Step 2  Create and edit the **etcd-statefulset.yaml** file.

**etcd-statefulset.yaml** is an example file name, and you can change it as required.

```
vi etcd-statefulset.yaml
```

The following provides an example of the file contents. For more information on StatefulSet, see the **Kubernetes documentation**.

```
apiversion: apps/v1
kind: StatefulSet
metadata:
  name: etcd
spec:
  replicas: 2
  selector:
    matchLabels:
      app: etcd
  serviceName: etcd-svc
  template:
    metadata:
      labels:
        app: etcd
    spec:
      containers:
        - env:
          - name: PAAS_APP_NAME
            value: tesyhj
          - name: PAAS_NAMESPACE
            value: default
          - name: PAAS_PROJECT_ID
            value: 9632fae707ce4416a0ab1e3e121fe555
        image: etcd # If you use an image from an open-source image registry, enter the image name. If you use an image in **My Images**, obtain the image path from SWR.
        imagePullPolicy: IfNotPresent
        name: container-0
      updateStrategy:
        type: RollingUpdate
vi etcd-headless.yaml
```
Step 3  Create a workload and the corresponding headless service.

```sh
ekubectl create -f etcd-statefulset.yaml
```

If the following information is displayed, the StatefulSet has been successfully created.

```
statefulset.apps/etcd created
```

```sh
ekubectl create -f etcd-headless.yaml
```

If the following information is displayed, the headless service has been successfully created.

```
service/etcd-svc created
```

Step 4  If the workload will be accessed through a ClusterIP or NodePort Service, set the corresponding workload access type. For details, see Networking.

---End

### 6.4 Creating a DaemonSet

**Scenario**

CCE provides deployment and management capabilities for multiple types of containers and supports features of container workloads, including creation, configuration, monitoring, scaling, upgrade, uninstall, service discovery, and load balancing.

DaemonSet ensures that only one pod runs on all or some nodes. When a node is added to a cluster, a new pod is also added for the node. When a node is removed from a cluster, the pod is also reclaimed. If a DaemonSet is deleted, all pods created by it will be deleted.

The typical application scenarios of a DaemonSet are as follows:

- Run the cluster storage daemon, such as glusterd or Ceph, on each node.
- Run the log collection daemon, such as Fluentd or Logstash, on each node.
- Run the monitoring daemon, such as Prometheus Node Exporter, collectd, Datadog agent, New Relic agent, or Ganglia (gmond), on each node.
You can deploy a DaemonSet for each type of daemons on all nodes, or deploy multiple DaemonSets for the same type of daemons. In the second case, DaemonSets have different flags and different requirements on memory and CPU for different hardware types.

**Prerequisites**

You must have one cluster available before creating a DaemonSet. For details on how to create a cluster, see [Buying a CCE Cluster](#).

**Procedure**

**Step 1** Log in to the CCE console.

**Step 2** In the navigation pane on the left, choose **Workloads > DaemonSets**. Click **Create DaemonSet** in the upper right corner of the page. Set basic workload parameters as described in **Table 6-8**. The parameters marked with an asterisk (*) are mandatory.

### Table 6-8 Basic workload parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>* Workload Name</td>
<td>Name of the containerized workload to be created. The name must be unique. Enter 4 to 63 characters starting with a letter and ending with a letter or digit. Only lowercase letters, digits, and hyphens (-) are allowed.</td>
</tr>
<tr>
<td>* Cluster Name</td>
<td>Cluster to which the workload belongs.</td>
</tr>
<tr>
<td>* Namespace</td>
<td>In a single cluster, data in different namespaces is isolated from each other. This enables applications to share the services of the same cluster without interfering each other. If no namespace is set, the default namespace is used.</td>
</tr>
<tr>
<td>Time Zone</td>
<td>Synchronization  If this parameter is enabled, the container and the node use the same time zone.</td>
</tr>
<tr>
<td></td>
<td><strong>NOTICE</strong> After time zone synchronization is enabled, disks of the HostPath type will be automatically added and listed in the <strong>Data Storage &gt; Local Volume</strong> area. Do not modify or delete the disks.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the workload.</td>
</tr>
</tbody>
</table>

**Step 3** Click **Next: Add Container**.

1. Click **Add Container** and select the image to be deployed.
   - **My Images**: Create a workload using an image in the image repository you created.
   - **Open Source Images**: Create a workload using an image in an open source image registry.
   - **Third-Party Images**: Create a workload using an image from any third-party image repository (image repositories other than SWR and open Cloud Container Engine User Guide 6 Workloads Issue 01 (2021-09-08) Copyright © Huawei Technologies Co., Ltd. 188
source image registries). When you create a workload using a third-party image, ensure that the node where the workload is running can access public networks. For details on how to create a workload using a third-party image, see Using a Third-Party Image.

- If your image repository does not require authentication, set Secret Authentication to No, enter an image pull address, and then click OK.
- If your image repository must be authenticated (account and password), you need to create a secret and then use a third-party image. For details, see Using a Third-Party Image.

- Shared Images: Create a workload using an image shared by another tenant through the SWR service.

2. Configure basic image information.

A workload is an abstract model of a group of pods. One pod can encapsulate one or more containers. You can click Add Container in the upper right corner to add multiple container images and set them separately.

Table 6-9 Image parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Image Name</td>
<td>Name of the image. You can click Change Image to update it.</td>
</tr>
<tr>
<td>*Image Version</td>
<td>Select the image tag to be deployed.</td>
</tr>
<tr>
<td>*Container Name</td>
<td>Name of the container. You can modify it.</td>
</tr>
<tr>
<td>Privileged Container</td>
<td>Programs in a privileged container have certain privileges. If Privileged Container is On, the container is granted superuser permissions. For example, privileged containers can manipulate network devices on the host machine and modify kernel parameters.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Container</td>
<td>Resources</td>
</tr>
<tr>
<td>CPU</td>
<td>- <strong>Request</strong>: minimum number of CPU cores required by a container. The default value is 0.25 cores.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Limit</strong>: maximum number of CPU cores available for a container. Do not leave <strong>Limit</strong> unspecified. Otherwise, intensive use of container resources will occur and your workload may exhibit unexpected behavior.</td>
</tr>
<tr>
<td>Memory</td>
<td>- <strong>Request</strong>: minimum amount of memory required by a container. The default value is 512 MiB.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Limit</strong>: maximum amount of memory available for a container. When memory usage exceeds the specified memory limit, the container will be terminated.</td>
</tr>
<tr>
<td></td>
<td>For more information about <strong>Request</strong> and <strong>Limit</strong>, see <strong>Setting Container Specifications</strong>.</td>
</tr>
<tr>
<td>GPU</td>
<td>- <strong>Use</strong> and set the percentage. For example, if this parameter is set to 10%, the container is allowed to use 10% of GPU resources. If you do not select <strong>Use</strong> or set this parameter to 0, no GPU resources can be used.</td>
</tr>
<tr>
<td></td>
<td><strong>GPU/Graphics Card</strong>: The workload's pods will be scheduled to the node with the specified GPU.</td>
</tr>
<tr>
<td></td>
<td><strong>Ascend 310 Quota</strong>: number of Ascend 310 processors requested by a container. The value must be an integer.</td>
</tr>
<tr>
<td></td>
<td>The settings of this parameter take effect only if an Ascend-accelerated node is selected and the <strong>huawei-npu</strong> add-on is installed. Ascend-accelerated nodes are available for open beta testing (OBT) now. This type of nodes, powered by HiSilicon Ascend 310 AI processors, are suitable for image recognition, video processing, AI inference, machine learning, and other scenarios that require high performance and low power consumption. Click <strong>here</strong> to apply for OBT now.</td>
</tr>
</tbody>
</table>

3. **Lifecycle**: Commands for starting and running containers can be set.
   - **Start Command**: executed when the workload is started. For details, see **Setting Container Startup Commands**.
   - **Post-Start**: executed after the workload runs successfully. For more information, see **Setting Container Lifecycle Parameters**.
- **Pre-Stop**: executed to delete logs or temporary files before the workload ends. For more information, see Setting Container Lifecycle Parameters.

4. **Health Check**: CCE provides two types of probes: liveness probe and readiness probe. They are used to determine whether containers and user services are running properly. For more information, see Setting Health Check for a Container.
   - **Liveness Probe**: used to restart the unhealthy container.
   - **Readiness Probe**: used to change the container to the unready state when detecting that the container is unhealthy. In this way, service traffic will not be directed to the container.

5. **Environment Variables**: Environment variables can be added to a container. In general, environment variables are used to set parameters. On the Environment Variables tab page, click Add Environment Variable. Currently, three types of environment variables are supported:
   - **Added manually**: Set Variable Name and Variable Value/Reference.
   - **Added from Secret**: Set Variable Name and select the desired secret name and data. A secret must be created in advance. For details, see Creating a Secret.
   - **Added from ConfigMap**: Set Variable Name and select the desired ConfigMap name and data. A ConfigMap must be created in advance. For details, see Creating a ConfigMap.

■ **NOTE**

To edit an environment variable that has been set, click Edit. To delete an environment variable that has been set, click Delete.

6. **Data Storage**: Data storage can be mounted to containers for persistent storage and high disk I/O. Local volume and cloud storage are supported. For details, see Storage (CSI).

7. **Security Context**: Container permissions can be configured to protect CCE and other containers from being affected.

Enter the user ID to set container permissions and prevent systems and other containers from being affected.

8. **Log Policies**: Log collection policies and log directory can be configured to collect container logs for unified management and analysis. For details, see Collecting Default Standard Output Container Logs and Collecting Container Logs from Mount Paths.

Step 4 Click Next: Set Application Access. Then, click Add Service and set the workload access type.

If your workload will be reachable to other workloads or public networks, add a Service to define the workload access type.

The workload access type determines the network attributes of the workload. Workloads with different access types can provide different network capabilities. For details, see Overview.

Step 5 Click Next: Configure Advanced Settings to configure advanced policies.

- **Upgrade Policy**:
  - **Upgrade Mode**: Only Rolling upgrade is supported. During a rolling upgrade, old pods are gradually replaced with new ones. During the
Upgrade, service traffic is evenly distributed to both pods to ensure service continuity.

- **Maximum Number of Unavailable Pods**: Maximum number of unavailable pods allowed in a rolling upgrade. If the number is equal to the total number of pods, services may be interrupted. Minimum number of alive pods = Total pods – Maximum number of unavailable pods

- **Graceful Deletion**
  
  **Graceful Time Window**: Enter the time. The graceful scale-in policy provides a time window for workload deletion and is reserved for executing commands in the PreStop phase in the lifecycle. If workload processes are not terminated after the time window elapses, the workload will be forcibly deleted.

- **Scheduling Policies**: You can combine static global scheduling policies or dynamic runtime scheduling policies as required. For details, see [Scheduling Policy Overview](#).

- **Advanced Pod Settings**
  
  - **Pod Label**: The built-in app label is specified when the workload is created. It is used to set affinity and anti-affinity scheduling and cannot be modified. You can click Add Label to add labels.

  ![Advanced Pod Settings](image)

  **Figure 6-4 Advanced pod settings**

- **Client DNS Configuration**: A CCE cluster has a built-in DNS add-on (CoreDNS) to provide domain name resolution for workloads in the cluster. For details, see [Using Kubernetes In-Cluster DNS](#).
  
  - **DNS Policy**
    
    - **ClusterFirst**: The default DNS configuration overrides the Nameserver and DNS Search Domain configurations of the client.
    
    - **None**: Only the Nameserver and DNS Search Domain configurations are used for domain name resolution.
    
    - **Default**: The pod inherits the DNS configuration from the node on which the pod runs.
  
  - **Nameserver**: You can configure a domain name server for a user-defined domain name. The value is one or a group of DNS IP addresses, for example, 1.2.3.4.
  
  - **DNS Search Domain**: a search list for host-name lookup. When a domain name cannot be resolved, DNS queries will be attempted combining the domain name with each domain in the search list in turn until a match is found or all domains in the search list are tried.
  
  - **Timeout (s)**: amount of time the resolver will wait for a response from a remote name server before retrying the query on a different name server. Set it based on the site requirements.
- **ndots**: threshold for the number of dots that must appear in a domain name before an initial absolute query will be made. If a domain name has **ndots** or more than **ndots** dots, the name is a fully qualified domain name (FQDN) and will be tried first as an absolute name. If a domain name has less than **ndots** dots, the operating system will look up the name in a list of search domain names.

- **Custom Monitoring**: You can specify the metrics to be collected and the path and port from which the monitoring system will regularly collect the specified metrics. For details, see *Working with Prometheus for Custom Monitoring*.

- **APM Settings**: APM helps you quickly locate workload problems and identify performance bottlenecks to improve user experience. For details, see *Configuring APM Settings for Performance Bottleneck Analysis*.

**Step 6** After the preceding configurations are complete, click **Create**. On the page displayed, click **Return to Workload List** to view the workload status.

If the workload is in the **Running** state, it has been successfully created.

Workload status is not updated in real time. Click in the upper right corner or press F5 to refresh the page.

----End

### 6.5 Creating a Job

**Scenario**

Jobs are short-lived and run for a certain time to completion. They can be executed immediately after being deployed. It is completed after it exits normally (exit 0).

A job is a resource object that is used to control batch tasks. It is different from a long-term servo workload (such as Deployment and StatefulSet).

A job is started and terminated at specific times, while a long-term servo workload runs unceasingly unless being terminated. The pods managed by a job automatically exit after successfully completing the job based on user configurations. The success flag varies according to the spec.completions policy.

- One-off jobs: A single pod runs once until successful termination.
- Jobs with a fixed success count: N pods run until successful termination.
- A queue job is considered completed based on the global success confirmed by the application.

**Prerequisites**

Nodes have been added. For more information, see *Creating Pay-per-Use Nodes*. If clusters and nodes are available, you need not create them again.

**Procedure**

**Step 1** (Optional) If you use a private container image to create your job, upload the container image to the image repository.
For details, see Uploading an Image Through a Docker Client.

**Step 2** Log in to the CCE console. In the navigation pane, choose Workloads > Jobs. Click Create Job.

**Step 3** Configure the basic job information listed in Table 6-10. The parameters marked with an asterisk (*) are mandatory.

<table>
<thead>
<tr>
<th>Table 6-10 Basic job information</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Parameter</strong></td>
</tr>
<tr>
<td><em>Job Name</em></td>
</tr>
<tr>
<td><em>Cluster</em></td>
</tr>
<tr>
<td><em>Namespace</em></td>
</tr>
<tr>
<td><em>Instances</em></td>
</tr>
<tr>
<td><strong>Description</strong></td>
</tr>
</tbody>
</table>

**Step 4** Click Next: Add Container to add a container and an image.

1. Click Select Container Image to select the image to be deployed.
   - **My Images**: displays all image repositories you created.
   - **Open Source Images**: Create a workload using an image in an open source image registry.
   - **Third-Party Images**: Create a job using an image from any third-party image repository (image repositories other than SWR and open source image registries). When you create a job using a third-party image, ensure that the node where the job is running can access public networks. For details about how to use a third-party image, see Using a Third-Party Image.

   - If your image repository does not require authentication, set Secret Authentication to No, enter an image address in Image Address, and then click OK.
   - If your image repository must be authenticated (account and password), you need to create a secret and then use a third-party image. For details, see Using a Third-Party Image.
   - **Shared Images**: The images shared by other tenants using the SWR service are displayed here. You can create workloads based on the shared images.
2. Set image parameters.

**Table 6-11 Image parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Image</td>
<td>Name of the image. You can click <strong>Change Image</strong> to update it.</td>
</tr>
<tr>
<td>*Image Version</td>
<td>Select the image tag to be deployed.</td>
</tr>
<tr>
<td>*Container Name</td>
<td>Name of the container. You can modify it.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| **CPU**            | **Request**: minimum number of CPU cores required by a container. The default value is 0.25 cores.  
|                    | **Limit**: maximum number of CPU cores available for a container. Do not leave Limit unspecified. Otherwise, intensive use of container resources will occur and your workload may exhibit unexpected behavior.  |
| **Memory**         | **Request**: minimum amount of memory required by a container. The default value is 0.5 GiB.  
|                    | **Limit**: maximum amount of memory available for a container. When memory usage exceeds the specified memory limit, the container will be terminated.                                                        |
| **GPU**            | Configurable only when the cluster contains GPU nodes.  
|                    | It indicates the percentage of GPU resources reserved for a container. Select **Use** and set the percentage. For example, if this parameter is set to 10%, the container is allowed to use 10% of GPU resources. If you do not select **Use** or set this parameter to 0, no GPU resources can be used.  |
| **GPU/Graphics Card** | The workload's pods will be scheduled to the node with the specified GPU.  
|                    | If **Any GPU type** is selected, the container uses a random GPU in the node. If you select a specific GPU, the container uses that GPU accordingly.                                                                 |
| **Ascend 310 Quota** | Number of Ascend 310 processors requested by a container. The value must be an integer.  
|                    | The settings of this parameter take effect only if an Ascend-accelerated node is selected and the **huawei-npu** add-on is installed. Ascend-accelerated nodes are available for open beta testing (OBT) now. This type of nodes, powered by HiSilicon Ascend 310 AI processors, are suitable for image recognition, video processing, AI inference, machine learning, and other scenarios that require high performance and low power consumption. Click **here** to apply for OBT now. |

3. (Optional) Configure advanced settings.
Table 6-12 Advanced settings

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lifecycle</td>
<td>Lifecycle scripts define the actions taken for container-related jobs when a lifecycle event occurs.</td>
</tr>
<tr>
<td></td>
<td>- Start Command: You can set the command to be executed immediately after the container is started. For details, see Configuring a Container.</td>
</tr>
<tr>
<td></td>
<td>- Post-Start: The command is triggered after a job starts. For details, see Setting Container Lifecycle Parameters.</td>
</tr>
<tr>
<td></td>
<td>- Pre-Stop: The command is triggered before a job is stopped. For details, see Setting Container Lifecycle Parameters.</td>
</tr>
<tr>
<td>Environment Variables</td>
<td>Environment variables can be added to a container. In general, environment variables are used to set parameters. On the Environment Variables tab page, click Add Environment Variable. Currently, environment variables can be added using any of the following methods:</td>
</tr>
<tr>
<td></td>
<td>- Added manually: Set Variable Name and Variable Value/Reference.</td>
</tr>
<tr>
<td></td>
<td>- Added from Secret: Set Variable Name and select the desired secret name and data. A secret must be created in advance. For details, see Creating a Secret.</td>
</tr>
<tr>
<td></td>
<td>- Added from ConfigMap: Set Variable Name and select the desired ConfigMap name and data. A ConfigMap must be created in advance. For details, see Creating a ConfigMap.</td>
</tr>
<tr>
<td>Data Storage</td>
<td>The local disk or cloud storage can be mounted to a container to implement persistent data file storage. For details, see Storage (CSI).</td>
</tr>
<tr>
<td>Log Policies</td>
<td>Set a log policy and log path for collecting workload logs and preventing logs from being over-sized. For details, see Collecting Default Standard Output Container Logs.</td>
</tr>
</tbody>
</table>

4. (Optional) One job pod contains one or more related containers. If your job contains multiple containers, click Add Container to add containers.

Step 5 Click Create.

If the status of the job is Executing, the job has been created successfully.

----End

Using kubectl

A job has the following configuration parameters:
- **spec.template**: has the same schema as a pod.
- **RestartPolicy**: can only be set to **Never** or **OnFailure**.
- For a single-pod job, the job ends after the pod runs successfully by default.
- **.spec.completions**: indicates the number of pods that need to run successfully to end a job. The default value is **1**.
- **.spec.parallelism**: indicates the number of pods that run concurrently. The default value is **1**.
- **.spec.backoffLimit**: indicates the maximum number of retries performed if a pod fails. When the limit is reached, the pod will not try again.
- **.spec.activeDeadlineSeconds**: indicates the running time of pods. Once the time is reached, all pods of the job are terminated. The priority of .spec.activeDeadlineSeconds is higher than that of .spec.backoffLimit. That is, if a job reaches the .spec.activeDeadlineSeconds, the spec.backoffLimit is ignored.

Based on the **.spec.completions** and **.spec.Parallelism** settings, jobs are classified into the following types.

<table>
<thead>
<tr>
<th>Job Type</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>One-off jobs</td>
<td>A single pod runs once until successful termination.</td>
<td>Database migration</td>
</tr>
<tr>
<td>Jobs with a fixed completion count</td>
<td>One pod runs until reaching the specified completions count.</td>
<td>Work queue processing pod</td>
</tr>
<tr>
<td>Parallel jobs with a fixed completion count</td>
<td>Multiple pods run until reaching the specified completions count.</td>
<td>Multiple pods for processing work queues concurrently</td>
</tr>
<tr>
<td>Parallel jobs</td>
<td>One or more pods run until successful termination.</td>
<td>Multiple pods for processing work queues concurrently</td>
</tr>
</tbody>
</table>

The following is an example job, which calculates π till the 2000th digit and prints the output.

```yaml
apiVersion: batch/v1
kind: Job
metadata:
  name: pi-with-timeout
spec:
  completions: 50
  parallelism: 5
  backoffLimit: 5
  template:
    spec:
      containers:
        - name: pi
          image: perl
          # 50 pods need to be run to finish a job. In this example, π is printed for 50 times.
          # 5 pods are run in parallel.
          # The maximum number of retry times is 5.
```
command: ["perl", "-Mbignum=bpi", "-wle", "print bpi(2000)""]
restartPolicy: Never

### Description

- **apiVersion:** batch/v1 indicates the version of the current job.
- **kind:** Job indicates that the current resource is a job.
- **restartPolicy:** Never indicates the current restart policy. For jobs, this parameter can only be set to Never or OnFailure. For other controllers (for example, Deployments), you can set this parameter to Always.

### Run the job.

**Step 1** Start the job.

```
[root@k8s-master k8s]# kubectl apply -f myjob.yaml
job.batch/myjob created
```

**Step 2** View the job details.

```
kubectl get job
```

<table>
<thead>
<tr>
<th>NAME</th>
<th>COMPLETIONS</th>
<th>DURATION</th>
<th>AGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>myjob</td>
<td>1/1</td>
<td>23s</td>
<td>3m45s</td>
</tr>
</tbody>
</table>

If the value of COMPLETIONS is 1/1, the job is successfully executed.

**Step 3** Query the pod status.

```
kubectl get pod
```

<table>
<thead>
<tr>
<th>NAME</th>
<th>READY</th>
<th>STATUS</th>
<th>RESTARTS</th>
<th>AGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>myjob-29qlw</td>
<td>0/1</td>
<td>Completed</td>
<td>0</td>
<td>4m5s</td>
</tr>
</tbody>
</table>

If the status is Completed, the job is complete.

**Step 4** View the pod logs.

```
kubectl logs
```

```
myjob-29qlw-29qlw
hello k8s job!
``` 

----End

### Related Operations

After a one-off job is created, you can perform operations listed in Table 6-14.

#### Table 6-14 Other operations

<table>
<thead>
<tr>
<th>Operation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Viewing a YAML</td>
<td>Click View YAML next to the job name to view the YAML file corresponding to the current job.</td>
</tr>
</tbody>
</table>
6.6 Creating a Cron Job

Scenario

A cron job runs on a repeating schedule. You can perform time synchronization for all active nodes at a fixed time point.

A cron job runs periodically at the specified time. It is similar with Linux crontab. A cron job has the following characteristics:

- Runs only once at the specified time.
- Runs periodically at the specified time.

The typical usage of a cron job is as follows:

- Schedules jobs at the specified time.
- Creates jobs to run periodically, for example, database backup and email sending.

Prerequisites

Nodes have been added. For more information, see Creating Pay-per-Use Nodes. If clusters and nodes are available, you need not create them again.

Procedure

Step 1  (Optional) If you use a private container image to create your cron job, upload the container image to the image repository.

For details, see Software Repository for Container User Guide.

Step 2  Log in to the CCE console. In the navigation pane, choose Workloads > Cron Jobs. Then, click Create Cron Job.

Step 3  Configure the basic cron job information listed in Table 6-15. The parameters marked with an asterisk (*) are mandatory.
Table 6-15 Basic cron job information

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>* Job Name</td>
<td>Name of a new cron job. The name must be unique. Enter 4 to 52 characters starting with a lowercase letter and ending with a letter or digit. Only lowercase letters, digits, and hyphens (-) are allowed.</td>
</tr>
<tr>
<td>* Cluster</td>
<td>Cluster to which a new cron job belongs.</td>
</tr>
<tr>
<td>* Namespace</td>
<td>Namespace to which a cron job belongs. If you do not specify this parameter, the value <code>default</code> is used by default.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of a cron job.</td>
</tr>
</tbody>
</table>

**Step 4** Click **Next: Configure Timing Rule**.

**Step 5** Set the timing rule.

Table 6-16 Timing rule parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>* Concurrency</td>
<td>The following policies are supported:</td>
</tr>
<tr>
<td>Policy</td>
<td></td>
</tr>
<tr>
<td></td>
<td>● Forbid: A new job cannot be created before the previous job is complete.</td>
</tr>
<tr>
<td></td>
<td>● Allow: The cron job allows concurrently running jobs, which preempt cluster resources.</td>
</tr>
<tr>
<td></td>
<td>● Replace: A new job replaces the previous job when it is time to create the job but the previous job is not complete.</td>
</tr>
<tr>
<td>* Schedule</td>
<td>Time when a new cron job is executed.</td>
</tr>
<tr>
<td>Job Records</td>
<td>You can set the number of jobs that are successfully executed or fail to be executed. Setting a limit to 0 corresponds to keeping none of the jobs after they finish.</td>
</tr>
</tbody>
</table>

**Step 6** Click **Next: Add Container** to add a container.

1. Click **Select Container Image** to select the image to be deployed.
   - **My Images**: displays all image repositories you created.
   - **Open Source Images**: Create a workload using an image in an open source image registry.
   - **Third-Party Images**: Create a job using an image from any third-party image repository (image repositories other than SWR and open source image registries). When you create a job using a third-party image, ensure that the node where the job is running can access public networks. For details about how to use a third-party image, see **Using a Third-Party Image**.
- If your image repository does not require authentication, set Secret Authentication to No, enter an image address in Image Address, and then click OK.

- If your image repository must be authenticated (account and password), you need to create a secret and then use a third-party image. For details, see Using a Third-Party Image.

  - **Shared Images**: The images shared by other tenants using the SWR service are displayed here. You can create workloads based on the shared images.

2. Set image parameters.

**Table 6-17 Image parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Image</td>
<td>Name of the image. You can click Change Image to update it.</td>
</tr>
<tr>
<td>*Image Version</td>
<td>Select the image tag to be deployed.</td>
</tr>
<tr>
<td>*Container Name</td>
<td>Name of the container. You can modify it.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Container Resources</td>
<td><strong>CPU</strong></td>
</tr>
<tr>
<td></td>
<td>- <strong>Request</strong>: minimum number of CPU cores required by a container. The default value is 0.25 cores.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Limit</strong>: maximum number of CPU cores available for a container. Do not leave <strong>Limit</strong> unspecified. Otherwise, intensive use of container resources will occur and your workload may exhibit unexpected behavior.</td>
</tr>
<tr>
<td></td>
<td><strong>Memory</strong></td>
</tr>
<tr>
<td></td>
<td>- <strong>Request</strong>: minimum amount of memory required by a container. The default value is 0.5 GiB.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Limit</strong>: maximum amount of memory available for a container. When memory usage exceeds the specified memory limit, the container will be terminated.</td>
</tr>
<tr>
<td></td>
<td>For more information about <strong>Request</strong> and <strong>Limit</strong>, see <a href="#">Setting Container Specifications</a>.</td>
</tr>
<tr>
<td></td>
<td><strong>GPU</strong>: configurable only when the cluster contains GPU nodes.</td>
</tr>
<tr>
<td></td>
<td>It indicates the percentage of GPU resources reserved for a container. Select <strong>Use</strong> and set the percentage. For example, if this parameter is set to 10%, the container is allowed to use 10% of GPU resources. If you do not select <strong>Use</strong> or set this parameter to 0, no GPU resources can be used.</td>
</tr>
<tr>
<td></td>
<td><strong>GPU/Graphics Card</strong>: The workload's pods will be scheduled to the node with the specified GPU.</td>
</tr>
<tr>
<td></td>
<td>If <strong>Any GPU type</strong> is selected, the container uses a random GPU in the node. If you select a specific GPU, the container uses that GPU accordingly.</td>
</tr>
<tr>
<td></td>
<td><strong>Ascend 310 Quota</strong>: number of Ascend 310 processors requested by a container. The value must be an integer.</td>
</tr>
<tr>
<td></td>
<td>The settings of this parameter take effect only if an Ascend-accelerated node is selected and the <a href="#">huawei-npu</a> add-on is installed. Ascend-accelerated nodes are available for open beta testing (OBT) now. This type of nodes, powered by HiSilicon Ascend 310 AI processors, are suitable for image recognition, video processing, AI inference, machine learning, and other scenarios that require high performance and low power consumption. Click <a href="#">here</a> to apply for OBT now.</td>
</tr>
</tbody>
</table>

3. (Optional) Configure advanced settings.
### Table 6-18 Advanced settings

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lifecycle</td>
<td>Actions defined in the lifecycle script definition are taken for the lifecycle events of container tasks.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Start Command</strong>: You can set the command to be executed immediately after the container is started. For details, see <a href="#">Configuring a Container</a>.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Post-Start</strong>: The command is triggered after a job starts. For details, see <a href="#">Setting Container Lifecycle Parameters</a>.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Pre-Stop</strong>: The command is triggered before a job is stopped. For details, see <a href="#">Setting Container Lifecycle Parameters</a>.</td>
</tr>
<tr>
<td>Environment Variables</td>
<td>Environment variables can be added to a container. In general, environment variables are used to set parameters. On the Environment Variables tab page, click <a href="#">Add Environment Variable</a>. Currently, environment variables can be added using any of the following methods:</td>
</tr>
<tr>
<td></td>
<td>- <strong>Added manually</strong>: Set <strong>Variable Name</strong> and <strong>Variable Value/Reference</strong>.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Added from Secret</strong>: Set <strong>Variable Name</strong> and select the desired secret name and data. A secret must be created in advance. For details, see <a href="#">Creating a Secret</a>.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Added from ConfigMap</strong>: Set <strong>Variable Name</strong> and select the desired ConfigMap name and data. A ConfigMap must be created in advance. For details, see <a href="#">Creating a ConfigMap</a>.</td>
</tr>
</tbody>
</table>

4. (Optional) One job pod contains one or more related containers. If your cron job contains multiple containers, click [Add Container](#) to add containers.

**Step 7** Click **Create**.

If the status is **Started**, the cron job has been created successfully.

----End

### Using kubectl

A cron job has the following configuration parameters:

- **.spec.schedule**: takes a **Cron** format string, for example, `0 * * * *` or `@hourly`, as schedule time of jobs to be created and executed.
- **.spec.jobTemplate**: specifies jobs to be run, and has exactly the same schema as when you are [Creating a Job Using kubectl](#).
- **.spec.startingDeadlineSeconds**: specifies the deadline for starting a job.
- **.spec.concurrencyPolicy**: specifies how to treat concurrent executions of a job created by the Cron job. The following options are supported:
- **Allow** (default value): allows concurrently running jobs.
- **Forbid**: forbids concurrent runs, skipping next run if previous has not finished yet.
- **Replace**: cancels the currently running job and replaces it with a new one.

The following is an example cron job, which is saved in the `cronjob.yaml` file.

```yaml
apiVersion: batch/v1beta1
kind: CronJob
metadata:
  name: hello
spec:
schedule: "*/1 * * * *"
jobTemplate:
  spec:
    template:
      spec:
        containers:
        - name: hello
          image: busybox
          args:
            - /bin/sh
            - /c
            - date; echo Hello from the Kubernetes cluster
        restartPolicy: OnFailure
```

**Run the job.**

**Step 1** Create a cron job.

`kubectl create -f cronjob.yaml`

Information similar to the following is displayed:

```
cronjob.batch/hello created
```

**Step 2** Query the running status of the cron job:

```
kubectl get cronjob
NAME  SCHEDULE  SUSPEND  ACTIVE  LAST SCHEDULE  AGE
hello  */1 * * * *  False  0  <none>  9s
```

```
kubectl get jobs
NAME               COMPLETIONS   DURATION   AGE
hello-1597387980   1/1           27s        45s
```

```
kubectl get pod
NAME                           READY  STATUS  RESTARTS  AGE
hello-1597387980-tjv8f         0/1     Completed  0  114s
hello-1597388040-lckg9         0/1     Completed  0  39s
```

```
kubectl logs hello-1597387980-tjv8f
Fri Aug 14 06:56:31 UTC 2020
Hello from the Kubernetes cluster
```

```
kubectl delete cronjob hello
cronjob.batch "hello" deleted
```
When a cron job is deleted, the related jobs and pods are deleted too.

Related Operations

After a cron job is created, you can perform operations listed in Table 6-19.

Table 6-19 Other operations

<table>
<thead>
<tr>
<th>Operation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Editing a YAML file</td>
<td>Click More &gt; View YAML next to the cron job name to view the YAML file of the current job.</td>
</tr>
</tbody>
</table>
| Stopping a cron job | 1. Select the job to be stopped and click Stop in the Operation column.  
                     2. Click OK.                                                          |
| Deleting a cron job | 1. Select the cron job to be deleted and click More > Delete in the Operation column.  
                     2. Click OK.  
                     Deleted jobs cannot be restored. Therefore, exercise caution when deleting a job. |

6.7 Managing Pods

Scenario

A pod is the smallest and simplest unit in the Kubernetes object model that you create or deploy. A pod encapsulates an application's container (or, in some cases, multiple containers), storage resources, a unique network identity (IP address), as well as options that govern how the container(s) should run. A pod represents a single instance of an application in Kubernetes, which might consist of either a single container or a small number of containers that are tightly coupled and that share resources.

Pods in a Kubernetes cluster can be used in either of the following ways:

- **Pods that run a single container.** The "one-container-per-pod" model is the most common Kubernetes use case. In this case, a pod functions as a wrapper around a single container, and Kubernetes manages the pods rather than the containers directly.

- **Pods that run multiple containers that need to work together.** A pod might encapsulate an application composed of multiple co-located containers that are tightly coupled and need to share resources. The possible scenarios are as follows:
You can easily manage pods on CCE, such as editing YAML files and monitoring pods.

**Editing a YAML File**

To edit and download the YAML file of a pod online, do as follows:

**Step 1** Log in to the CCE console. In the navigation pane, choose **Workloads > Pods**.

**Step 2** Click **Edit YAML** at the same row as the target pod. In the **Edit YAML** dialog box displayed, modify the YAML file of the pod.

**Step 3** Click **Edit** and then **OK** to save the changes.

--- NOTE

If a pod is created by another workload, its YAML file cannot be modified individually on the **Pods** page.

**Step 4** (Optional) In the **Edit YAML** window, click **Download** to download the YAML file.

--- End

**Monitoring Pods**

On the CCE console, you can view the CPU and memory usage, upstream and downstream rates, and disk read/write rates of a workload pod to determine the required resource specifications.

**Step 1** Log in to the CCE console. In the navigation pane, choose **Workloads > Pods**.

**Step 2** Click **Monitoring** at the same row as the target pod to view the CPU and memory usage, upstream and downstream rates, and disk read/write rates of the pod.

--- NOTE

You cannot view the monitoring data of a pod that is not running.

--- End

**Deleting a Pod**

If a pod is no longer needed, you can delete it. Deleted pods cannot be recovered. Exercise caution when performing this operation.

**Step 1** Log in to the CCE console. In the navigation pane, choose **Workloads > Pods**.

**Step 2** Click **Delete** at the same row as the pod to be deleted.
Read the system prompts carefully. A pod cannot be restored after it is deleted. Exercise caution when performing this operation.

**Step 3** Click Yes to delete the pod.

**NOTE**
- If the node where the pod is located is unavailable or shut down and the workload cannot be deleted, you can forcibly delete the pod from the pod list on the workload details page.
- Ensure that the storage volumes to be deleted are not used by other workloads. If these volumes are imported or have snapshots, you can only unbind them.

---End

**Helpful Links**
- The Distributed System Toolkit: Patterns for Composite Containers
- Container Design Patterns

### 6.8 Managing Workloads and Jobs

**Scenario**

After a workload is created, you can scale, upgrade, monitor, roll back, or delete the workload, as well as edit its YAML file.

**Table 6-20 Workload/Job management**

<table>
<thead>
<tr>
<th>Operation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logging</td>
<td>You can view logs of Deployments, StatefulSets, DaemonSets, and jobs.</td>
</tr>
<tr>
<td>Upgrade</td>
<td>You can replace images or image tags to quickly upgrade Deployments, StatefulSets, and DaemonSets without interrupting services.</td>
</tr>
<tr>
<td>Editing a YAML file</td>
<td>You can modify and download the YAML files of Deployments, StatefulSets, DaemonSets, and pods on the CCE console. YAML files of jobs and cron jobs can only be viewed, copied, and downloaded.</td>
</tr>
<tr>
<td>Scaling</td>
<td>A workload can be automatically resized according to scaling policies, freeing you from the efforts to manually adjust resources for fluctuating service traffic. This saves you big on both resources and labors.</td>
</tr>
<tr>
<td>Monitoring</td>
<td>You can view the CPU and memory usage of Deployments, DaemonSets, and pods on the CCE console to determine the resource specifications you may need.</td>
</tr>
<tr>
<td>Rollback</td>
<td>Only Deployments can be rolled back.</td>
</tr>
<tr>
<td>Pausing</td>
<td>Only Deployments can be paused.</td>
</tr>
<tr>
<td>Operation</td>
<td>Description</td>
</tr>
<tr>
<td>----------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Resuming</td>
<td>Only Deployments can be resumed.</td>
</tr>
<tr>
<td>Labeling</td>
<td>Labels are key-value pairs and can be attached to workloads for affinity and anti-affinity scheduling.</td>
</tr>
<tr>
<td>Deletion</td>
<td>You can delete a workload or job that is no longer needed. Deleted workloads or jobs cannot be recovered.</td>
</tr>
<tr>
<td>Access settings</td>
<td>You can determine how your workloads can be accessed. For details, see Overview.</td>
</tr>
<tr>
<td>Scheduling policies</td>
<td>CCE supports custom and simple scheduling policies. Custom scheduling policies allow you to customize node affinity, workload affinity, and workload anti-affinity. Simple scheduling policies allow easy and convenient scheduling.</td>
</tr>
<tr>
<td>Workload O&amp;M</td>
<td>You can set custom metrics to monitor your workloads and configure performance management to quickly locate workload problems and analyze performance bottlenecks.</td>
</tr>
<tr>
<td>Event</td>
<td>CCE provides event names, event types, number of occurrences, Kubernetes events, first occurrence time, and last occurrence time by workload or pod.</td>
</tr>
</tbody>
</table>

**Viewing Logs**

You can view logs of Deployments, StatefulSets, DaemonSets, and jobs. This section uses a Deployment as an example to describe how to view logs.

**Step 1** Log in to the CCE console. In the navigation pane, choose Workloads > Deployments.

**Step 2** In the same row as the workload you will view, click Logs.

In the displayed Logs window, view the logs generated in the last 5 minutes, 30 minutes, or 1 hour.

**Figure 6-5** Viewing logs of a workload
CCE uses Application Operations Management (AOM) to provide the functions of viewing and querying logs. Prices of these log-related AOM functions have been adjusted on Oct 10, 2019 (see here). You can click here to view logs on the AOM console.

Upgrading a Workload

You can replace images or image tags to quickly upgrade Deployments, StatefulSets, and DaemonSets without interrupting services.

This section uses a Deployment as an example to describe how to upgrade a workload.

Before replacing an image or image version, upload the new image to the SWR service. For details, see Uploading an Image Through a Container Engine Client.

Step 1 Log in to the CCE console. In the navigation pane, choose Workloads > Deployments, and click Upgrade for the Deployment to be upgraded.

- Workloads cannot be upgraded in batches.
- Before performing an in-place StatefulSet upgrade, you must manually delete old pods. Otherwise, the upgrade status is always displayed as Upgrading.

Step 2 Upgrade the Deployment.

- **Image Name**: To replace the Deployment image, click Replace Image and select a new image.

- **Image Version**: To replace the Deployment image version, select a new version from the Image Version drop-down list.

Figure 6-6 Replacing an image
● **Container Name**: To change the container name, click next to **Container Name** and enter a new name.

● **Privileged Container**: After this function is enabled, the container can access all devices on the host.

● **Container Resources**: You can set the CPU, memory, Ascend 310, and GPU quotas.

● **Advanced Settings**:
  - **Lifecycle**: Commands for starting and running containers can be set.
    - **Start Command**: executed when the workload is started. For details, see [Setting Container Startup Commands](#).
    - **Post-Start**: executed after the workload runs successfully. For more information, see [Setting Container Lifecycle Parameters](#).
    - **Pre-Stop**: executed to delete logs or temporary files before the workload ends. For more information, see [Setting Container Lifecycle Parameters](#).
  - **Health Check**: CCE provides two types of probes: liveness probe and readiness probe. They are used to determine whether containers and user services are running properly. For more information, see [Setting Health Check for a Container](#).
    - **Liveness Probe**: used to restart the unhealthy container.
    - **Readiness Probe**: used to change the container to the unready state when detecting that the container is unhealthy. In this way, service traffic will not be directed to the container.
  - **Environment Variables**: Environment variables can be added to a container. In general, environment variables are used to set parameters.

On the **Environment Variables** tab page, click **Add Environment Variable**. Currently, three types of environment variables are supported:
- **Added manually**: Set **Variable Name** and **Variable Value/Reference**.

- **Added from Secret**: Set **Variable Name** and select the desired secret name and data. A secret must be created in advance. For details, see [Creating a Secret](#).

- **Added from ConfigMap**: Set **Variable Name** and select the desired ConfigMap name and data. A ConfigMap must be created in advance. For details, see [Creating a ConfigMap](#)

**NOTE**

To edit an environment variable that has been set, click **Edit**. To delete an environment variable that has been set, click **Delete**.

- **Data Storage**: Data storage can be mounted to containers for persistent storage and high disk I/O. Local disks and cloud storage volumes are supported. For details, see [Storage (CSI)](#).

**NOTE**

You can add data storage volumes only when creating a StatefulSet.

- **Security Context**: Container permissions can be configured to protect CCE and other containers from being affected.

  Enter the user ID to set container permissions and prevent systems and other containers from being affected.

- **Log Policies**: Log collection policies and log directory can be configured to collect container logs for unified management and analysis. For details, see [Collecting Default Standard Output Container Logs](#) and [Collecting Container Logs from Mount Paths](#).

**Step 3** Click **Submit**.

-----End

### Editing a YAML file

You can modify and download the YAML files of Deployments, StatefulSets, DaemonSets, and pods on the CCE console. YAML files of jobs and cron jobs can only be viewed, copied, and downloaded. This section uses a Deployment as an example to describe how to edit the YAML file.

**Step 1** Log in to the CCE console. In the navigation pane, choose **Workloads > Deployments**.

**Step 2** In the same row as the workload you will edit, choose **Operation > More > Edit YAML**. In the **Edit YAML** window, edit the YAML file of the current workload.

**Step 3** Click **Edit** and then **OK** to save the changes.

**Step 4** (Optional) In the **Edit YAML** window, click **Download** to download the YAML file.

-----End
Scaling a Workload

A workload can be automatically resized according to custom scaling policies, freeing you from the efforts to manually adjust the amount of resources for fluctuating service traffic. This saves you big on both resources and labors. This section uses a Deployment as an example to describe how to scale a workload.

Step 1 Log in to the CCE console. In the navigation pane, choose Workloads > Deployments.

Step 2 In the same row as the workload for which you will add a scaling policy, choose Operation > More > Scaling.

Step 3 On the Scaling tab page, add or edit scaling policies. Scaling policies are classified as auto and manual scaling policies.

For details, see Scaling a Workload.

----End

Monitoring a Workload

You can view the CPU and memory usage of Deployments, DaemonSets, and pods on the CCE console to determine the resource specifications you may need. This section uses a Deployment as an example to describe how to monitor a workload.

Step 1 Log in to the CCE console. In the navigation pane, choose Workloads > Deployments.

Step 2 Click the name of the Deployment to be monitored. On the displayed Deployment details page, click the Monitoring tab to view CPU usage and memory usage of the Deployment.

Step 3 Click the Pods tab. Click ☑ next to a pod to be monitored and click Monitoring.

Step 4 Check CPU usage and memory usage of the pod.

- CPU usage
  The horizontal axis indicates time while the vertical axis indicates the CPU usage. The green line indicates the CPU usage while the red line indicates the CPU usage limit.

  ☐ NOTE

  It takes some time to calculate CPU usage. Therefore, when CPU and memory usage are displayed for the first time, CPU usage is displayed about one minute later than memory usage.

  CPU and memory usage are displayed only for pods in the running state.

- Memory usage
  The horizontal axis indicates time while the vertical axis indicates the memory usage. The green line indicates the memory usage while the red line indicates the memory usage limit.

  ☐ NOTE

  Memory usage is displayed only for a running pod.

----End
Rolling Back a Workload (Available Only for Deployments)

CCE records the release history of all Deployments. You can roll back a Deployment to a specified version.

**Step 1** Log in to the CCE console. In the navigation pane, choose **Workloads > Deployments**.

**Step 2** In the same row as the Deployment you will roll back, choose **Operation > More > Roll Back**.

**Step 3** In the **Roll Back to This Version** drop-down list, select the version to which you will roll back the Deployment. Then, click **OK**.

----End

Pausing a Workload (Available Only for Deployments)

You can pause Deployments. After a Deployment is paused, the upgrade command can be successfully issued but will not be applied to the pods.

If you are performing a rolling upgrade, the rolling upgrade stops after the pause command is issued. In this case, the new and old pods coexist.

**Step 1** Log in to the CCE console. In the navigation pane, choose **Workloads > Deployments**.

**Step 2** In the same row as the Deployment you will pause, choose **Operation > More > Pause**.

**Step 3** In the displayed **Pause Workload** dialog box, click **OK**.

**Step 4** Click **OK**.

NOTICE

Deployments in the paused state cannot be rolled back.

----End

Resuming a Workload (Available Only for Deployments)

You can resume paused Deployments. After a Deployment is resumed, it can be upgraded or rolled back. Its pods will inherit the latest updates of the Deployment. If they are inconsistent, the pods are upgraded automatically according to the latest information of the Deployment.

**Step 1** Log in to the CCE console. In the navigation pane, choose **Workloads > Deployments**.

**Step 2** In the same row as the Deployment you will resume, choose **Operation > More > Resume**.

**Step 3** In the displayed **Resume Workload** dialog box, click **OK**.

----End
Managing Labels

Labels are key-value pairs and can be attached to workloads. Workload labels are often used for affinity and anti-affinity scheduling. You can add labels to multiple workloads or a specified workload.

You can manage the labels of Deployments, StatefulSets, and DaemonSets based on service requirements. This section uses Deployments as an example to describe how to manage labels.

In the following figure, three labels (release, env, and role) are defined for workload APP 1, APP 2, and APP 3. The values of these labels vary with workload.

- Label of APP 1: [release:alpha;env:development;role:frontend]
- Label of APP 2: [release:beta;env:testing;role:frontend]
- Label of APP 3: [release:alpha;env:production;role:backend]

If you set key to role and value to frontend when using workload scheduling or another function, APP 1 and APP 2 will be selected.

![Figure 6-8 Label example](image)

**Step 1** Log in to the CCE console. In the navigation pane, choose Workloads > Deployments.

**Step 2** Click the name of the workload whose labels will be managed.

**Step 3** On the workload details page, click Manage Label. In the displayed dialog box, click Add Label. Enter the label key and value, and click OK.

![Figure 6-9 Managing labels](image)
Deleting a Workload/Job

You can delete a workload or job that is no longer needed. Deleted workloads or jobs cannot be recovered. Exercise caution when you perform this operation. This section uses a Deployment as an example to describe how to delete a workload.

Step 1 Log in to the CCE console. In the navigation pane, choose Workloads > Deployments.

Step 2 In the same row as the workload you will delete, choose Operation > More > Delete.

Read the system prompts carefully. A workload cannot be recovered after it is deleted. Exercise caution when performing this operation.

Step 3 Click Yes.

NOTE

- If the node where the pod is located is unavailable or shut down and the workload cannot be deleted, you can forcibly delete the pod from the pod list on the workload details page.
- Ensure that the storage volumes to be deleted are not used by other workloads. If these volumes are imported or have snapshots, you can only unbind them.

Events

On the workload details page, click the Events or Pods tab to view the events, event types, number of occurrences, Kubernetes events, first occurrence time, and last occurrence time.

NOTE

Event data will be retained for one hour and then automatically deleted.

6.9 Scaling a Workload

After scaling policies are defined, pods can be automatically added or deleted based on resource changes, fixed time, and fixed periods. You do not need to manually adjust resource allocation to cope with service changes and data traffic spikes.

- Auto scaling: You can set metric-based, scheduled, and periodic policies. After configuration, pods can be automatically added or deleted based on resource changes or the specified schedule.
- Manual scaling: Pods are immediately added or deleted after the configuration is complete.
NOTE

Scaling policy priority: If you do not manually adjust the number of pods, auto scaling policies will take effect for resource scheduling. If manual scaling is triggered, auto scaling policies will be temporarily invalid.

Auto Scaling - HPA/CustomedHPA

HPA or CustomedHPA policies can be used for auto scaling. You can view all policies or perform more operations in Auto Scaling.

Auto Scaling - AOM

You can define auto scaling policies as required, which can intelligently adjust resources in response to service changes and data traffic spikes.

Auto scaling can be backed by Application Operations Management (AOM), but not for clusters of v1.17 and later.

Currently, CCE supports the following types of auto scaling policies:

- **Metric-based policy**: After a workload is created, pods will be automatically scaled when the workload's CPU or memory usage exceeds or falls below a preset limit.

- **Scheduled policy**: scaling at a specified time. Scheduled auto scaling is applicable for flash sales, premier shopping events, and other regular events that bring a high burst of traffic load.

- **Periodic policy**: scaling at a specified time on a daily, weekly, or monthly basis. Periodic scheduling is applicable to scenarios where traffic changes periodically.

- **Metric-based policy**: Supports auto scaling of a workload based on the CPU/memory usage.
  a. Log in to the CCE console. In the navigation pane, choose Workloads > Deployments or StatefulSets. In the same row as the target workload, choose More > Scaling.
  b. In the Auto Scaling area, click Add Scaling Policy.

Figure 6-10 Adding a metric-based policy
c. Set the policy parameters as listed in Table 6-21.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Policy Name</strong></td>
<td>Enter the name of the scaling policy. The policy name must be 1 to 64 characters in length and start with a letter. Only letters, digits, underscores (_), and hyphens (-) are allowed.</td>
</tr>
<tr>
<td><strong>Policy Type</strong></td>
<td>Set this parameter to <strong>Metric-based policy</strong>. The alarm policy is triggered based on historical data. The system checks whether the indicators set by the user in the monitoring window meet the triggering conditions <strong>every minute</strong>. If the triggering conditions are met for N consecutive periods, the system performs the action specified by the policy.</td>
</tr>
</tbody>
</table>
| **Metric** | Set the metrics that describe the resource performance data or status.  
- **CPU Usage**: CPU usage of the measured object. The value is the percentage of the used CPU cores to the total CPU cores.  
- **Data Sending Rate**: data volume sent by the measured object per second.  
- **Disk Write Rate**: data volume written into the disk per second.  
- **Physical Memory Usage**: percentage of the physical memory size used by the measured object to the physical memory size that the measured object has applied for.  
- **Physical Memory Size**: total physical memory size that the measured object has applied for.  
- **Data Receive Rate**: data volume received by the measured object per second.  
- **CPU Core Limit**: total number of CPU cores that the measured object has applied for.  
- **Physical Memory Used**: physical memory size used by the measured object.  
- **Disk Read Rate**: data volume read from the disk per second.  
- **CPU Cores Used**: number of CPU cores used by the measured object.  
- **Error Packets Received**: number of error packets received by the measured object. |
### Table: Scaling Policy Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| Trigger Condition       | The value can be higher (>) or lower (<) than a threshold. When the usage of the preceding metrics reaches the specified value, the scaling policy is triggered.  
For example, if Metric is set to CPU Usage and this parameter is set to > 70%, the scaling policy is triggered when the CPU usage exceeds 70%. |
| Monitoring window       | Size of the data aggregation window. If the value is set to 60, metric statistics are collected every 60 seconds.                               |
| Threshold Crossings     | Number of consecutive times that the threshold is reached within the monitoring window. The calculation cycle is fixed at one minute.  
If the parameter is set to 3, the action is triggered if threshold is reached for three consecutive measurement periods. |
| Action                  | Action executed after a policy is triggered. Two actions are available: add or reduce pods.                                                  |

d. Click **OK**.

e. In the **Auto Scaling** area, check that the policy has been started.

**Figure 6-11** Viewing a metric-based policy

When the trigger condition is met, the auto scaling policy starts automatically.

- **Scheduled policy**: scaling at a specified time.

a. In the **Auto Scaling** area, click **Add Scaling Policy**. Select **Scheduled policy**.
**Figure 6-12 Scheduled Policy**

![Scheduled Policy](image)

**Table 6-22 Parameters for adding a scheduled policy**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Policy Name</td>
<td>Enter the name of the scaling policy. The policy name must be 1 to 64 characters in length and start with a letter. Only letters, digits, underscores (_), and hyphens (-) are allowed.</td>
</tr>
<tr>
<td>Policy Type</td>
<td>Set this parameter to <em>Scheduled policy</em>.</td>
</tr>
<tr>
<td>Trigger Time</td>
<td>Time at which the policy is enforced.</td>
</tr>
<tr>
<td>Action</td>
<td>Action executed after a policy is triggered. Three actions are available: add pods, reduce pods, and set the number of pods.</td>
</tr>
</tbody>
</table>

b. Click **OK**.

c. In the **Auto Scaling** area, check that the policy has been started.

**Figure 6-13 Viewing a scheduled policy**

![Viewing a scheduled policy](image)

When the trigger time is reached, you can see on the **Pods** tab page that the auto scaling policy has taken effect.
**Figure 6-14** Auto scaling having taken effect

![Auto scaling having taken effect](image)

- **Periodic policy:** scaling at a specified time on a daily, weekly, or monthly basis.
  
  a. In the Auto Scaling area, click Add Scaling Policy. Select Periodic policy.

**Figure 6-15** Periodic policy

![Periodic policy](image)

**Table 6-23** Parameters for adding a periodic policy

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Policy Name</td>
<td>Enter the name of the scaling policy. The policy name must be 1 to 64 characters in length and start with a letter. Only letters, digits, underscores (_), and hyphens (-) are allowed.</td>
</tr>
<tr>
<td>Policy Type</td>
<td>Set this parameter to <strong>Periodic policy</strong>.</td>
</tr>
<tr>
<td>Time Range</td>
<td>Specify the time for triggering the policy.</td>
</tr>
<tr>
<td>Action</td>
<td>Action executed after a policy is triggered. Three actions are available: add pods, reduce pods, and set the number of pods.</td>
</tr>
</tbody>
</table>

b. Click **OK**.
c. In the **Auto Scaling** area, check that the policy has been started.

**Figure 6-16** Viewing a periodic policy

When the trigger condition is met, the auto scaling policy starts automatically.

**Manual Scaling**

**Step 1** Log in to the CCE console. In the navigation pane, choose **Workloads > Deployments** or **StatefulSets**. In the same row as the target workload, choose **More > Scaling**.

**Step 2** In the **Manual Scaling** area, click and change the number of pods to, for example, 3. Then, click **Save**. The scaling takes effect immediately.

**Figure 6-17** Changing the number of pods

**Step 3** On the **Pods** tab page, check that a new pod is being created. When the pod status becomes **Running**, pod scaling is complete.

**Figure 6-18** Manual scaling

---End

### 6.10 Configuring a Container

#### 6.10.1 Using a Third-Party Image

**Scenario**

CCE allows you to create workloads using images pulled from third-party image repositories.

Generally, a third-party image repository can be accessed only after authentication (using your account and password). CCE uses the secret-based authentication to
pull images. Therefore, you need to create a secret for an image repository before pulling images from the repository.

Prerequisites

The node where the workload is running is accessible from public networks. You can access public networks through LoadBalancer or DNAT.

Using the Console

Step 1 Create a secret for accessing a third-party image repository.

In the navigation pane, choose Configuration Center > Secret, and click Create Secret. Type must be set to kubernetes.io/dockerconfigjson. For details, see Creating a Secret.

Enter the user name and password used to access the third-party image repository.

Figure 6-19 Creating a secret

![Creating a secret](image)

Step 2 Create a workload. For details, see Creating a Deployment or Creating a StatefulSet. If the workload will be created from a third-party image, set the image parameters as follows:

1. Set Secret Authentication to Yes.
2. Select the secret created in step Step 1.
3. Enter the image address.

Step 3 Click Create.

----End
Using kubectl

**Step 1** Use kubectl to connect to the cluster. For details, see Connecting to a Cluster Using kubectl.

**Step 2** Create a secret of the dockercfg type using kubectl.

```bash
kubectl create secret docker-registry myregistrykey
--docker-server=DOCKER_REGISTRY_SERVER
--docker-username=DOCKER_USER
--docker-password=DOCKER_PASSWORD
--docker-email=DOCKER_EMAIL
```

In the preceding commands, `myregistrykey` indicates the secret name, and other parameters are described as follows:

- **DOCKER_REGISTRY_SERVER**: address of a third-party image repository, for example, `www.3rdregistry.com` or `10.10.10.10:443`
- **DOCKER_USER**: account used for logging in to a third-party image repository
- **DOCKER_PASSWORD**: password used for logging in to a third-party image repository
- **DOCKER_EMAIL**: email of a third-party image repository

**Step 3** Use a third-party image to create a workload.

A dockercfg secret is used for authentication when you obtain a private image. The following is an example of using the myregistrykey for authentication.

```yaml
apiVersion: v1
category: Pod
metadata:
  name: foo
  namespace: default
spec:
  containers:
  - name: foo
    image: www.3rdregistry.com/janedoe/awesomeapp:v1
    imagePullSecrets:
    - name: myregistrykey
      #Use the created secret.
----End
```

### 6.10.2 Setting Container Specifications

**Scenario**

CCE allows you to set resource limits for added containers during workload creation. You can request and limit the CPU and memory quotas used by each pod in the workload, and set whether to use GPU and Ascend 310 resources for each pod.

**Meanings**

For CPU and Memory, the meanings of Request and Limit are as follows:

- If **Request** is selected, the system schedules the pod to the node that meets the requirements for workload deployment based on the request value.
- If **Request** is deselected, the system schedules the pod to a random node for workload deployment.
- If **Limit** is selected, the system limits the resources used by the workload based on the preset value.
If Limit is deselected, the system does not limit the resources used by the pod. If the memory resources used by the pod exceed the memory allocated to the node, the workload or node may be unavailable.

**NOTE**

When creating a workload, you are advised to set the upper and lower limits of CPU and memory resources. If the upper and lower resource limits are not set for a workload, a resource leak of this workload will make resources unavailable for other workloads deployed on the same node. In addition, workloads that do not have upper and lower resource limits cannot be accurately monitored.

For GPU and Ascend 310 quotas, the meanings of Use and Any GPU type are as follows:

- If Use is selected, the system schedules the pod to a node that meets the requirements for workload deployment based on the configured value.
- Any GPU type is selected by default and cannot be deselected. This option indicates that the resources used by pods are not limited.

**Configuration Description**

- CPU quotas:

  **Table 6-24 Description of CPU quotas**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU request</td>
<td>Minimum number of CPU cores required by a container. Resources are scheduled for the container based on this value. The container can be scheduled to this node only when the total available CPU on the node is greater than or equal to the number of containerized CPU applications.</td>
</tr>
<tr>
<td>CPU limit</td>
<td>Maximum number of CPU cores available for a container.</td>
</tr>
</tbody>
</table>

**Recommended configuration**

Actual available CPU of a node $\geq$ Sum of CPU limits of all containers on the current node $\geq$ Sum of CPU requests of all containers on the current node. You can view the actual available CPUs of a node on the CCE console (Resource Management > Nodes > Allocatable).

- Memory quotas:

  **Table 6-25 Description of memory quotas**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Memory request</td>
<td>Minimum amount of memory required by a container. Resources are scheduled for the container based on this value. The container can be scheduled to this node only when the total available memory on the node is greater than or equal to the number of containerized memory applications.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>----------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Memory Limit</td>
<td>Maximum amount of memory available for a container. When the memory usage exceeds the configured memory limit, the instance may be restarted, which affects the normal use of the workload.</td>
</tr>
</tbody>
</table>

**Recommended configuration**

Actual available memory of a node ≥ Sum of memory limits of all containers on the current node ≥ Sum of memory requests of all containers on the current node. You can view the actual available memory of a node on the CCE console (Resource Management > Nodes > Allocatable).

**NOTE**

The allocatable resources are calculated based on the resource request value (Request), which indicates the upper limit of resources that can be requested by pods on this node, but does not indicate the actual available resources of the node. The calculation formula is as follows:

- Allocatable CPU = Total CPU – Requested CPU of all pods – Reserved CPU for other resources
- Allocatable memory = Total memory – Requested memory of all pods – Reserved memory for other resources

**Example**

Assume that a cluster contains a node with 4 cores and 8 GB. A workload containing two pods has been deployed on the cluster. The resources of the two pods (pods 1 and 2) are as follows: {CPU request, CPU limit, memory request, memory limit} = {1 core, 2 cores, 2 GB, 2 GB}.

The CPU and memory usage of the node is as follows:

- Allocatable CPU = 4 cores - (1 core requested by pod 1 + 1 core requested by pod 2) = 2 cores
- Allocatable memory = 8 GB - (2 GB requested by pod 1 + 2 GB requested by pod 2) = 4 GB

Therefore, the remaining 2 cores and 4 GB can be used by the next new pod.

**6.10.3 Setting Container Lifecycle Parameters**

**Scenario**

CCE provides callback functions for the lifecycle management of containerized applications. For example, if you want a container to perform a certain operation before stopping, you can register a hook function.

CCE provides the following lifecycle callback functions:

- **Start Command**: executed to start a container. For details, see Setting Container Startup Commands.
- **Post-Start**: executed immediately after a container is started. For details, see [Post-Start Processing](#).
- **Pre-Stop**: executed before a container is stopped. The pre-stop processing function helps you ensure that the services running on the pods can be completed in advance in the case of pod upgrade or deletion. For details, see [Pre-Stop Processing](#).

### Commands and Parameters Used to Run a Container

A Docker image has metadata that stores image information. If lifecycle commands and arguments are not set, CCE runs the default commands and arguments, that is, Docker instructions **ENTRYPOINT** and **CMD**, provided during image creation.

If the commands and arguments used to run a container are set during application creation, the default commands **ENTRYPOINT** and **CMD** are overwritten during image build. The rules are as follows:

<table>
<thead>
<tr>
<th>Image Entrypoint</th>
<th>Image CMD</th>
<th>Command to Run a Container</th>
<th>Parameters to Run a Container</th>
<th>Command Executed</th>
</tr>
</thead>
<tbody>
<tr>
<td>[touch]</td>
<td>[/root/test]</td>
<td>Not set</td>
<td>Not set</td>
<td>[touch /root/test]</td>
</tr>
<tr>
<td>[touch]</td>
<td>[/root/test]</td>
<td>[mkdir]</td>
<td>Not set</td>
<td>[mkdir]</td>
</tr>
<tr>
<td>[touch]</td>
<td>[/root/test]</td>
<td>Not set</td>
<td>[/opt/test]</td>
<td>[touch /opt/test]</td>
</tr>
<tr>
<td>[touch]</td>
<td>[/root/test]</td>
<td>[mkdir]</td>
<td>[/opt/test]</td>
<td>[mkdir /opt/test]</td>
</tr>
</tbody>
</table>

### Startup Commands

By default, the default command during image start. To run a specific command or rewrite the default image value, you must perform specific settings: For details, see [Setting Container Startup Commands](#).

### Post-Start Processing

**Step 1** Log in to the CCE console. Expand **Lifecycle** when adding a container during workload creation.

**Step 2** Set the post-start processing parameters, as listed in Table 6-27.
Table 6-27 Post-start processing parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| CLI         | Set commands to be executed in the container for post-start processing. The command format is **Command Args[1] Args[2]...** **Command** is a system command or a user-defined executable program. If no path is specified, an executable program in the default path will be selected. If multiple commands need to be executed, write the commands into a script for execution. **Example command:**  
  ```
  exec:  
  command:  
  - /install.sh  
  - install_agent
  ```  
  Enter `/install install_agent` in the script. This command indicates that `install.sh` will be executed after the container is created successfully. |
| HTTP request| Send an HTTP request for post-start processing. The related parameters are described as follows:  
  - **Path**: (optional) request URL.  
  - **Port**: (mandatory) request port.  
  - **Host Address**: (optional) IP address of the request. The default value is the IP address of the node where the container resides. |

---

**Pre-Stop Processing**

**Step 1** When creating a workload and adding a container, expand **Lifecycle**.

**Step 2** Set **pre-stop** parameters, as shown in **Table 6-27**.
### Table 6-28 Pre-stop parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| CLI       | Set commands to be executed in the container for pre-stop processing. The command format is **Command Args[1]** **Args[2]**... **Command** is a system command or a user-defined executable program. If no path is specified, an executable program in the default path will be selected. If multiple commands need to be executed, write the commands into a script for execution.  
Example command:  
```
exec:
  command:
    - /uninstall.sh
    - uninstall_agent
```
Enter `/uninstall uninstall_agent` in the script. This command indicates that the `uninstall.sh` script will be executed before the container completes its execution and stops running. |
| HTTP request | Send an HTTP request for pre-stop processing. The related parameters are described as follows:  
- **Path**: (optional) request URL.  
- **Port**: (mandatory) request port.  
- **Host Address**: (optional) IP address of the request. The default value is the IP address of the node where the container resides. |

### Container Restart Policy

The **restartPolicy** field is used to specify the pod restart policy. The restart policy type can be **Always**, **OnFailure**, or **Never**. The default value is **Always**.

When **restartPolicy** is used, containers are restarted only through kubelet on the same node.

<table>
<thead>
<tr>
<th>Restart Policy</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Always</td>
<td>When a container fails, kubelet automatically restarts the container.</td>
</tr>
<tr>
<td>OnFailure</td>
<td>When the container stops running and the exit code is not 0, kubelet automatically restarts the container.</td>
</tr>
<tr>
<td>Never</td>
<td>kubelet does not restart the container regardless of the container running status.</td>
</tr>
</tbody>
</table>
Controllers that can manage pods include ReplicaSet Controllers, jobs, DaemonSets, and kubelet (static pod).

- ReplicaSet Controller and DaemonSet: The policy must be set to **Always** to ensure that containers run continuously.
- Job: The policy can be set to **OnFailure** or **Never** to ensure that containers are not restarted after being executed.
- kubelet will restart a pod whenever it fails, regardless of the value of **restartPolicy**. In addition, no health check is performed on the pod.

### Example YAML for Setting the Container Lifecycle

This section uses Nginx as an example to describe how to set the container lifecycle.

**Step 1** Use `kubectl` to connect to the cluster. For details, see [Connecting to a Cluster Using kubectl](#).

**Step 2** Create and edit the `nginx-deployment.yaml` file. `nginx-deployment.yaml` is an example file name, and you can change it as required.

This YAML file defines the `nginx` deployment:

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx
spec:
  replicas: 1
  selector:
    matchLabels:
      app: nginx
  strategy:
    type: RollingUpdate
  template:
    metadata:
      labels:
        app: nginx
    spec:
      restartPolicy: Always               #Restart policy
      containers:
        - image: nginx
          command:
            - sleep 3600
          #Startup command
          imagePullPolicy: Always
          lifecycle:
            postStart:
              exec:
                command:
                  - /bin/bash
                  - install.sh
            #Post-start command
            preStop:
              exec:
                command:
                  - /bin/bash
                  - uninstall.sh
            #Pre-stop command
```

Cloud Container Engine
User Guide
6 Workloads

Issue 01 (2021-09-08)  Copyright © Huawei Technologies Co., Ltd.  230
6.10.4 Setting Container Startup Commands

**Scenario**

When creating a workload or job, you can use an image to specify the processes running in the container.

By default, the image runs the default command. To run a specific command or rewrite the default image value, you must perform the following settings:

- **Working directory**: working directory of the command.
  
  If the working directory is not specified in the image or on the console, the default value is `/`

- **Command**: command that controls the running of an image.

- **Args**: parameters transferred to the running command.

---

**NOTICE**

After a container is started, do not modify configurations in the container. If configurations in the container are modified (for example, passwords, certificates, and environment variables of a containerized application are added to the container), the configurations will be lost after the container restarts and container services will become abnormal. An example scenario of container restart is pod rescheduling due to node anomalies.

Configurations must be imported to a container as arguments. Otherwise, configurations will be lost after the container restarts.

---

**Commands and Arguments Used to Run a Container**

A Docker image has metadata that stores image information. If lifecycle commands and arguments are not set, CCE runs the default commands and arguments, that is, Docker instructions `ENTRYPOINT` and `CMD`, provided during image creation.

If the commands and arguments used to run a container are set during application creation, the default commands `ENTRYPOINT` and `CMD` are overwritten during image build. The rules are as follows:

**Table 6-29** Commands and parameters used to run a container

<table>
<thead>
<tr>
<th>Image Entrypoint</th>
<th>Image CMD</th>
<th>Command to Run a Container</th>
<th>Args to Run a Container</th>
<th>Command Executed</th>
</tr>
</thead>
<tbody>
<tr>
<td>[touch]</td>
<td>[/root/test]</td>
<td>Not set</td>
<td>Not set</td>
<td>[touch /root/test]</td>
</tr>
</tbody>
</table>
Setting the Startup Command

**Step 1** Log in to the CCE console. Expand **Lifecycle** when adding a container during workload or job creation.

**Step 2** Enter the running command and parameters, as shown in **Table 6-30**.

- **NOTE**
  - The current startup command is provided as a string array and corresponds to the Entrypoint startup command of Docker. The format is as follows: ["executable", "param1", "param2", ...]. For details about how to start Kubernetes containers, click [here](#).
  - The lifecycle of a container is the same as that of the startup command. That is, the lifecycle of the container ends after the command is executed.

<table>
<thead>
<tr>
<th>Configuration Item</th>
<th>Procedure</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Command</strong></td>
<td>Enter an executable command, for example, <code>/run/server</code>. If there are multiple commands, separate them with spaces. If the command contains a space, you need to add a quotation mark (&quot;&quot;&quot;). <strong>NOTE</strong> If there are multiple commands, you are advised to run the <code>/bin/sh</code> or other shell commands. Other commands are used as parameters, as shown in Figure 6-20.</td>
</tr>
<tr>
<td><strong>Args</strong></td>
<td>Enter the argument that controls the container running command, for example, <code>--port=8080</code>. If there are multiple arguments, separate them in different lines.</td>
</tr>
</tbody>
</table>

---

Cloud Container Engine
User Guide
6 Workloads

Issue 01 (2021-09-08) Copyright © Huawei Technologies Co., Ltd. 232
Figure 6-20 Example startup command

The following uses Nginx as an example to describe three typical application scenarios of the container startup command:

Example code:
```
nginx -c nginx.conf
```

- **Scenario 1:** Both the **command** and **arguments** are set.

Figure 6-21 Setting the startup command and arguments

Example YAML file:
```
command:
  - nginx
args:
  - '-c'
  - nginx.conf
```

- **Scenario 2:** Only the **command** is set.

Figure 6-22 Setting the startup command

**NOTE**

A command must be enclosed in double quotes. If no double quotes are added, the command is split into multiple commands based on space character.
Example YAML file:

```
command:
  - nginx -c nginx.conf
args:
```

- Scenario 3: Only **arguments** are set.

**Figure 6-23** Setting startup arguments

[Image: Command startup settings]

[NOTE]

If the container startup command is not added to the system path, run the `/bin/sh` command to execute the container startup command. The container startup command must be enclosed in double quotes.

Example YAML file:

```
command:
  - /bin/sh
args:
  - '-c'
  - '"nginx -c nginx.conf"'
```

**Step 3** Check or modify the YAML file.

- When creating a workload, in the **Configure Advanced Settings** step, click YAML on the right.

**Figure 6-24** Checking or editing a YAML file

[Image: YAML editing interface]

- After the workload is created, go to the workload list. In the same row as the workload, choose **More > Edit YAML**.
After the workload is created, go to the workload details page. On the displayed page, click **Edit YAML** in the upper right corner.

---End

**Example YAML for Setting Container Startup Commands**

This section uses Nginx as an example to describe how to set container startup commands using kubectl.

Use kubectl to connect to the cluster. For details, see **Connecting to a Cluster Using kubectl**. See **Using kubectl to create a Deployment** or **Using kubectl to create a StatefulSet**. For more details on how to set container startup commands, see [official Kubernetes documentation](#).

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx
spec:
  replicas: 1
  selector:
    matchLabels:
      app: nginx
  strategy:
    type: RollingUpdate
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
        - image: nginx
          command:
            - sleep
            - '3600'
          lifecycle:
            postStart:
              exec:
              #Startup command
```
6.10.5 Setting Health Check for a Container

Scenario

Health check regularly checks the health status of containers during container running. If the health check function is not configured, a pod cannot detect service exceptions or automatically restart the service to restore it. This will result in a situation where the pod status is normal but the service in the pod is abnormal.

CCE provides the following health check probes:

- **Liveness probe**: checks whether a container is still alive. It is similar to the `ps` command that checks whether a process exists. If the liveness check of a container fails, the cluster restarts the container. If the liveness check is successful, no operation is executed.

- **Readiness probe**: checks whether a container is ready to process user requests. Upon that the container is detected unready, service traffic will not be directed to the container. It may take a long time for some applications to start up before they can provide services. This is because that they need to load disk data or rely on startup of an external module. In this case, the application process is running, but the application cannot provide services. To address this issue, this health check probe is used. If the container readiness check fails, the cluster masks all requests sent to the container. If the container readiness check is successful, the container can be accessed.

Health Check Methods

- **HTTP request**

  This health check mode is applicable to containers that provide HTTP/HTTPS services. The cluster periodically initiates an HTTP/HTTPS GET request to such containers. If the return code of the HTTP/HTTPS response is within 200–399, the probe is successful. Otherwise, the probe fails. In this health check mode, you must specify a container listening port and an HTTP/HTTPS request path.

  For example, for a container that provides HTTP services, the HTTP check path is `/health-check`, the port is 80, and the host address is optional (which defaults to the container IP address). Here, 172.16.0.186 is used as an example, and we can get such a request: GET http://172.16.0.186:80/health-check. The cluster periodically initiates this request to the container.
For a container that provides TCP communication services, the cluster periodically establishes a TCP connection to the container. If the connection is successful, the probe is successful. Otherwise, the probe fails. In this health check mode, you must specify a container listening port.

For example, if you have a Nginx container with service port 80, after you specify TCP port 80 for container listening, the cluster will periodically initiate a TCP connection to port 80 of the container. If the connection is successful, the probe is successful. Otherwise, the probe fails. An example is as follows:
**CLI**

CLI is an efficient tool for health check. When using the CLI, you must specify an executable command in a container. The cluster periodically runs the command in the container. If the command output is 0, the health check is successful. Otherwise, the health check fails.

The CLI mode can be used to replace the HTTP request-based and TCP port-based health check.

- For a TCP port, you can write a program script to connect to a container port. If the connection is successful, the script returns 0. Otherwise, the script returns -1.
- For an HTTP request, you can write a program script to run the `wget` command for a container.

  `wget http://127.0.0.1:80/health-check`

  Check the return code of the response. If the return code is within 200–399, the script returns 0. Otherwise, the script returns -1.
NOTICE

- Put the program to be executed in the container image so that the program can be executed.
- If the command to be executed is a shell script, do not directly specify the script as the command, but add a script interpreter. For example, if the script is `/data/scripts/health_check.sh`, you must specify `sh/data/scripts/health_check.sh` for command execution. The reason is that the cluster is not in the terminal environment when executing programs in a container.

Common Parameter Description

Table 6-31 Common parameter description

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| Initial Delay (s) | Check delay time in seconds. Set this parameter according to the normal startup time of services.  
For example, if this parameter is set to 30, the health check will be started 30 seconds after the container is started. The time is reserved for containerized services to start. |
| Timeout (s)     | Timeout duration. Unit: second.  
For example, if this parameter is set to 10, the timeout wait time for performing a health check is 10s. If the wait time elapses, the health check is regarded as a failure. If the parameter is left blank or set to 0, the default timeout time is 1s. |

6.10.6 Setting an Environment Variable

Scenario

An environment variable is a variable whose value can affect the way a running container will behave. You can modify environment variables even after workloads are deployed, increasing flexibility in workload configuration.

The function of setting environment variables on CCE is the same as that of specifying `ENV` in a Dockerfile.

CCE provides three ways to add environment variables: Manually add environment variables, import environment variables from a secret, and import environment variables from a configMap.
After a container is started, do not modify configurations in the container. If configurations in the container are modified (for example, passwords, certificates, and environment variables of a containerized application are added to the container), the configurations will be lost after the container restarts and container services will become abnormal. An example scenario of container restart is pod rescheduling due to node anomalies. Configurations must be imported to a container as arguments. Otherwise, configurations will be lost after the container restarts.

Manually Adding Environment Variables

Step 1  When creating a workload, add a container image. Then, expand Environment Variables and click Add Environment Variables.

Step 2  Configure the following parameters as required:
   - **Type**: Set this to Added manually.
   - **Variable Name**: Enter a variable name, for example, demo.
   - **Variable Value/Reference**: Enter a variable value, for example, value.

---

Importing Environment Variables from a Secret

Step 1  You need to create a key first. For details, see Creating a Secret.

Step 2  When creating a workload, add a container image. Then, expand Environment Variables and click Add Environment Variables.

Step 3  Configure the following parameters as required:
   - **Type**: Set this to Added from Secret.
   - **Variable Name**: Enter a variable name.
   - **Variable Value/Reference**: Select the corresponding secret name and key.
Importing Environment Variables from a ConfigMap

Step 1  Create a ConfigMap first. For details, see Creating a ConfigMap.

Step 2  When creating a workload, add a container image. Then, expand Environment Variables and click Add Environment Variables.

Step 3  Configure the following parameters as required:
- **Type**: Set this to Added from ConfigMap.
- **Variable Name**: Enter a variable name.
- **Variable Value/Reference**: Select the corresponding ConfigMap name and key.

6.10.7 Collecting Default Standard Output Container Logs

**Scenario**

CCE allows you to configure policies for collecting, managing, and analyzing workload logs periodically to prevent logs from being over-sized.

This topic describes how to collect standard output logs of containers. If you want to configure a log policy to collect container logs from specified paths, see Collecting Container Logs from Mount Paths.

**Procedure**

Step 1  When creating a workload, add a container. Then, expand Log Policies.

Step 2  If only the default standard output logs of the container are required, you do not need to click Add Log Policy.

Step 3  Click Next: Set Application Access. The following uses an Nginx workload as an example.
Step 4  Perform the following operations to view logs.

After the workload is created, access Nginx. Go to the workload details page and click the **Logs** button in the upper right corner to view the log details. Wait for about 5 minutes to view logs.

**NOTE**

CCE uses Application Operations Management (AOM) to provide the functions of viewing and querying logs. Prices of these log-related AOM functions have been adjusted on Oct 10, 2019 (see [here](https://example.com)). You can click **here** to view logs on the AOM console.

--- End
6.10.8 Collecting Container Logs from Mount Paths

Scenario

CCE allows you to configure policies for collecting, managing, and analyzing workload logs periodically to prevent logs from being over-sized.

CCE uses Application Operations Management (AOM) to provide the functions of collecting, viewing and querying container logs. Before using these functions, you need to configure a log collection path. This section describes how to collect container logs from a specified path.

If you want to collect standard output logs of containers, see Collecting Default Standard Output Container Logs.

Notes

- The ICAgent only collects *.log, *.trace, and *.out text log files.
- AOM collects standard container output logs by default.

Procedure

Adding a Log Policy on CCE

Step 1 When creating a workload on the CCE console, add a container and expand Log Policies.

Figure 6-35 Container logs

Step 2 In the Log Policies area, click Add Log Policy. Configure parameters in the log policy. The following uses Nginx as an example.
Figure 6-36 Adding a log policy

Step 3 Set **Storage Type.** You can select **Host Path** or **Container Path.**

- **Host Path**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Storage Type</strong></td>
<td>Select <strong>Host Path.</strong> The host path is mounted to the specified container path (mount path). After the mounting, in the node host path, you can view the container logs output into the mount path.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>*Host Path</td>
<td>Enter the host path, for example, <code>/var/paas/sys/log/nginx</code>.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Container Path</td>
<td>Container path (for example, <code>/tmp</code>) to which the storage resources will be mounted.</td>
</tr>
<tr>
<td></td>
<td><strong>NOTICE</strong></td>
</tr>
<tr>
<td></td>
<td>- Do not mount storage to a system directory such as <code>/</code> or <code>/var/run</code>; this action may cause a container error to occur. You are advised to mount the container to an empty directory. If the directory is not empty, ensure that there are no files affecting container startup in the directory. Otherwise, such files will be replaced, resulting in failures to start the container and create the workload.</td>
</tr>
<tr>
<td></td>
<td>- When the container is mounted to a high-risk directory, you are advised to use an account with minimum permissions to start the container; otherwise, high-risk files on the host machine may be damaged.</td>
</tr>
<tr>
<td></td>
<td>- AOM collects only the first 20 log files that have been modified recently. It collects files from 2 levels of subdirectories by default.</td>
</tr>
<tr>
<td></td>
<td>- AOM only collects <code>.log</code>, <code>.trace</code>, and <code>.out</code> text log files in mounting paths.</td>
</tr>
<tr>
<td></td>
<td>- For details about how to set permissions for mount points in a container, see <a href="#">Configure a Security Context for a Pod or Container</a>.</td>
</tr>
<tr>
<td>Extended Host Path</td>
<td>Extended host paths contain pod IDs or container names to distinguish different containers into which the host path is mounted.</td>
</tr>
<tr>
<td></td>
<td>A level-3 directory is added to the original volume directory/subdirectory. You can easily obtain the files output by a single Pod.</td>
</tr>
<tr>
<td></td>
<td>- <strong>None</strong>: No extended path is configured.</td>
</tr>
<tr>
<td></td>
<td>- <strong>PodUID</strong>: ID of a pod.</td>
</tr>
<tr>
<td></td>
<td>- <strong>PodName</strong>: name of a pod.</td>
</tr>
<tr>
<td></td>
<td>- <strong>PodUID/ContainerName</strong>: ID of a pod or name of a container.</td>
</tr>
<tr>
<td></td>
<td>- <strong>PodName/ContainerName</strong>: name of a pod or container.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| Collection Path | A collection path narrows down the scope of collection to specified logs.  
- If no collection path is specified, log files in `.log`, `.trace`, and `.out` formats will be collected from the specified path.  
- `/Path/**/` indicates that all log files in `.log`, `.trace`, and `.out` formats will be recursively collected from the specified path and all subdirectories at 5 levels deep.  
- `*` in log file names indicates a fuzzy match.  
Example: The collection path `/tmp/**/test*.log` indicates that all `.log` files prefixed with `test` will be collected from `/tmp` and subdirectories at 5 levels deep.  
CAUTION: Ensure that the ICAgent version is 5.12.22 or later. |
| Log Dumping     | Log dump refers to rolling log files on a local host.  
- **Enabled**: AOM scans log files every minute. When a log file exceeds 50 MB, it is dumped immediately. A new `.zip` file is generated in the directory where the log file locates. For a log file, AOM stores only the latest 20 `.zip` files. When the number of `.zip` files exceeds 20, earlier `.zip` files will be deleted. After the dump is complete, the log file in AOM will be cleared.  
- **Disabled**: AOM does not dump log files.  
NOTE:  
- Log file rolling of AOM is implemented in the copytruncate mode. Before enabling log dumping, ensure that log files are written in the append mode. Otherwise, file holes may occur.  
- Currently, mainstream log components such as Log4j and Logback support log file rolling. If your log files already support rolling, skip the configuration. Otherwise, conflicts may occur.  
- You are advised to configure log file rolling for your own services to flexibly control the size and number of rolled files. |
| Multi-line Log  | If enabled, log data is sorted and displayed based on the user-defined multi-line policy.  
**Split Mode**  
- **Log Time**: Enter a time wildcard. For example, if the time in the log is 2017-01-01 23:59:59, the wildcard is `YYYY-MM-DD hh:mm:ss`.  
- **Regular Pattern**: Enter a regular expression. |
| Add Container Path | You can map one host path to multiple container mount paths. |

- **Container Path**
NOTE

Ensure that the **ICAgent** version is 5.10.79 or later.

**Table 6-33 Adding a log policy (Container Path)**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Storage Type</td>
<td>Select <strong>Container Path</strong>. The emptyDir mode is used. The temporary path of the node is mounted to the specified container path (mount path). After the mounting, the log data that exists in the temporary path but is not reported to AOM by the collector will disappear after the pod is deleted.</td>
</tr>
<tr>
<td>Container Path</td>
<td>Container path (for example, <code>/tmp</code>) to which the storage resources will be mounted.</td>
</tr>
<tr>
<td><strong>NOTICE</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Do not mount storage to a system directory such as <code>/</code> or <code>/var/</code> <code>run</code>; this action may cause a container error to occur. You are advised to mount the container to an empty directory. If the directory is not empty, ensure that there are no files affecting container startup in the directory. Otherwise, such files will be replaced, resulting in failures to start the container and create the workload.</td>
</tr>
<tr>
<td></td>
<td>- If the file system is mounted to a high-risk directory, you are advised to use an account with minimum permissions to start the container; otherwise, high-risk files on the host machine may be damaged.</td>
</tr>
<tr>
<td></td>
<td>- AOM collects only the first 20 log files that have been modified recently. It collects files from 2 levels of subdirectories by default.</td>
</tr>
<tr>
<td></td>
<td>- AOM only collects <code>.log</code>, <code>.trace</code>, and <code>.out</code> text log files in mounting paths.</td>
</tr>
<tr>
<td></td>
<td>- For details about how to set permissions for mount points in a container, see <a href="#">Configure a Security Context for a Pod or Container</a>.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| **Collection Path** | A collection path narrows down the scope of collection to specified logs.  
- If no collection path is specified, log files in `.log`, `.trace`, and `.out` formats will be collected from the specified path.  
- `/Path/**/` indicates that all log files in `.log`, `.trace`, and `.out` formats will be recursively collected from the specified path and all subdirectories at 5 levels deep.  
- `*` in log file names indicates a fuzzy match.  
Example: The collection path `/tmp/**/test*.log` indicates that all `.log` files prefixed with `test` will be collected from `/tmp` and subdirectories at 5 levels deep.  
**CAUTION**  
Ensure that the ICAgent version is 5.12.22 or later. |
| **Log Dumping**  | Log dump refers to rolling log files on a local host.  
- **Enabled**: AOM scans log files every minute. When a log file exceeds 50 MB, it is dumped immediately. A new `.zip` file is generated in the directory where the log file locates. For a log file, AOM stores only the latest 20 `.zip` files. When the number of `.zip` files exceeds 20, earlier `.zip` files will be deleted. After the dump is complete, the log file in AOM will be cleared.  
- **Disabled**: AOM does not dump log files.  
**NOTE**  
- Log file rolling of AOM is implemented in the copytruncate mode. Before enabling log dumping, ensure that log files are written in the append mode. Otherwise, file holes may occur.  
- Currently, mainstream log components such as Log4j and Logback support log file rolling. If your log files already support rolling, skip the configuration. Otherwise, conflicts may occur.  
- You are advised to configure log file rolling for your own services to flexibly control the size and number of rolled files. |
| **Multi-line Log** | If enabled, log data is sorted and displayed based on the user-defined multi-line policy.  
**Split Mode**  
- **Log Time**: Enter a time wildcard. For example, if the time in the log is 2017-01-01 23:59:59, the wildcard is `YYYY-MM-DD hh:mm:ss`.  
- **Regular Pattern**: Enter a regular expression. |
| **Add Container Path** | You can configure multiple container mount paths. |
Step 4  Click OK.

---End

Viewing Logs

After a log collection path is configured and the workload is created, the ICAgent collects log files from the configured path. The collection takes about 1 minute.

After the log collection is complete, go to the workload details page and click Logs in the upper right corner to view and analyze logs, as shown in Figure 6-37.

Figure 6-37  Viewing container logs in the workload details page

For details, see Viewing Log Files.

NOTE

CCE uses Application Operations Management (AOM) to provide the functions of viewing and querying logs. Prices of these log-related AOM functions have been adjusted on Oct 10, 2019 (see here). You can click here to view logs on the AOM console.

6.10.9 Working with Prometheus for Custom Monitoring

Background

CCE can interconnect with Prometheus for reporting custom metrics. You can view these metrics on Monitoring > Metric Monitoring on the Application Operations Management (AOM) console.

AOM uses metrics to describe resource data or states. Metrics can be divided into:

- System metrics: basic metrics provided by AOM, such as CPU usage and used CPU cores.
- Custom metrics: user-defined metrics. You can report custom metrics using the following methods:
  - Method 1: Use AOM APIs. For details, see Adding Monitoring Data and Querying Monitoring Data.
Method 2: Use Prometheus (interconnected during workload creation on CCE). For details, see Procedure.

Figure 6-38 User-defined metrics on the AOM console

Prerequisites
- You have created a Kubernetes cluster. For details, see Buying a CCE Cluster.
- You have connected to the master node of the cluster, which allows you to quickly view node information such as node labels. For details, see Connecting to a Cluster Using kubectl.
- Before setting custom monitoring metrics for a workload, you need to understand and install Prometheus and provide the GET API to obtain custom metric data from your workload.

**NOTE**
Only **Gauge metrics** of Prometheus can be obtained.

Procedure

**Step 1** In the step of Configure Advanced Settings during workload creation, navigate to the Custom Monitoring area.

**Step 2** Configure the values by referring to Figure 6-39. The port and path from which the custom metrics will be collected must be specified in your exporter. After the configuration, CCE will collect the custom metric data in response to the GET request "http://Pod IP:Specified port/Specified path", for example, http://192.168.1.19:8080/metrics.

Figure 6-39 Setting custom monitoring
Table 6-34 Parameter description

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Mandatory (Yes/No)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Report Path</td>
<td>URL provided by the exporter for CCE to obtain custom metric data.</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>The path consists of letters, digits, slashes (/), and underscores (_), and</td>
<td></td>
</tr>
<tr>
<td></td>
<td>must start with a slash (/). For example, /metrics.</td>
<td></td>
</tr>
<tr>
<td>Report Port</td>
<td>Port provided by the exporter for CCE to obtain custom metric data.</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>The port number is an integer from 1 to 65535. For example, 8080.</td>
<td></td>
</tr>
<tr>
<td>Monitoring Metrics</td>
<td>Name of the custom metric provided by the exporter. The name of a custom</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>metric is a string of 5 to 100 characters. Only letters, digits, and</td>
<td></td>
</tr>
<tr>
<td></td>
<td>underscores (_) are allowed. The format is as follows: [&quot;Custom metric</td>
<td></td>
</tr>
<tr>
<td></td>
<td>name 1&quot;,&quot;Custom metric name 2&quot;]. Use commas (_) to separate multiple</td>
<td></td>
</tr>
<tr>
<td></td>
<td>custom metric names. For example, [&quot;cpu_usage&quot;,&quot;mem_usage&quot;].</td>
<td></td>
</tr>
<tr>
<td></td>
<td>● If this parameter is not configured, CCE obtains all custom metric data.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>● If this parameter is configured, for example, [&quot;cpu_usage&quot;,&quot;mem_usage&quot;],</td>
<td></td>
</tr>
<tr>
<td></td>
<td>CCE filters custom metrics and obtains only the data of cpu_usage and</td>
<td></td>
</tr>
<tr>
<td></td>
<td>mem_usage.</td>
<td></td>
</tr>
</tbody>
</table>

Helpful Links

- Metric Introduction
- Host Metrics and Dimensions
- Cluster Metrics and Dimensions
- Container Metrics and Dimensions

6.10.10 Configuring APM Settings for Performance Bottleneck Analysis

Scenario

Currently, tracing, topology, and other O&M tools are supported for Java workloads. If you want to monitor the status of a Java workload, select Java probe in the APM Settings area of the Configure Advanced Settings page during workload creation and enter a monitoring group name.
You can configure Java workload monitoring when and after a workload is created.

NOTE

- You need to connect your Java application on CCE to Application Performance Management (APM) through the Pinpoint probe. For details, see Connecting a HUAWEI CLOUD Containerized Application to APM.
- If you have not enabled the Application Performance Management (APM) service, click Buy Package, select the number of application instances as prompted, and click Pay Now. After confirming the order, click Pay Now again to submit the order.

Procedure

Configuring Java workload monitoring during workload creation

Step 1 Log in to the CCE console. In the navigation pane, choose Workloads > Deployments or Workloads > StatefulSets, and click Create Deployment or Create StatefulSet.

Step 2 In the APM Settings area on the Configure Advanced Settings page, select Java probe. The Application Performance Management (APM) service will be enabled and a probe will be installed on the node.

NOTE

The probe provides call chains, topology, SQL analysis, and stack tracing for Java workloads. For running the probe, a small amount of CPU and memory resources will be consumed.

Step 3 Enter a monitoring group name, for example, testapp. If one or more monitoring groups are available, you can also select one from the drop-down list.

Step 4 Select a probe version. By default, latest is selected. For details about probe versions, click Version Description.

Step 5 Select a probe upgrade policy. By default, Automatic upgrade upon restart is selected.
The probe upgrade policy is the policy for obtaining the latest probe image. Available options are *Automatic upgrade upon restart* and *Manual upgrade*.

- **Automatic upgrade upon restart**: The system downloads the probe image each time the pod is restarted.
- **Manual upgrade**: This policy means that if a local image is available, the local image will be used. The system downloads the probe image only when a local image is unavailable.

---

**Configuring or Modifying Java Workload Monitoring After a Workload is Created**

**Step 1** Log in to the CCE console, and choose **Workloads > Deployments** or **Workload > StatefulSets** in the navigation pane.

**Step 2** In the workload list, click the name of the workload to be configured. On the workload details page, click the **Workload O&M** tab.

**Step 3** In the **APM Settings** area, click **Edit** and then select **Java probe**. The APM service will be enabled and a probe will be installed on the node.

**Figure 6-41 Editing APM settings**

![Editing APM settings](image)

**NOTE**

The probe provides call chains, topology, SQL analysis, and stack tracing for Java workloads. For running the probe, a small amount of CPU and memory resources will be consumed.

**Step 4** Enter a monitoring group name or change the existing group name to, for example, **testapp**. If one or more monitoring groups are available, you can select one from the drop-down list.

**Step 5** Select a probe version. By default, **latest** is selected. For details about probe versions, click **Version Description**.

**Step 6** Select a probe upgrade policy. By default, **Automatic upgrade upon restart** is selected.
Step 7  Click Restart Pod.

NOTICE

For clusters earlier than v1.9.7, APM settings cannot be modified for StatefulSets.

Step 8  Three minutes after the application is started, its data will be displayed on the APM console. You can log in to the APM console and optimize application performance through topology and tracing. For details, see APM User Guide.
6.10.11 Enabling ICMP Security Group Rules

**Scenario**

If a workload uses UDP for both load balancing and health check, enable ICMP security group rules for the backend servers.

**Procedure**

**Step 1** Log in to the Elastic Cloud Server (ECS) console, find the ECS corresponding to any node where the workload runs, and click the ECS name. The ECS details page is displayed.

**Step 2** On the **Security Groups** tab page, click **Modify Security Group Rule**.

*Figure 6-43* Modifying security group rules

**Step 3** On the page displayed, click the **Inbound Rules** tab and click **Add Rule** to add an inbound rule for the ECS. For details, see *Figure 6-44*. Then, click **OK**.
### 6.10.12 Using Kubernetes In-Cluster DNS

#### DNS Configuration Overview

Every Kubernetes cluster has a built-in DNS add-on (Kube-DNS or CoreDNS) to provide domain name resolution for workloads in the cluster. When handling a high concurrency of DNS queries, Kube-DNS/CoreDNS may encounter a performance bottleneck, that is, it may fail occasionally to fulfill DNS queries. There are cases when Kubernetes workloads initiate unnecessary DNS queries. This makes DNS overloaded if there are many concurrent DNS queries. Tuning DNS configuration for workloads will reduce the risks of DNS query failures to some extent.

For more information about the DNS, see [coredns (System Resource Add-on, Mandatory)](coredns) or [Configuring High Availability of kube-dns/CoreDNS Using kubectl](Configuring High Availability of kube-dns/CoreDNS Using kubectl).

#### Configuration Options in the DNS Resolver Configuration File for Linux Operating Systems

Run the `cat /etc/resolv.conf` command on a Linux node or container to view the DNS resolver configuration file. The following is an example DNS resolver configuration of a container in a Kubernetes cluster:

```
nameserver 10.247.x.x
search default.svc.cluster.local svc.cluster.local cluster.local
options ndots:5
```

#### Configuration Options

- **nameserver**: an IP address list of a name server that the resolver will query. If this parameter is set to 10.247.x.x, the resolver will query the Kube-DNS/CoreDNS. If this parameter is set to another IP address, the resolver will query a HUAWEI CLOUD DNS or on-premises DNS server.
- **search**: a search list for host-name lookup. When a domain name cannot be resolved, DNS queries will be attempted combining the domain name with
each domain in the search list in turn until a match is found or all domains in the search list are tried. For CCE clusters, the search list is currently limited to three domains per container. When a nonexistent domain name is being resolved, eight DNS queries will be initiated because each domain name (including those in the search list) will be queried twice, one for IPv4 and the other for IPv6.

- **options**: options that allow certain internal resolver variables to be modified. Common options include timeout, attempts, and ndots. The value `ndots:5` means that if a domain name has fewer than 5 dots `(`), DNS queries will be attempted by combining the domain name with each domain in the search list in turn. If no match is found after all the domains in the search list are tried, the domain name is then used for DNS query. If the domain name has 5 or more than 5 dots, it will be tried first for DNS query. In case that the domain name cannot be resolved, DNS queries will be attempted by combining the domain name with each domain in the search list in turn. For example, the domain name `www.huaweicloud.com` contains two dots `()` and the number of dots `()` is less than the value of ndots. Therefore, when this domain name is queried, the DNS query request sequence is as follows: `www.huaweicloud.com.default.svc.cluster.local`, `www.huaweicloud.com.svc.cluster.local`, `www.huaweicloud.com.cluster.local`, and `www.huaweicloud.com`. The IP address of the domain name can be resolved only after at least seven DNS query requests are initiated. In such cases, a large number of redundant DNS queries will be performed when the external domain name is accessed.

**NOTE**

For more information about configuration options in the resolver configuration file used by Linux operating systems, visit [http://man7.org/linux/man-pages/man5/resolv.conf.5.html](http://man7.org/linux/man-pages/man5/resolv.conf.5.html).

- **Configuration Options in the DNS Configuration File for Applications in Kubernetes Clusters**

As explained earlier, applications may initiate unnecessary DNS queries in some scenarios. Kubernetes provides DNS-related configuration options for applications. The use of application’s DNS configuration can effectively reduce unnecessary DNS queries in certain scenarios and improve service concurrency. In application configuration, there are two DNS-related fields: `dnsPolicy` and `dnsConfig`.

**dnsPolicy**

The `dnsPolicy` field is used to configure a DNS policy for an application. The default value is `ClusterFirst`. The DNS parameters in `dnsConfig` will be merged to the DNS file generated according to `dnsPolicy`. The merge rules are later explained in `dnsConfig` description. Currently, `dnsPolicy` supports the following four values:

- **ClusterFirst**: The Kube-DNS/CoreDNS of the cluster is used for workloads. The Kube-DNS/CoreDNS of clusters in CCE is cascaded with a HUAWEI CLOUD DNS by default. Containers can resolve both the cluster-internal domain names registered by a Service and the external domain names exposed to public networks. The search list (`search` option) and `ndots: 5` are present in the DNS configuration file. Therefore, when accessing an external domain name and a long cluster-internal domain name (for example, `kubernetes.default.svc.cluster.local`), the search list...
will usually be traversed first, resulting in at least six invalid DNS queries. The issue of invalid DNS queries disappears only when a short cluster-internal domain name (for example, kubernetes) is being accessed.

- **ClusterFirstWithHostNet**: By default, the DNS configuration file that the kubelet's --resolv-conf flag points to is configured for workloads running with hostNetwork, that is, a HUAWEI CLOUD DNS is used for CCE clusters.

  If dnsPolicy is set to ClusterFirstWithHostNet, Kube-DNS/CoreDNS is used for workloads, and container's DNS configuration file is the same as ClusterFirst, in which invalid DNS queries still exist.

- **Default**: Container's DNS configuration file is the DNS configuration file that the kubelet's --resolv-conf flag points to. In this case, a HUAWEI CLOUD DNS is used for CCE clusters. Both search and options fields are left unspecified. This configuration can only resolve the external domain names registered with the Internet, and not cluster-internal domain names. This configuration is free from the issue of invalid DNS queries.

- **None**: This value is introduced since Kubernetes v1.9 (Beta in v1.10). If dnsPolicy is set to None, the dnsConfig field must be specified because all DNS settings are supposed to be provided using the dnsConfig field.

**dnsConfig**

The dnsConfig field is used to configure DNS parameters for workloads. The configured parameters are merged to the DNS configuration file generated according to dnsPolicy. If dnsPolicy is set to None, the workload's DNS configuration file is specified by the dnsConfig field. If dnsPolicy is not set to None, the DNS parameters configured in dnsConfig are added to the DNS configuration file generated according to dnsPolicy.

- **nameservers**: a list of IP addresses that will be used as DNS servers. If workload's dnsPolicy is set to None, the list must contain at least one IP address, otherwise this property is optional. The servers listed will be combined to the nameservers generated from the specified DNS policy in dnsPolicy with duplicate addresses removed.

- **searches**: a list of DNS search domains for hostname lookup in the Pod. This property is optional. When specified, the provided list will be merged into the search domain names generated from the chosen DNS policy in dnsPolicy. Duplicate domain names are removed. Kubernetes allows for at most 6 search domains.

- **options**: an optional list of objects where each object may have a name property (required) and a value property (optional). The contents in this property will be merged to the options generated from the specified DNS policy in dnsPolicy. Duplicate entries are removed.

**Configuration Examples**

The following example describes how to configure DNS for workloads.

- **Use Case 1: Using Kube-DNS/CoreDNS Built in Kubernetes Clusters**

  **Scenario**

  Kubernetes in-cluster Kube-DNS/CoreDNS is applicable to resolving only cluster-internal domain names or cluster-internal domain names + external domain names. This is the default DNS for workloads.
**Example:**
```
apiVersion: v1
kind: Pod
metadata:
  namespace: default
  name: dns-example
spec:
  containers:
  - name: test
    image: nginx
dnsPolicy: ClusterFirst
```

Container's DNS configuration file:
```
nameserver 10.247.3.10
search default.svc.cluster.local svc.cluster.local cluster.local
options ndots:5
```

- **Use Case 2: Using a Cloud DNS**

  **Scenario**

  A DNS cannot resolve cluster-internal domain names and therefore is applicable to the scenario where workloads access only external domain names registered with the Internet.

  **Example:**
  ```
  apiVersion: v1
  kind: Pod
  metadata:
    namespace: default
    name: dns-example
  spec:
    containers:
    - name: test
      image: nginx
dnsPolicy: Default
  //The DNS configuration file that the kubelet's --resolv-conf flag points to is used. In this case, a DNS is used for CCE clusters.
  Container's DNS configuration file:
  nameserver 10.125.x.x
  ```

- **Use Case 3: Using Kube-DNS/CoreDNS for Workloads Running with hostNetwork**

  **Scenario**

  By default, a DNS is used for workloads running with hostNetwork. If workloads need to use Kube-DNS/CoreDNS, set `dnsPolicy` to `ClusterFirstWithHostNet`.

  **Example:**
  ```
  apiVersion: apps/v1
  kind: Deployment
  metadata:
    name: nginx
  spec:
    template:
      metadata:
        labels:
          app: nginx
      spec:
        hostNetwork: true
dnsPolicy: ClusterFirstWithHostNet
  containers:
  - name: nginx
    image: nginx:1.7.9
    ports:
    - containerPort: 80
  ```
Use Case 4: Customizing Application's DNS Configuration

Scenario

You can flexibly customize the DNS configuration file for applications. Using `dnsPolicy` and `dnsConfig` together can address almost all scenarios, including the scenarios in which an on-premises DNS will be used, multiple DNSs will be cascaded, and DNS configuration options will be modified.

Example 1: Using Your On-Premises DNS

Set `dnsPolicy` to `None` so application’s DNS configuration file is generated based on `dnsConfig`.

```yaml
apiVersion: v1
class: Pod
metadata:
  namespace: default
  name: dns-example
spec:
  containers:
    - name: test
      image: nginx
      dnsPolicy: "None"
      dnsConfig:  
        nameservers:
        - 10.2.3.4 //IP address of your on-premises DNS
        searches:
        - ns1.svc.cluster.local
        - my.dns.search.suffix
        options:
        - name: ndots
          value: "2"
        - name: timeout
          value: "3"
```

Container’s DNS configuration file:

```
nameserver 10.2.3.4
search ns1.svc.cluster.local my.dns.search.suffix
options timeout:3 ndots:2
```

Example 2: Modifying the ndots Option in the DNS Configuration File to Reduce Invalid DNS Queries

Set `dnsPolicy` to a value other than `None` so the DNS parameters configured in `dnsConfig` are added to the DNS configuration file generated based on `dnsPolicy`.

```yaml
apiVersion: v1
class: Pod
metadata:
  namespace: default
  name: dns-example
spec:
  containers:
    - name: test
      image: nginx
      dnsPolicy: "ClusterFirst"
      dnsConfig:  
        nameservers:
        - 10.2.3.4 //IP address of your on-premises DNS
        searches:
        - ns1.svc.cluster.local
        - my.dns.search.suffix
        options:
        - name: ndots
          value: "2" //Changes the ndots:5 option in the DNS configuration file generated based on the ClusterFirst policy to ndots:2.
```

Container’s DNS configuration file:

```
nameserver 10.2.3.4
search ns1.svc.cluster.local my.dns.search.suffix
options timeout:3 ndots:2
```
nameserver 10.247.3.10
search default.svc.cluster.local svc.cluster.local cluster.local
options ndots:2
7 Affinity and Anti-Affinity Scheduling

7.1 Scheduling Policy Overview

CCE supports custom and simple scheduling policies. A custom scheduling policy allows you to customize node affinity, workload affinity, and workload anti-affinity to meet higher requirements. A simple scheduling policy provides a simple and convenient scheduling mode with sufficient functions.

Custom Scheduling Policies

You can configure node affinity, workload affinity, and workload anti-affinity in custom scheduling policies.

- **Node Affinity**
- **Workload Affinity**
- **Workload Anti-Affinity**

**NOTE**

Custom scheduling policies depend on node labels and pod labels. You can use default labels or customize labels as required.

Simple Scheduling Policies

A simple scheduling policy allows you to configure affinity between workloads and AZs, between workloads and nodes, and between workloads.

- **Workload-AZ affinity**: Multiple AZ-based scheduling policies (including affinity and anti-affinity policies) can be configured. However, scheduling is performed as long as one of the scheduling policies is met.
  - Affinity between workloads and AZs: **Workload-AZ Affinity**
  - Anti-affinity between workloads and AZs: **Workload-AZ Anti-Affinity**
- **Workload-node affinity**: Multiple node-based scheduling policies (including affinity and anti-affinity scheduling) can be configured. However, scheduling is performed as long as one of the scheduling policies is met. For example, if a cluster contains nodes A, B, and C and two scheduling policies are set (one policy defines node A as an affinity node and the other policy defines node B...
as an anti-affinity node), then the workload can be scheduled to any node other than B.

- **Affinity between workloads and nodes**: Workload-Node Affinity
- **Anti-affinity between workloads and nodes**: Workload-Node Anti-Affinity

**Workload-workload affinity**: Multiple workload-based scheduling policies can be configured, but the labels in these policies must belong to the same workload.

- **Affinity between workloads**: For details, see Workload-Workload Affinity. You can deploy workloads on the same node to reduce consumption of network resources.

  *Figure 7-1* shows an example of affinity deployment, in which all workloads are deployed on the same node.

  **Figure 7-1** Affinity between workloads

  ![Node A](Node A)
  ![Workload1](Workload1) ![Workload2](Workload2)
  ![Workload3](Workload3) ![Workload4](Workload4)

- **Anti-affinity between workloads**: For details, see Workload-Workload Anti-Affinity. Constraining multiple instances of the same workload from being deployed on the same node reduces the impact of system breakdowns. Anti-affinity deployment is also recommended for workloads that may interfere with each other.

  *Figure 7-2* shows an example of anti-affinity deployment, in which four workloads are deployed on four different nodes.

  **Figure 7-2** Anti-affinity between workloads

  ![Node A](Node A) ![Node B](Node B) ![Node C](Node C) ![Node D](Node D)
  ![Workload1](Workload1) ![Workload2](Workload2) ![Workload3](Workload3) ![Workload4](Workload4)
When setting workload-workload affinity and workload-node affinity, ensure that the affinity relationships do not contradict each other; otherwise, workload deployment will fail.

For example, Workload 3 will fail to be deployed when the following conditions are met:

- Anti-affinity is configured for Workload 1 and Workload 2. Workload 1 is deployed on Node A and Workload 2 is deployed on Node B.
- Affinity is configured between Workload 2 and Workload 3, but the target node on which Workload 3 is to be deployed is Node C or Node A.

### 7.2 Custom Scheduling Policies

#### 7.2.1 Node Affinity

##### Using the CCE Console

**Step 1** Log in to the CCE console and choose Workloads > Deployments or Workloads > StatefulSets in the navigation pane.

**Step 2** Click a workload name in the Deployment or StatefulSet list. On the displayed workload details page, click the Scheduling Policies tab and then click Add Custom Scheduling Policy.

**Figure 7-3 Adding a custom scheduling policy**

**Step 3** In the Node Affinity area, you can specify node labels to meet required or preferred rules in scheduling.

**Table 7-1 Node affinity settings**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required</td>
<td>It specifies a rule that must be met in scheduling. It corresponds to requiredDuringSchedulingIgnoredDuringExecution in Kubernetes. You can click Add Rule to add multiple required rules. A pod will be scheduled on a node that meets any of the rules configured.</td>
</tr>
</tbody>
</table>
**Preferred**

It specifies a preference in scheduling. It corresponds to `preferredDuringSchedulingIgnoredDuringExecution` in Kubernetes. You can click **Add Rule** to add multiple preferred rules. The scheduler will try to enforce the rules but will not guarantee. If the scheduler cannot satisfy any one of the rules, the pod will still be scheduled.

### Step 4

Set a rule according to the following table. You can click **Add Selector** to configure multiple selectors for a rule.

**Table 7-2 Selector settings**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Preferred</strong></td>
<td>It specifies a preference in scheduling. It corresponds to <code>preferredDuringSchedulingIgnoredDuringExecution</code> in Kubernetes. You can click <strong>Add Rule</strong> to add multiple preferred rules. The scheduler will try to enforce the rules but will not guarantee. If the scheduler cannot satisfy any one of the rules, the pod will still be scheduled.</td>
</tr>
</tbody>
</table>
| **Weight** | ● This parameter is unavailable for a required rule.  
● Set the weight of a preferred rule. A higher weight indicates a higher priority. |
| **Label** | Node label. You can use the default label or customize a label. |
| **Operator** | The following relations are supported: **In**, **NotIn**, **Exists**, **DoesNotExist**, **Gt**, and **Lt** |
| **Value** | Tag value.  
Operators **In** and **NotIn** allow one or more label values. Values are separated with colons (`:`). Operators **Exists** and **DoesNotExist** are used to determine whether a label exists, and do not require a label value. If you set the operator to **Gt** or **Lt** for a label, the label value must be greater than or less than a certain integer. |
| **Operation** | You can click **Delete** to delete a selector. |
| **Add Selector** | A selector corresponds to **matchExpressions** in Kubernetes. You can click **Add Selector** to add multiple selectors for a scheduling rule. The rule is applied in scheduling only when all its selectors are satisfied. |
Using kubectl

This section uses Nginx as an example to describe how to configure node affinity. The dialog box for configuring node affinity is shown in Figure 7-4.

**Prerequisites**

A workload that uses the nginx container image has been deployed on a node.

**Procedure**

Set **Label** to **kubernetes.io/hostname**, add affinity nodes, and set the operator to **In**. Then, click **OK**.

**YAML file of the workload with node affinity:**

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx
  namespace: default
spec:
  replicas: 2
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
        - image: nginx
          imagePullPolicy: Always
          name: nginx
```
imagePullSecrets:
  - name: default-secret

affinity:
  nodeAffinity:
    requiredDuringSchedulingIgnoredDuringExecution:
      nodeSelectorTerms:
        - matchExpressions:
          - key: kubernetes.io/hostname
            operator: In
            values:
              - 192.168.6.174

7.2.2 Workload Affinity

Using the CCE Console

Workload affinity determines the pods as which the target workload will be deployed in the same topology domain.

**Step 1** Log in to the CCE console and choose **Workloads > Deployments** or **Workloads > StatefulSets** in the navigation pane.

**Step 2** Click a workload name in the Deployment or StatefulSet list. On the displayed workload details page, click the **Scheduling Policies** tab and then click **Add Custom Scheduling Policy**.

**Figure 7-5 Adding a custom scheduling policy**

Step 3 In the **Pod Affinity** area, set the namespace, topology key, and the label requirements to be met.

There are two types of pod affinity rules: **Required** (hard rule) and **Preferred** (soft rule). The label operators include **In**, **NotIn**, **Exists**, and **DoesNotExist**.

**Table 7-3 Pod affinity settings**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required</td>
<td>It specifies a rule that must be met in scheduling. It corresponds to <strong>requiredDuringSchedulingIgnoredDuringExecution</strong> in Kubernetes. You can click <strong>Add Rule</strong> to add multiple required rules. Ensure that all the labels specified in the rules must be in the same workload. Each rule requires a namespace and topology key.</td>
</tr>
</tbody>
</table>
**Preferred**

It specifies a preference in scheduling. It corresponds to `preferredDuringSchedulingIgnoredDuringExecution` in Kubernetes. You can click **Add Rule** to add multiple preferred rules. The scheduler will try to enforce the rules but will not guarantee. If the scheduler cannot satisfy any one of the rules, the pod will still be scheduled.

---

**Step 4** Set a rule according to the following table. You can click **Add Selector** to configure multiple selectors for a rule.

### Table 7-4 Selector settings

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| **Weight**  | • This parameter is unavailable for a required rule.  
• Set the weight of a preferred rule. A higher weight indicates a higher priority. |
| **Namespace** | By default, the namespace of the current pod is used. You can also use another namespace. |
| **Topology Key** | Key of the worker node label that the system uses to denote a topology domain in which scheduling can be performed. Default and custom node labels can be used. |
| **Label** | Label of the workload. You can customize the label name. |
| **Operator** | The following relations are supported: **In**, **NotIn**, **Exists**, and **DoesNotExist** |
| **Value** | Tag value.  
Operators **In** and **NotIn** allow one or more label values. Values are separated with colons (:). Operators **Exists** and **DoesNotExist** are used to determine whether a label exists, and do not require a label value. |
| **Operation** | You can click **Delete** to delete a selector. |
| **Add Selector** | A selector corresponds to **matchExpressions** in Kubernetes. You can click **Add Selector** to add multiple selectors for a scheduling rule. The rule is applied in scheduling only when all its selectors are satisfied. |
This section uses Nginx as an example to describe how to configure pod affinity. The dialog box for configuring pod affinity is shown in Figure 7-6.

**Prerequisites**

A workload that uses the nginx container image has been deployed on a node.

**Procedure**

Set `Namespace` to `default` and `Topology Key` to the built-in node label `kubernetes.io/hostname`, which means that the scheduling scope is a node. Set labels `app` and `type` and their value to `redis` and `database`, respectively. Set `Operator` to `In` and click OK.

The YAML of the workload with pod affinity is as follows:

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx
  namespace: default
spec:
  replicas: 2
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
        - image: nginx
```

---

Using kubectl

This section uses Nginx as an example to describe how to configure pod affinity. The dialog box for configuring pod affinity is shown in Figure 7-6.
imagePullPolicy: Always
name: nginx
imagePullSecrets:
- name: default-secret
affinity:
  nodeAffinity: {}
podAffinity: requiredDuringSchedulingIgnoredDuringExecution:
  - labelSelector:
    matchExpressions:
    - key: app
      operator: In
      values:
      - redis
    - key: type
      operator: In
      values:
      - database
    namespaces:
    - default
topologyKey: kubernetes.io/hostname

NOTICE
In this example, only when a candidate workload (for example, workload A) with both labels app=redis and type=database is found can the workload Nginx be successfully scheduled to the node of the candidate workload.

7.2.3 Workload Anti-Affinity

Using the CCE Console

Workload anti-affinity determines the pods from which the target workload will be deployed in a different topology domain.

Step 1 Log in to the CCE console and choose Workloads > Deployments or Workloads > StatefulSets in the navigation pane.

Step 2 Click a workload name in the Deployment or StatefulSet list. On the displayed workload details page, click the Scheduling Policies tab and then click Add Custom Scheduling Policy.

Step 3 In the Pod Anti-Affinity area, set the namespace, topology key, and the label requirements to be met.

There are two types of pod anti-affinity rules: Required (hard rule) and Preferred (soft rule), and the label operators include In, NotIn, Exists, and DoesNotExist.
Table 7-5 Workload anti-affinity settings

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required</td>
<td>It specifies a rule that must be met in scheduling. It corresponds to <code>requiredDuringSchedulingIgnoredDuringExecution</code> in Kubernetes. You can add multiple required rules. Ensure that all the labels specified in the rules must be in the same workload. Each rule requires a namespace and topology key.</td>
</tr>
<tr>
<td>Preferred</td>
<td>It specifies a preference in scheduling. It corresponds to <code>preferredDuringSchedulingIgnoredDuringExecution</code> in Kubernetes. You can add multiple preferred rules. The scheduler will try to enforce the rules but will not guarantee. If the scheduler cannot satisfy any one of the rules, the pod will still be scheduled.</td>
</tr>
</tbody>
</table>

Step 4 Set a rule according to the following table. You can click **Add Selector** to configure multiple selectors for a rule.

Table 7-6 Selector settings

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| Weight    | ● This parameter is unavailable for a required rule.  
            ● Set the weight of a preferred rule. A higher weight indicates a higher priority. |
| Namespace | By default, the namespace of the current pod is used. You can also use another namespace. |
| Topology Key | Key of the worker node label that the system uses to denote a topology domain in which scheduling can be performed. Default and custom node labels can be used. |
| Label     | Label of the workload. You can customize the label name. |
| Operator  | The following relations are supported: **In**, **NotIn**, **Exists**, and **DoesNotExist** |
| Value     | Tag value.  
            Operators **In** and **NotIn** allow one or more label values. Values are separated with colons (:). Operators **Exists** and **DoesNotExist** are used to determine whether a label exists, and do not require a label value. |
| Operation | You can click **Delete** to delete a selector. |
| Add Selector | A selector corresponds to `matchExpressions` in Kubernetes. You can click **Add Selector** to add multiple selectors for a scheduling rule. The rule is applied in scheduling only when all its selectors are satisfied. |
Figure 7-7 Pod anti-affinity scheduling policy

Using kubectl

This section uses Nginx as an example to describe how to configure pod anti-affinity. The dialog box for configuring pod affinity is shown in Figure 7-7.

**Prerequisites**

A workload that uses the nginx container image has been deployed on a node.

**Procedure**

Set **Namespace** to **default** and **Topology Key** to the built-in node label **kubernetes.io/hostname**, which means that the scheduling scope is a node. Set the label **app** and its value to **redis**. Set **Operator** to **In** and click **OK**.

The YAML of the workload with pod anti-affinity:

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx
namespace: default
spec:
  replicas: 2
  selector:
    matchLabels:
      app: nginx
template:
  metadata:
    labels:
      app: nginx
  spec:
    containers:
      - image: nginx
        imagePullPolicy: Always
```
7.3 Simple Scheduling Policies

7.3.1 Workload-AZ Affinity

Using the CCE Console

**Step 1** When [Creating a Deployment](#) or [Creating a StatefulSet](#), in the **Scheduling Policies** area on the **Configure Advanced Settings** page, click next to **Workload-AZ Affinity and Anti-affinity > Affinity with AZs.**

![Figure 7-8 Scheduling policies](image)

**Step 2** Select the AZ in which you want to deploy the workload.

The created workload will be deployed in the selected AZ.

----End

Using kubectl

This section uses an Nginx workload as an example to describe how to create a workload using kubectl.

**Prerequisites**

The ECS where the kubectl client runs has been connected to your cluster. For details, see [Connecting to a Cluster Using kubectl](#).

**Procedure**
When using `kubectl` to create a `Deployment` or using `kubectl` to create a `StatefulSet`, configure workload-AZ affinity. The following is an example YAML file for workload-AZ affinity.

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: az-in-deployment
spec:
  replicas: 1
  selector:
    matchLabels:
      app: az-in-deployment
strategy:
  type: RollingUpdate
template:
  metadata:
    labels:
      app: az-in-deployment
  spec:
    containers:
    - image: nginx
      imagePullPolicy: Always
      name: nginx
      imagePullSecrets:
      - name: default-secret
    affinity:
      nodeAffinity:
        requiredDuringSchedulingIgnoredDuringExecution:
          nodeSelectorTerms:
          - matchExpressions:
            - key: failure-domain.beta.kubernetes.io/zone #node's label key
              operator: In
              values:
              - az1 #node's key value
```

**Setting the Object Type After Creating a Workload**

**Step 1** Log in to the CCE console and choose **Workloads > Deployments** or **Workloads > StatefulSets** in the navigation pane.

**Step 2** Click the name of the workload for which you will add a scheduling policy. On the workload details page, choose **Scheduling Policies > Add Simple Scheduling Policy > Add Affinity Object**.

**Figure 7-9 Adding an affinity object – Availability Zone**

**Step 3** Set **Object Type** to **Availability Zone**, and select the AZ in which the workload is eligible to be deployed. The workload will be deployed in the selected AZ.
This method can be used to add, edit, or delete scheduling policies.

---End

7.3.2 Workload-AZ Anti-Affinity

Using the CCE Console

Step 1 When Creating a Deployment or Creating a StatefulSet, in the Scheduling Policies area on the Configure Advanced Settings page, click next to Workload-AZ Affinity and Anti-affinity > Anti-affinity with AZs.

Figure 7-10 Scheduling policies

Step 2 Select an AZ in which the workload is ineligible to be deployed.

The created workload is not deployed on the selected AZ.

---End

Using kubectl

This section uses Nginx as an example to describe how to create a workload using kubectl.

Prerequisites

The ECS where the kubectl client runs has been connected to your cluster. For details, see Connecting to a Cluster Using kubectl.

Procedure

When using kubectl to create a Deployment or using kubectl to create a StatefulSet, configure workload-AZ anti-affinity. The following is an example YAML file for workload-AZ anti-affinity.

```
apiVersion: apps/v1
group: nginx
kind: Deployment
metadata:
  name: nginx
spec:
  replicas: 1
  selector:
    matchLabels: app: nginx
  strategy:
    type: RollingUpdate
```

```yaml
template:
  Cloud Container Engine
  User Guide
  7
  Affinity and Anti-Affinity Scheduling
  Issue 01 (2021-09-08)
  Copyright © Huawei Technologies Co., Ltd.
```
Setting the Object Type After Creating a Workload

Step 1 Log in to the CCE console and choose Workloads > Deployments or Workloads > StatefulSets in the navigation pane.

Step 2 Click the name of the workload for which you will add a scheduling policy. On the workload details page, choose Scheduling Policies > Add Simple Scheduling Policy > Add Anti-affinity Object.

Figure 7-11 Adding an anti-affinity object – Availability Zone

Step 3 Set Object Type to Availability Zone and select the AZ in which the workload is ineligible to be deployed. The workload will be constrained from being deployed in the selected AZ.

NOTE

This method can be used to add, edit, or delete scheduling policies.

-----End
7.3.3 Workload-Node Affinity

Using the CCE Console

**Step 1** When Creating a Deployment or Creating a StatefulSet, in the Scheduling Policies area on the Configure Advanced Settings page, choose Workload-Node Affinity and Anti-affinity > Affinity with Nodes > Add.

![Affinity with nodes](image)

**Step 2** Select the node on which you want to deploy the workload, and click OK.

If you select multiple nodes, the system automatically chooses one of them during workload deployment.

----End

Using kubectl

This section uses an Nginx workload as an example to describe how to create a workload using kubectl.

**Prerequisites**

The ECS where the kubectl client runs has been connected to your cluster. For details, see Connecting to a Cluster Using kubectl.

**Procedure**

When using kubectl to create a Deployment or using kubectl to create a StatefulSet, configure workload-node affinity. The following is an example YAML file for workload-node affinity.

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx
spec:
  replicas: 1
  selector:
    matchLabels:
      app: nginx
```
Setting the Object Type After Creating a Workload

**Step 1** Log in to the CCE console and choose Workloads > Deployments or Workloads > StatefulSets in the navigation pane.

**Step 2** Click the name of the workload for which you will add a scheduling policy. On the workload details page, choose Scheduling Policies > Add Simple Scheduling Policy > Add Affinity Object.

**Step 3** Set Object Type to Node and select the node where the workload is to be deployed. The workload will be deployed on the selected node.

![Figure 7-13 Adding an affinity object - Node](image)

**NOTE**

This method can be used to add, edit, or delete scheduling policies.
### 7.3.4 Workload-Node Anti-Affinity

#### Using the CCE Console

**Step 1** When Creating a Deployment or Creating a StatefulSet, in the Scheduling Policies area on the Configure Advanced Settings page, choose Workload-Node Affinity and Anti-affinity > Anti-affinity with Nodes > Add.

![Ant-affinity with nodes](image)

**Step 2** Select the node on which the workload is ineligible to be deployed, and click **OK**. If you select multiple nodes, the workload will not be deployed on these nodes.

----End

#### Using kubectl

This section uses Nginx as an example to describe how to create a workload using kubectl.

**Prerequisites**

The ECS where the kubectl client runs has been connected to your cluster. For details, see Connecting to a Cluster Using kubectl.

**Procedure**

When using kubectl to create a Deployment or using kubectl to create a StatefulSet, configure workload-node affinity. The following is an example YAML file for workload-node anti-affinity.

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx
spec:
  replicas: 1
  selector:
    matchLabels:
      app: nginx
strategy:
  type: RollingUpdate
```

Setting the Object Type After Creating a Workload

**Step 1** Log in to the CCE console and choose **Workloads > Deployments** or **Workloads > StatefulSets** in the navigation pane.

**Step 2** Click the name of the workload for which you will add a scheduling policy. On the workload details page, choose **Scheduling Policies > Add Simple Scheduling Policy > Add Anti-affinity Object**.

**Step 3** Set **Object Type** to **Node** and select the node on which the workload is ineligible to be deployed. The workload will be constrained from being deployed on the selected node.

---

**Figure 7-15 Adding an anti-affinity object - Node**

- **NOTE**

This method can be used to add, edit, or delete scheduling policies.

---End
7.3.5 Workload-Workload Affinity

Using the CCE Console

Step 1 When Creating a Deployment or Creating a StatefulSet, in the Scheduling Policies area on the Configure Advanced Settings page, choose Inter-Pod Affinity and Anti-affinity > Affinity with Pods > Add.

Figure 7-16 Affinity between pods

Step 2 Select the workloads that will be co-located with the current workload on the same node, and click OK.

The workload to be created will be deployed on the same node as the selected affinity workloads.

-----End

Using kubectl

This section uses Nginx as an example to describe how to create a workload using kubectl.

Prerequisites

The ECS where the kubectl client runs has been connected to your cluster. For details, see Connecting to a Cluster Using kubectl.

Procedure

When using kubectl to create a Deployment or using kubectl to create a StatefulSet, configure workload-workload affinity. The following is an example YAML file for workload-workload affinity.

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx
spec:
  replicas: 1
  selector:
    matchLabels:
      app: nginx
  strategy:
    type: RollingUpdate
```
Setting the Object Type After Creating a Workload

**Step 1** Log in to the CCE console and choose Workloads > Deployments or Workloads > StatefulSets in the navigation pane.

**Step 2** Click the name of the workload for which you will add a scheduling policy. On the workload details page, choose Scheduling Policies > Add Simple Scheduling Policy > Add Affinity Object.

**Step 3** Set Object Type to Workload and select the workloads to be deployed on the same node as the created workload. The created workload and the selected workloads will be deployed on the same node.

---

**Figure 7-17** Adding an affinity object – Workload

[Image of the CCE console showing the process of adding an affinity object for workloads]

---

**NOTE**

This method can be used to add, edit, or delete scheduling policies.
7.3.6 Workload-Workload Anti-Affinity

Using the CCE Console

**Step 1** When Creating a Deployment or Creating a StatefulSet, in the Scheduling Policies area on the Configure Advanced Settings page, choose Inter-Pod Affinity and Anti-affinity > Anti-affinity with Pods > Add.

![Figure 7-18 Anti-affinity between pods](image)

**Step 2** Select the workloads to which you want to deploy the target workload on a different node, and click OK.

The workload to be created and the selected workloads will be deployed on different nodes.

----End

Using kubectl

This section uses Nginx as an example to describe how to create a workload using kubectl.

**Prerequisites**

The ECS where the kubectl client runs has been connected to your cluster. For details, see Connecting to a Cluster Using kubectl.

**Procedure**

When using kubectl to create a Deployment or using kubectl to create a StatefulSet, configure workload-workload anti-affinity. The following is an example YAML file for workload-workload anti-affinity.

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx
spec:
  replicas: 1
  selector:
    matchLabels:
      app: nginx
strategy:
  type: RollingUpdate
template:
```

Setting the Object Type After Creating a Workload

**Step 1** Log in to the CCE console and choose Workloads > Deployments or Workloads > StatefulSets in the navigation pane.

**Step 2** Click the name of the workload for which you will add a scheduling policy. On the workload details page, choose Scheduling Policies > Add Simple Scheduling Policy > Add Anti-affinity Object.

**Step 3** Set Object Type to Workload and select the workloads to be deployed on a different node from the created workload. The created workload and the selected workloads will be deployed on different nodes.

**Figure 7-19 Adding an anti-affinity object – Workload**

This method can be used to add, edit, or delete scheduling policies.

---

Note

This method can be used to add, edit, or delete scheduling policies.
8.1 Overview

By deeply integrating the Kubernetes networking capabilities with VPC, CCE provides stable and high-performance container networking for mutual access of workloads in complex scenarios.

Access Scenarios

Workload access scenarios can be categorized as follows:

- Workloads in the same cluster access each other.
- A workload accesses workloads outside the cluster.
- A workload accesses the Internet.

Figure 8-1 Network access diagram

- Use a Service for access.
- Use a private ELB in a VPC.
- Use a private ELB after peer connection is set up for VPCs.
- Use a public network ELB/public IP/DNAT for access.
- Access public networks using NAT.
Notes and Constraints

- A maximum of 6,000 Services can be created in each namespace. The Service mentioned here refers to the Kubernetes Service resource object which defines a logical set of pods and a policy by which to access them.
- If containers running with hostPort or hostNetwork need to be accessed by external networks, the container ports corresponding to the nodes where the containers are located must be enabled.

Service

Each workload in Kubernetes has one or more pods. The IP address of each pod is dynamically and randomly allocated by the network plug-in. (The IP address changes after the pod is restarted.) To shield the dynamic changes of these backend pods and balance the load of multiple pods, the resource object Service is introduced.

Services allow you to access a single or multiple containerized applications. Each Service has a fixed IP address and port during its lifecycle. Each Service targets one or more backend pods. In this way, frontend clients do not need to keep track of these pods, allowing pods to be added or reduced without worrying IP address changes.

Containers deployed in Kubernetes provide layer-7 network services using HTTP and HTTPS, and layer-4 network services using TCP and UDP. Services in Kubernetes are used to manage layer-4 network access in a cluster.

You can configure the following types of Services:

- **ClusterIP**: default type of Service. A ClusterIP Service is exposed to a virtual cluster IP address. This virtual address is only reachable based on the Service name from within the cluster by the pods.
- **NodePort**: A NodePort Service is exposed on each node's IP address at a static port. You can access a NodePort Service from outside the cluster by requesting `Node IP: Node port`. kube-proxy will forward the request to the virtual cluster IP address, and then the virtual IP address is used to address the backend pods.
- **LoadBalancer**: A LoadBalancer Service is exposed externally using a cloud provider's load balancer. The specified load balancer routes to the NodePort and ClusterIP Services.

ClusterIP and NodePort Services have the same behavior in different cloud service providers or self-built clusters, while LoadBalancer Services vary depending on the load balancing capability supported by the cloud service provider. For example, you can set the network type for the load balancer and adjust the weight bound to the backend. For details, see the following sections in this chapter.

Service Access Mode

CCE supports the following types of Services for internal and external access:

- **Intra-cluster access (ClusterIP)**
  
  A workload can be accessed by other workloads in the same cluster through a cluster-internal domain name. A cluster-internal domain name is in the format of `<User-defined Service name>`.<Namespace of the
workload\.svc\.cluster\.local, for example, nginx\.default\.svc\.cluster\.local. For details, see \textit{Intra-Cluster Access (ClusterIP)}.

- **NodePort**
  A Service is exposed on each node's IP address at a static port (NodePort). A ClusterIP Service, to which the NodePort Service will route, is automatically created. By requesting \(<\text{NodeIP}>:<\text{NodePort}>, you can access a NodePort Service from outside the cluster. For details, see \textit{NodePort}.

- **LoadBalancer**
  A workload can be accessed from public networks through a load balancer. LoadBalancer provides higher reliability than EIP-based NodePort because an EIP is no longer bound to a single node. The LoadBalancer access type is applicable to the scenario in which a Service exposed to public networks is required. The access address is in the format of \(<\text{IP address of public network load balancer}>:<\text{access port}\. For example, 10.117.117.117:80. For details, see \textit{LoadBalancer}.

- **Destination Network Address Translation (DNAT)**
  A NAT gateway is situated between cluster nodes and public networks and assigned an EIP. After receiving inbound requests from public networks, the NAT gateway translates the EIP (destination address in the inbound requests) into a cluster-internal address. It appears to workload users as if all nodes running the workload share the same EIP. DNAT provides higher reliability than EIP-based NodePort in which the EIP is bound to a single node and once the node is down, all inbound requests to the workload will not be distributed. The access address is in the format of \(<\text{EIP}>:<\text{access port}>, for example, 10.117.117.117:80. For details, see \textit{DNAT}.

### Layer-7 Load Balancing (Ingress)

Generally, the IP addresses of Services and pods are accessible to workloads in the same cluster. An external request needs to be forwarded by a load balancer to the NodePort exposing the Service on a node and then be forwarded by kube-proxy to corresponding pods through an edge router or be discarded by kube-proxy. An ingress is a set of rules that route external requests to the cluster.

Ingress provides the URL, load balancer, SSL offloading, and HTTP route for external access to the cluster. The cluster administrator needs to deploy an ingress controller to monitor changes of ingresses and Services, configure load balancing based on the rules, and provide access entries.

An ingress consists of:

- **Nginx**: implements load balancing among pods.
- **Ingress controller**: obtains the IP addresses of the pods corresponding to services from cluster APIs and adds the IP addresses to the Nginx configuration file.
- **Ingress**: creates virtual hosts for Nginx.

For details on how to create and manage ingresses, see \textit{Overview}.

### Network Policy

Network policies define communication rules between pods and between a pod and other network endpoints.
A network policy selects pods by labels and defines the communication rules allowed by the selected pods. For details, see Network Policies.

### 8.2 Networking Models

#### 8.2.1 Overview

Network models assign IP addresses to pods in a cluster and provide networking services. In CCE, you can select the following network models for your cluster:

- Container tunnel network
- VPC network

### Selecting a Network Model

Table 8-1 describes the differences of network models supported by CCE.

---

<table>
<thead>
<tr>
<th>Dimension</th>
<th>Tunnel Network</th>
<th>VPC Network</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core technology</td>
<td>OVS</td>
<td>IPvlan and VPC route</td>
</tr>
<tr>
<td>Applicable clusters</td>
<td>CCE cluster</td>
<td>CCE cluster</td>
</tr>
<tr>
<td>Network isolation</td>
<td>Yes. For details, see Network Policies.</td>
<td>No</td>
</tr>
<tr>
<td>Passthrough networking</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>
| IP address management      | ● The container CIDR block is allocated separately.  
                               ● CIDR blocks are divided by node and can be dynamically allocated (CIDR blocks can be dynamically added after being allocated.) | ● The container CIDR block is allocated separately.  
                               ● CIDR blocks are divided by node and statically allocated (the CIDR block cannot be changed after a node is created). |

---

**CAUTION**

After a cluster is created, the network model cannot be changed.
### Container Tunnel Network

The container tunnel network is constructed on but independent of the node network through tunnel encapsulation. This network model uses VXLAN to encapsulate Ethernet packets into UDP packets and transmits them in tunnels. Open vSwitch serves as the backend virtual switch. Though at some costs of performance, packet encapsulation and tunnel transmission enable higher interoperability and compatibility with advanced features (such as network policy-based isolation) for most common scenarios.
Pod-to-pod communication

- On the same node: Packets are directly forwarded via the OVS bridge on the node.
- Across nodes: Packets are encapsulated in the OVS bridge and then forwarded to the peer node.

Advantages and Disadvantages

Advantages: Not limited by the VPC quotas and response speed (number of routes, number of ENIs, and creation speed).

Disadvantages: High encapsulation overhead, complex networking, and failure to use the load balancing and security group capabilities provided by the VPC.

Applicable Scenarios

This model is suitable to common container services that do not have high requirements on network latency and bandwidth.

Applicable Clusters

CCE cluster
### 8.2.3 VPC Network

In the VPC network model, the container network uses VPC routing to integrate with the underlying network. This network model is suitable for performance-intensive scenarios. The maximum number of nodes allowed in a cluster depends on the VPC route quota. Each node is assigned a CIDR block of a fixed size. This networking model is free from tunnel encapsulation overhead and outperforms the container tunnel network model. In addition, as VPC routing includes routes to node IP addresses and the container CIDR block, container pods in a cluster can be directly accessed from outside the cluster.

**Figure 8-3 VPC network model**

#### Pod-to-pod communication

- On the same node: Packets are directly forwarded through IPVlan.
- Across nodes: Packets are forwarded to the default gateway through default routes, and then to the peer node via the VPC routes.

#### Advantages and Disadvantages

**Advantage:** The native VPC networking functions can be used, providing high performance.

**Disadvantage:** The number of nodes is limited by the VPC route quota. Each node is assigned a CIDR block of a fixed size.
Applicable Scenarios

Scenarios that have high requirements on network latency, bandwidth, and performance

Applicable Clusters

CCE clusters

8.3 Services

8.3.1 Overview

Direct Access to a Pod

After a pod is created, the following problems may occur if you directly access the pod:

- The pod can be deleted and recreated at any time by a controller such as a Deployment, and the result of accessing the pod becomes unpredictable.
- The IP address of the pod is allocated only after the pod is started. Before the pod is started, the IP address of the pod is unknown.
- An application is usually composed of multiple pods that run the same image. Accessing pods one by one is not efficient.

For example, an application uses Deployments to create the frontend and backend. The frontend calls the backend for computing, as shown in Figure 8-4. Three pods are running in the backend, which are independent and replaceable. When a backend pod is re-created, the new pod is assigned with a new IP address, of which the frontend pod is unaware.

Figure 8-4 Inter-pod access

Using Services for Pod Access

Kubernetes Services are used to solve the preceding pod access problems. A Service has a fixed IP address. (When a CCE cluster is created, a Service CIDR block
is set, which is used to allocate IP addresses to Services.) A Service forwards requests accessing the Service to pods based on labels, and at the same time, perform load balancing for these pods.

In the preceding example, a Service is added for the frontend pod to access the backend pods. In this way, the frontend pod does not need to be aware of the changes on backend pods, as shown in Figure 8-5.

**Figure 8-5 Accessing pods through a Service**

---

**Service Types**

Kubernetes allows you to specify a Service of a required type. The values and actions of different types of Services are as follows:

- **ClusterIP**
  A ClusterIP Service allows workloads in the same cluster to use their cluster-internal domain names to access each other.

- **NodePort**
  A NodePort Service is exposed on each node's IP at a static port. A ClusterIP Service, to which the NodePort Service routes, is automatically created. By requesting `<NodeIP>:<NodePort>`, you can access a NodePort Service from outside the cluster.

- **LoadBalancer**
  A workload can be accessed from public networks through a load balancer, which is more secure and reliable than EIP.

- **DNAT**
  A DNAT gateway translates addresses for cluster nodes and allows multiple cluster nodes to share an EIP. DNAT Services provide higher reliability than EIP-based NodePort Services. You do not need to bind an EIP to a single node and requests can still be distributed to the workload even any of the nodes insides is down.
8.3.2 Intra-Cluster Access (ClusterIP)

Scenario

ClusterIP Services allow workloads in the same cluster to use their cluster-internal domain names to access each other.

The cluster-internal domain name format is `<Service name>.<Namespace of the workload>.svc.cluster.local:<Port>`, for example, `nginx.default.svc.cluster.local:80`.

*Figure 8-6* shows the mapping relationships between access channels, container ports, and access ports.

*Figure 8-6* Intra-cluster access (ClusterIP)

Adding a Service When Creating a Workload

You can set the access type (Service) when creating a workload on the CCE console.

**Step 1** In the Set Application Access step of *Creating a Deployment*, *Creating a StatefulSet*, or *Creating a DaemonSet*, click Add Service and set the following parameters:

- **Access Type**: Select ClusterIP.
- **Service Name**: Specify a Service name, which can be the same as the workload name.
- **IPv6**: This function is disabled by default. After this function is enabled, the cluster IP address of the Service changes to an IPv6 address. For details, see *How Do I Create an IPv4/IPv6 Dual-Stack Cluster?* This parameter is available only in clusters of v1.15 or later with IPv6 enabled (set during cluster creation).
- **Port Settings**
  - **Protocol**: protocol used by the Service.
  - **Container Port**: port on which the workload listens. The Nginx application listens on port 80.
  - **Access Port**: a port mapped to the container port at the cluster-internal IP address. The workload can be accessed at <cluster-internal IP address>:<access port>. The port number range is 1–65535.

*Figure 8-7 Adding a ClusterIP Service*

Step 2 After the configuration, click **OK** and then **Next: Configure Advanced Settings**. On the page displayed, click **Create**.

Step 3 Click **View Deployment Details** or **View StatefulSet Details**. On the **Services** tab page, obtain the access address, for example, 10.247.74.100:8080.

----End

**Adding a Service After Creating a Workload**

You can set the Service after creating a workload. This has no impact on the workload status and takes effect immediately. The procedure is as follows:

Step 1 Log in to the CCE console. In the navigation pane, choose **Workloads > Deployments**. On the workload list, click the name of the workload for which you will create a Service.

Step 2 On the **Services** tab page, click **Add Service**.

Step 3 On the **Create Service** page, select **ClusterIP** from the **Access Type** drop-down list.

Step 4 Set intra-cluster access parameters.

- **Service Name**: Service name, which can be the same as the workload name.
- **Cluster Name**: name of the cluster where the workload runs. The value is inherited from the workload creation page and cannot be changed.
- **Namespace**: namespace where the workload is located. The value is inherited from the workload creation page and cannot be changed.
**Workload**: workload for which you want to add a Service.

**IPv6**: This function is disabled by default. After this function is enabled, the cluster IP address of the Service changes to an IPv6 address. For details, see *How Do I Create an IPv4/IPv6 Dual-Stack Cluster?* This parameter is available only in clusters of v1.15 or later with IPv6 enabled (set during cluster creation).

**Port Settings**
- **Protocol**: protocol used by the Service.
- **Container Port**: port on which the workload listens. The Nginx application listens on port 80.
- **Access Port**: port mapped to the container port at the cluster-internal IP address. The workload can be accessed at `<cluster-internal IP address>:<access port>`. The port number range is 1–65535.

**Step 5** Click **Create**. The ClusterIP Service will be added for the workload.

----End

**Verifying the Service**

**Step 1** Log in to any node in the cluster where the workload is located.

**Step 2** Run the **curl** command to verify whether access to the workload is successful. You can perform the verification by using the IP address or domain name.

**Method 1: Use the IP address.**

```bash
curl 10.247.74.100:8080
```

10.247.74.100:8080 is the access address obtained in **Step 3**.

If information similar to the following is displayed, the workload is accessible.

```html>
<head>
<title>Welcome to nginx!</title>
<style>
  body {
    width: 35em;
    margin: 0 auto;
    font-family: Tahoma, Verdana, Arial, sans-serif;
  }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and working. Further configuration is required.</p>
<p>For online documentation and support please refer to
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>
</body>
</html>
```

**Method 2: Access the container and verify the domain name in the container.**

```bash
curl nginx.default.svc.cluster.local:8080
```
nginx.default.svc.cluster.local is the domain name obtained in Step 3.

If information similar to the following is displayed, the workload is accessible.

```html
<html>
<head>
<title>Welcome to nginx!</title>
<style>
  body {
    width: 35em;
    margin: 0 auto;
    font-family: Tahoma, Verdana, Arial, sans-serif;
  }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and working. Further configuration is required.</p>
<p>For online documentation and support please refer to <a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at <a href="http://nginx.com/">nginx.com</a>.</p>
<p>Thank you for using nginx.</p>
</body>
</html>
```

----End

Setting the Access Type Using kubectl

You can run kubectl commands to set the access type (Service). This section uses an Nginx workload as an example to describe how to implement intra-cluster access using kubectl.

**Step 1** Use kubectl to connect to the cluster. For details, see [Connecting to a Cluster Using kubectl](#).

**Step 2** Create and edit the nginx-deployment.yaml and nginx-clusterip-svc.yaml files.

The file names are user-defined. nginx-deployment.yaml and nginx-clusterip-svc.yaml are merely example file names.

```yaml
vi nginx-deployment.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx
spec:
  replicas: 1
  selector:
    matchLabels:
    app: nginx
strategy:
  type: RollingUpdate
template:
  metadata:
    labels:
      app: nginx
  spec:
    containers:
      - image: nginx
```

```yaml
vi nginx-clusterip-svc.yaml
apiVersion: v1
kind: Service
metadata:
  name: nginx
spec:
  type: ClusterIP
  selector:
    app: nginx
  ports:
  - name: http
    port: 80
    targetPort: 80
```
```yaml
imagePullSecrets:
  - name: default-secret

vi nginx-ClusterIp-svc.yaml

apiVersion: v1
kind: Service
metadata:
  labels:
    app: nginx
    name: nginx-clusterip
spec:
  ports:
    - name: service0
      port: 8080
      protocol: TCP
      targetPort: 80
  selector:
    app: nginx
    type: ClusterIP
```

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Mandatory</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>port</td>
<td>Yes</td>
<td>Integer</td>
<td>Port at which the Service is exposed, that is, the access port on the CCE console.</td>
</tr>
<tr>
<td>targetPort</td>
<td>Yes</td>
<td>Integer</td>
<td>Container port on the CCE console.</td>
</tr>
<tr>
<td>type</td>
<td>No</td>
<td>String</td>
<td>Access type on the CCE console. The options are as follows:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• ClusterIP</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• NodePort</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• LoadBalancer</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>The default value is <strong>ClusterIP</strong>, indicating the cluster-internal IP address.</td>
</tr>
</tbody>
</table>

**Step 3** Create a workload.

```bash
kubectl create -f nginx-deployment.yaml
```

If information similar to the following is displayed, the workload is being created.

```
deployment “nginx” created
```

**Step 4** Create a Service.

```bash
kubectl create -f nginx-ClusterIp-svc.yaml
```

If information similar to the following is displayed, the workload is running.

```
NAME                     READY     STATUS             RESTARTS   AGE
etcd-0                   0/1         ImagePullBackOff 0  27m
icagent-m9dkt            0/0         Running            0          3d
nginx-2601814895-znhbr   1/1         Running            0          15s
```
If information similar to the following is displayed, the Service is being created.

```
service "nginx-clusterip" created
```

**kubectl get svc**

If information similar to the following is displayed, the Service has been created, and a cluster-internal IP address has been assigned to the Service.

<table>
<thead>
<tr>
<th>NAME</th>
<th>TYPE</th>
<th>CLUSTER-IP</th>
<th>EXTERNAL-IP</th>
<th>PORT(S)</th>
<th>AGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>etcd-svc</td>
<td>ClusterIP</td>
<td>None</td>
<td>&lt;none&gt;</td>
<td>3120/TCP</td>
<td>30m</td>
</tr>
<tr>
<td>kubernetes</td>
<td>ClusterIP</td>
<td>10.247.0.1</td>
<td>&lt;none&gt;</td>
<td>443/TCP</td>
<td>3d</td>
</tr>
<tr>
<td>nginx-clusterip</td>
<td>ClusterIP</td>
<td>10.247.200.134</td>
<td>&lt;none&gt;</td>
<td>80/TCP</td>
<td>20s</td>
</tr>
</tbody>
</table>

**Step 5** Log in to any node in the cluster where the workload is located. For details, see [Logging In to a Linux ECS](#).

**Step 6** Run the `curl` command to check whether the workload is accessible. You can perform the verification by using the IP address or domain name.

**Method 1: Use the IP address.**

```
curl 10.247.200.134:80
```

If information similar to the following is displayed, the workload is accessible.

```html
<html>
<head>
<title>Welcome to nginx!</title>
<style>
body { 
    width: 35em;
    margin: 0 auto;
    font-family: Tahoma, Verdana, Arial, sans-serif;
}
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and working. Further configuration is required.</p>
<p>For online documentation and support please refer to <a href="http://nginx.org">nginx.org</a>. Commercial support is available at <a href="http://nginx.com">nginx.com</a>.</p>
<p>Thank you for using nginx.</p>
</body>
</html>
```

**Method 2: Use the domain name.**

```
curl nginx-clusterip.default.svc.cluster.local:8080
```

If information similar to the following is displayed, the workload is accessible.

```html
<html>
<head>
<title>Welcome to nginx!</title>
<style>
body { 
    width: 35em;
    margin: 0 auto;
    font-family: Tahoma, Verdana, Arial, sans-serif;
}
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
```
### 8.3.3 NodePort

**Scenario**

A Service is exposed on each node's IP address at a static port (NodePort). A ClusterIP Service, to which the NodePort Service will route, is automatically created. By requesting `<NodeIP>:<NodePort>`, you can access a NodePort Service from outside the cluster.

![NodePort access](image)

**Notes and Constraints**

- By default, a NodePort Service is accessed within a VPC. If you need to use an EIP to access a NodePort Service through public networks, bind an EIP to the node in the cluster in advance.
- After a Service is created, if the **affinity setting** is switched from the cluster level to the node level, the connection tracing table will not be cleared. You are advised not to modify the Service affinity setting after the Service is created. If you need to modify it, create a Service again.
- Containers on the same node cannot access Services whose **externalTrafficPolicy** is **local**.
- The Service port of a NodePort Service created on the CCE console is the same as the configured container port.

Adding a Service When Creating a Workload

You can set the access type when creating a workload on the CCE console. An Nginx workload is used as an example.

Step 1  In the Set Application Access step of Creating a Deployment, Creating a StatefulSet, or Creating a DaemonSet, click Add Service and set the following parameters:

- **Access Type**: Select NodePort.
  
  **NOTE**
  
  If you want to use an EIP to access a NodePort Service through public networks, bind an EIP to the node in the cluster in advance.

- **Service Name**: Specify a Service name, which can be the same as the workload name.

- **Service Affinity**
  
  - Cluster level: The IP addresses and access ports of all nodes in a cluster can access the workload associated with the Service. Service access will cause performance loss due to route redirection, and the source IP address of the client cannot be obtained.
  
  - Node level: Only the IP address and access port of the node where the workload is located can access the workload associated with the Service. Service access will not cause performance loss due to route redirection, and the source IP address of the client can be obtained.

- **IPv6**: This function is disabled by default. After this function is enabled, the cluster IP address of the Service changes to an IPv6 address. For details, see How Do I Create an IPv4/IPv6 Dual-Stack Cluster? This parameter is available only in clusters of v1.15 or later with IPv6 enabled (set during cluster creation).

- **Port Settings**
  
  - **Protocol**: protocol used by the Service.
  
  - **Container Port**: port on which the workload in the container image listens. The value ranges from 1 to 65535.
  
  - **Access Port**: node port (with a private IP address) to which the container port will be mapped. You are advised to select Automatically generated.
    
    - **Automatically generated**: The system automatically assigns a port number.
    
    - **Specified port**: You have to manually specify a fixed node port number in the range of 30000–32767. Ensure that the port is unique in a cluster.

Step 2  After the configuration is complete, click OK.

Step 3  Click Next: Configure Advanced Settings. On the page displayed, click Create.
**Step 4** Click **View Deployment Details** or **View StatefulSet Details**. On the **Services** tab page, obtain the access address, for example, 192.168.0.160:30358.

----End

### Adding a Service After Creating a Workload

You can set the Service after creating a workload. This has no impact on the workload status and takes effect immediately. The procedure is as follows:

**Step 1** Log in to the CCE console. In the navigation pane, choose **Workloads > Deployments**. On the workload list, click the name of the workload for which you will create a Service.

.general tip

If the Service is associated with an ingress, the ingress is unavailable after the port information of the Service is updated. In this case, you need to delete and recreate the Service.

**Step 2** On the **Services** tab page, click **Add Service**.

**Step 3** On the **Create Service** page, select **NodePort** from the **Access Type** drop-down list.

.general tip

If you want to use an EIP to access a NodePort Service through public networks, bind an EIP to the node in the cluster in advance.

**Step 4** Set node access parameters.

- **Service Name**: Service name, which can be the same as the workload name.
- **Cluster Name**: name of the cluster where the workload runs. The value is inherited from the workload creation page and cannot be changed.
- **Namespace**: namespace where the workload is located. The value is inherited from the workload creation page and cannot be changed.
- **Workload**: workload for which you want to add a Service. The value is inherited from the workload creation page and cannot be changed.
- **Service Affinity**
  - Cluster level: The IP addresses and access ports of all nodes in a cluster can access the workload associated with the Service. Service access will cause performance loss due to route redirection, and the source IP address of the client cannot be obtained.
  - Node level: Only the IP address and access port of the node where the workload is located can access the workload associated with the Service. Service access will not cause performance loss due to route redirection, and the source IP address of the client can be obtained.
- **IPv6**: This function is disabled by default. After this function is enabled, the cluster IP address of the Service changes to an IPv6 address. For details, see **How Do I Create an IPv4/IPv6 Dual-Stack Cluster?** This parameter is available only in clusters of v1.15 or later with IPv6 enabled (set during cluster creation).
- **Port Settings**
- **Protocol**: protocol used by the Service.
- **Container Port**: port on which the workload in the container image listens. The Nginx workload listens on port 80.
- **Access Port**: node port (with a private IP address) to which the container port will be mapped. You are advised to select **Automatically generated**.
  - **Automatically generated**: The system automatically assigns a port number.
  - **Specified port**: You have to manually specify a fixed node port number in the range of 30000–32767. Ensure that the port is unique in a cluster.

**Step 5** Click **Create**. A NodePort Service will be added for the workload.

---End

### Verifying the Service

**Step 1** On the homepage of management console, choose **Computing > Elastic Cloud Server**.

**Step 2** Select any ECS in the same VPC as the workload that will be accessed, and confirm that the security group is open to the IP address and port to be connected.

**Figure 8-9** Confirming that the security group is open

**Step 3** Click **Remote Login**. On the login page, enter the username and password.

**Step 4** Run the **curl** command to check whether access to the workload is successful.

**NOTE**

If you want to use an EIP to access a NodePort Service through public networks, bind an EIP to the node in the cluster in advance.

```bash
curl 192.168.0.160:30358
```

In the preceding command, **192.168.0.160:30358** indicates the access address obtained in **Step 4**, that is, the node virtual IP address and the access port.

If information similar to the following is displayed, the workload is accessible.

```
<html>
<head>
<title>Welcome to nginx!</title>
<style>
body {
  width: 35em;
  margin: 0 auto;
  font-family: Tahoma, Verdana, Arial, sans-serif;
}
```
Using kubectl

You can run kubectl commands to set the access type. This section uses a Nginx workload as an example to describe how to set a NodePort Service using kubectl.

**Step 1** Use kubectl to connect to the cluster. For details, see Connecting to a Cluster Using kubectl.

**Step 2** Create and edit the `nginx-deployment.yaml` and `nginx-nodeport-svc.yaml` files.

The file names are user-defined. `nginx-deployment.yaml` and `nginx-nodeport-svc.yaml` are merely example file names.

**vi nginx-deployment.yaml**

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx
spec:
  replicas: 1
  selector:
    matchLabels:
      app: nginx
  strategy:
    type: RollingUpdate
    template:
      metadata:
        labels:
          app: nginx
      spec:
        containers:
          - image: nginx
```

**vi nginx-nodeport-svc.yaml**

```yaml
apiVersion: v1
kind: Service
metadata:
  labels:
    app: nginx
    name: nginx-nodeport
spec:
  ports:
```
- name: service
  nodePort: 30000
  port: 80
  protocol: TCP
  targetPort: 80
  selector:
    app: nginx
    type: NodePort

**Table 8-3** Key parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Mandatory</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>nodePort</td>
<td>No</td>
<td>Integer</td>
<td>Access port set on the console. The value ranges from 30000 to 32767. If this parameter is left unspecified, the value is automatically generated.</td>
</tr>
<tr>
<td>port</td>
<td>Yes</td>
<td>Integer</td>
<td>Access port of the cluster-internal IP address. The value ranges from 1 to 65535.</td>
</tr>
<tr>
<td>protocol</td>
<td>No</td>
<td>String</td>
<td>IP protocol used by the port. The value can be <strong>TCP</strong> or <strong>UDP</strong>. Default value: <strong>TCP</strong></td>
</tr>
<tr>
<td>targetPort</td>
<td>Yes</td>
<td>String</td>
<td>Container port set on the console. The value ranges from 1 to 65535.</td>
</tr>
<tr>
<td>type</td>
<td>Yes</td>
<td>String</td>
<td>Access type set on the console. NodePort indicates the private IP address of a node.</td>
</tr>
</tbody>
</table>
Table 8-4 Data structure of the metadata.annotations field (Health check and sticky session must be enabled for interconnection with ingresses. The following table lists the required fields and descriptions.)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Mandatory</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>kubernetes.io/elb.session-affinity-mode</td>
<td>No</td>
<td>String</td>
<td>Listeners ensure session stickiness based on IP addresses. Requests from the same IP address will be forwarded to the same backend server.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Disabling sticky session: Do not set this parameter.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Enabling sticky session: Set this parameter to SOURCE_IP, indicating that the sticky session is based on the source IP address.</td>
</tr>
<tr>
<td>kubernetes.io/elb.session-affinity-option</td>
<td>No</td>
<td>Table 8-5</td>
<td>This parameter indicates the sticky session configuration for layer-7 load balancing.</td>
</tr>
<tr>
<td>kubernetes.io/elb.lb-algorithm</td>
<td>No</td>
<td>String</td>
<td>This parameter indicates the load balancing algorithm of the backend server group. The default value is ROUND_ROBIN.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Options:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• ROUND_ROBIN: weighted round robin algorithm</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• LEAST_CONNECTIONS: weighted least connections algorithm</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• SOURCE_IP: source IP hash algorithm</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>When the value is SOURCE_IP, the weights of backend servers in the server group are invalid.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Mandatory</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>-----------</td>
<td>----------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| kubernetes.io/elb.health-check-flag | No        | String   | This parameter indicates whether to enable the ELB health check function. The health check is enabled by default.  
- Enabling health check: Leave blank this parameter or set it to `on`.  
- Disabling health check: Set this parameter to `off`. |
| kubernetes.io/elb.health-check-option | No        | Table 8-6 Object | This parameter indicates the ELB health check configuration. |

### Table 8-5 Data structure of the elb.session-affinity-option field

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Mandatory</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| persistence_timeout | No        | String   | Sticky session timeout, in seconds. This parameter is valid only when `elb.session-affinity-mode` is set to `HTTP_COOKIE`.  
Value range: 1 to 1440. Default value: **1440** |
| app_cookie_name    | Yes       | String   | Sticky session timeout, in seconds. This parameter is valid only when `elb.session-affinity-mode` is set to `APP_COOKIE`.  
The value can contain 1 to 10,000 characters. |
Table 8-6 Data structure description of the elb.health-check-option field

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Mandatory</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>delay</td>
<td>Yes</td>
<td>String</td>
<td>Initial waiting time (in seconds) for starting the health check. This parameter is optional. Value range: 1 to 50. Default value: 5</td>
</tr>
<tr>
<td>timeout</td>
<td>Yes</td>
<td>String</td>
<td>Health check timeout, in seconds. This parameter is optional. Value range: 1 to 50. Default value: 10</td>
</tr>
<tr>
<td>max_retries</td>
<td>Yes</td>
<td>String</td>
<td>Maximum number of health check retries. This parameter is optional. Value range: 1 to 10. Default value: 3</td>
</tr>
<tr>
<td>protocol</td>
<td>Yes</td>
<td>String</td>
<td>Health check protocol. This parameter is optional. Default value: protocol of the associated Service Value options: TCP or HTTP</td>
</tr>
<tr>
<td>path</td>
<td>Yes</td>
<td>String</td>
<td>Health check URL. This parameter is optional, and needs to be configured when the protocol is HTTP. Default value: / The value can contain 1 to 10,000 characters.</td>
</tr>
</tbody>
</table>

Step 3  Create a workload.

```
kubectl create -f nginx-deployment.yaml
```

If information similar to the following is displayed, the workload is being created.

```
deployment “nginx” created
```

```
kubectl get po
```

If information similar to the following is displayed, the workload is running.

```
NAME                     READY     STATUS             RESTARTS   AGE
etcd-0                   0/1        ImagePullBackOff 0          48m
icagent-m9dkt            0/0        Running            0          3d
nginx-2601814895-qhxqv   1/1        Running            0          9s
```

Step 4  Create a Service.

```
kubectl create -f nginx-nodeport-svc.yaml
```

If information similar to the following is displayed, the Service is being created.
**service "nginx-nodeport" created**

```
kubectl get svc
```

If information similar to the following is displayed, the Service has been created.

<table>
<thead>
<tr>
<th>NAME</th>
<th>TYPE</th>
<th>CLUSTER-IP</th>
<th>EXTERNAL-IP</th>
<th>PORT(S)</th>
<th>AGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>etcd-svc</td>
<td>ClusterIP</td>
<td>None</td>
<td>&lt;none&gt;</td>
<td>3120/TCP</td>
<td>49m</td>
</tr>
<tr>
<td>kubernetes</td>
<td>ClusterIP</td>
<td>10.247.0.1</td>
<td>&lt;none&gt;</td>
<td>443/TCP</td>
<td>3d</td>
</tr>
<tr>
<td>nginx-nodeport</td>
<td>NodePort</td>
<td>10.247.4.225</td>
<td>&lt;none&gt;</td>
<td>80:30000/TCP</td>
<td>7s</td>
</tr>
</tbody>
</table>

**Step 5** Run the `curl` command to verify whether the workload is accessible.

```
curl 192.168.2.240:30000
```

*192.168.2.240* is the IP address of any node in the cluster, and *30000* is the port opened on all nodes in the cluster.

If information similar to the following is displayed, the workload is accessible.

```html
<html>
<head>
<title>Welcome to nginx!</title>
<style>
body {
  width: 35em;
  margin: 0 auto;
  font-family: Tahoma, Verdana, Arial, sans-serif;
}
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and working. Further configuration is required.</p>
<p>For online documentation and support please refer to <a href="http://nginx.org/">nginx.org</a>.<br/>Commercial support is available at <a href="http://nginx.com/">nginx.com</a>.</p>
<p>Thank you for using nginx.</p>
</body>
</html>
```

----End

**8.3.4 LoadBalancer**

**Scenario**

A workload can be accessed from public networks through a load balancer. LoadBalancer provides higher reliability than EIP-based NodePort because an EIP is no longer bound to a single node. The LoadBalancer access type is applicable to the scenario in which a Service exposed to public networks is required.

The access address is in the format of `<IP address of public network load balancer>:<access port>`. For example, `10.117.117.117:80`.

In this access mode, requests are transmitted through an ELB load balancer to a node and then forwarded to the destination pod through the Service.
Notes and Constraints

- LoadBalancer Services allow workloads to be accessed from public networks through ELB. This access mode has the following restrictions:
  - It is recommended that automatically created load balancers not be used by other resources. Otherwise, these load balancers cannot be completely deleted, causing residual resources.
  - Do not change the listener name for the load balancer in use. Otherwise, the load balancer cannot be accessed.

- After a Service is created, if the affinity setting is switched from the cluster level to the node level, the connection tracing table will not be cleared. You are advised not to modify the Service affinity setting after the Service is created. If you need to modify it, create a Service again.

- The Service whose externalTrafficPolicy is local cannot be accessed.

- Dedicated ELB load balancers can be used only in clusters of v1.17 and later.

- If you create a LoadBalancer Service on the CCE console, a random node port is automatically generated. If you use kubectl to create a LoadBalancer Service, a random node port is generated unless you specify one.

Adding a Service When Creating a Workload

You can set the Service when creating a workload on the CCE console. An Nginx workload is used as an example.
Step 1  In the Set Application Access step of Creating a Deployment, Creating a StatefulSet, or Creating a DaemonSet, click Add Service and set the following parameters:

- **Access Type:** Select LoadBalancer (ELB).
- **Service Name:** Specify a Service name, which can be the same as the workload name.
- **Service Affinity:**
  - Cluster level: The IP addresses and access ports of all nodes in a cluster can access the workload associated with the Service. Service access will cause performance loss due to route redirection, and the source IP address of the client cannot be obtained.
  - Node level: Only the IP address and access port of the node where the workload is located can access the workload associated with the Service. Service access will not cause performance loss due to route redirection, and the source IP address of the client can be obtained.

**ELB Configuration**

- **Elastic Load Balancer:** You can customize your load balancer. For the differences between load balancer types, see Differences Between Shared and Dedicated Load Balancers.
  - Shared load balancers provide domain name- and URL-based route balancing. Resources are shared among load balancers, and the performance of a load balancer is affected by other load balancers.
  - **Dedicated:** Resources are shared among load balancers, and the performance of a load balancer is not affected by other load balancers. IPv6 is supported.
    - **AZ:** Dedicated load balancers can be deployed across AZs to provide higher reliability.
    - **Subnet:** subnet where the backend server of the load balancer is located.
      Load balancers occupy different number of subnet IP addresses based on their specifications. Therefore, you are not advised to use the subnet CIDR blocks of other resources (such as clusters and nodes) as the load balancer subnet CIDR block.
    - **Specifications:** Specifications determine the types of listeners that can be added to a load balancer. Select specifications that best fit your needs. For details, see Specifications of Dedicated Load Balancers.

You can create public network or private network load balancers.

- **Public network:** You can select an existing public network load balancer or have the system automatically create a new one.
- **Private network:** You can select an existing private network load balancer or have the system automatically create a new private network load balancer.

The selected or created load balancer must be in the same VPC as the current cluster, and it must match the load balancer type (private or public network).
- **Enterprise Project**: Select an enterprise project in which the load balancer is created.

- **Specifications**: This field is displayed only when you select **Public network** and **Automatically created** for **Elastic Load Balancer**. You can click to modify the name, specifications, billing mode, and bandwidth of the load balancer.

- **Algorithm Type**: **Weighted round robin**, **Weighted least connections**, and **Source IP hash** are available.

  **NOTE**

  - **Weighted round robin**: Requests are forwarded to different servers based on their weights, which indicate server processing performance. Backend servers with higher weights receive proportionately more requests, whereas equal-weighted servers receive the same number of requests. This algorithm is often used for short connections, such as HTTP services.

  - **Weighted least connections**: In addition to the weight assigned to each server, the number of connections processed by each backend server is also considered. Requests are forwarded to the server with the lowest connections-to-weight ratio. Building on **least connections**, the **weighted least connections** algorithm assigns a weight to each server based on their processing capability. This algorithm is often used for persistent connections, such as database connections.

  - **Source IP hash**: The source IP address of each request is calculated using the hash algorithm to obtain a unique hash key, and all backend servers are numbered. The generated key allocates the client to a particular server. This enables requests from different clients to be distributed in load balancing mode and ensures that requests from the same client are forwarded to the same server. This algorithm applies to TCP connections without cookies.

- **Sticky Session**: This function is disabled by default. You can select **Based on source IP address**. Listeners ensure session stickiness based on IP addresses. Requests from the same IP address will be forwarded to the same backend server.

- **Health Check**: This function is enabled by default. Enabling it will perform health checks on your load balancer. For details about how to configure the ELB health check parameters, see [Configuring a Health Check](#).

- **Port Settings**
  - **Protocol**: protocol used by the Service.
  - **Container Port**: port defined in the container image and on which the workload listens. The Nginx application listens on port 80.
  - **Access Port**: port mapped to the container port at the load balancer's IP address. The workload can be accessed at `<Load balancer's IP address>:<Access port>`. The port number range is 1–65535.

**Step 2** After the configuration is complete, click **OK**.

**Step 3** On the workload creation page, click **Next: Configure Advanced Settings**. On the page displayed, click **Create**.

**Step 4** After the workload is successfully created, choose **Workloads > Deployments** or **Workloads > StatefulSets** on the CCE console. Click the name of the workload to
show the details of the workload. On the workload details page, click the **Services** tab and obtain the access address, as shown in Figure 8-11.

**Figure 8-11 Obtaining the load balancer IP address**

![Obtaining the load balancer IP address](image)

**Adding a Service After Creating a Workload**

You can set the Service after creating a workload. This has no impact on the workload status and takes effect immediately. The procedure is as follows:

**Step 1** Log in to the CCE console. In the navigation pane, choose **Workloads** > **Deployments** or **Workloads** > **StatefulSets**. On the workload list, click the name of the workload for which you will add a Service.

**Step 2** On the **Services** tab page, click **Create Service**.

The parameters are the same as those in **Adding a Service When Creating a Workload**.

**Step 3** Click **Create**.

----End

**Using kubectl to Create a Service (Using an Existing Load Balancer)**

You can set the access type when creating a workload using kubectl. This section uses a Nginx workload as an example to describe how to add a LoadBalancer Service using kubectl.

**Step 1** Use kubectl to connect to the cluster. For details, see **Connecting to a Cluster Using kubectl**.

**Step 2** Create and edit the **nginx-deployment.yaml** and **nginx-elb-svc.yaml** files.

The file names are user-defined. **nginx-deployment.yaml** and **nginx-elb-svc.yaml** are merely example file names.

```bash
vi nginx-deployment.yaml
```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx
spec:
  replicas: 1
  selector:
    matchLabels:
      app: nginx
  strategy:
    type: RollingUpdate
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
        - image: nginx
          imagePullPolicy: Always
          name: nginx
          imagePullSecrets:
            - name: default-secret

vi nginx-elb-svc.yaml

NOTE

Before enabling sticky session, ensure that the following conditions are met:

- The workload protocol is TCP.
- Anti-affinity has been configured between pods of the workload. That is, all pods of the workload are deployed on different nodes. For details, see Workload-Node Anti-Affinity.

apiVersion: v1
kind: Service
metadata:
  annotations:
    kubernetes.io/elb.class: union
    kubernetes.io/session-affinity-mode: SOURCE_IP
    kubernetes.io/elb.id: 3c7caaa5-a641-4b9f-801a-feace27424b6
    kubernetes.io/elb.subnet-id: 5083f225-9bf8-48fa-9c8b-67bd9693c4c0
  labels:
    app: nginx
    name: nginx
spec:
  loadBalancerIP: 10.78.42.242
  externalTrafficPolicy: Local
  ports:
    - name: service0
      port: 80
      protocol: TCP
      targetPort: 80
    selector:
      app: nginx
type: LoadBalancer
### Table 8-7 Key parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Mandatory</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>kubernetes.io/elb.class</td>
<td>No</td>
<td>String</td>
<td>Select a proper load balancer type as required. The value can be:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- union: shared load balancer</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- performance: dedicated load balancer, which can be used only in clusters</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>of v1.17 and later. For details, see Differences Between Shared and Dedicated</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Load Balancers.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Default value: union</td>
</tr>
<tr>
<td>kubernetes.io/session-affinity-</td>
<td>No</td>
<td>String</td>
<td>Listeners ensure session stickiness based on IP addresses. Requests from</td>
</tr>
<tr>
<td>mode</td>
<td></td>
<td></td>
<td>the same IP address will be forwarded to the same backend server.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- Disabling sticky session: Do not set this parameter.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- Enabling sticky session: Set this parameter to SOURCE_IP, indicating that</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>the sticky session is based on the source IP address.</td>
</tr>
<tr>
<td>kubernetes.io/elb.session-</td>
<td>No</td>
<td>Table</td>
<td>This parameter specifies the sticky session timeout.</td>
</tr>
<tr>
<td>affinity-option</td>
<td></td>
<td>8-8 Object</td>
<td></td>
</tr>
<tr>
<td>kubernetes.io/elb.id</td>
<td>Yes</td>
<td>String</td>
<td>This parameter indicates the ID of a load balancer. The value can contain 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>to 100 characters. Mandatory when an existing load balancer is to be</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>associated.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Obtaining the load balancer ID:</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>On the management console, click Service List, and choose Network &gt; Elastic</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Load Balance. Click the name of the target load balancer. On the Summary</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>tab page, find and copy the ID.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Mandatory</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>-----------</td>
<td>-----------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>kubernetes.io/elb.subnet-id</td>
<td>Yes</td>
<td>String</td>
<td>This parameter indicates the ID of the subnet where the cluster is located. The value can contain 1 to 100 characters.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>● Mandatory when a cluster of v1.11.7-r0 or earlier is to be automatically created.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>● Optional for clusters later than v1.11.7-r0.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>For details on how to obtain the subnet ID, see What Is the Difference Between the VPC Subnet API and the OpenStack Neutron Subnet API?</td>
</tr>
<tr>
<td>kubernetes.io/elb.lb-algorithm</td>
<td>No</td>
<td>String</td>
<td>This parameter indicates the load balancing algorithm of the backend server group. The default value is ROUND_ROBIN.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Options:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>● ROUND_ROBIN: weighted round robin algorithm</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>● LEAST_CONNECTIONS: weighted least connections algorithm</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>● SOURCE_IP: source IP hash algorithm</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>When the value is SOURCE_IP, the weights of backend servers in the server group are invalid.</td>
</tr>
<tr>
<td>kubernetes.io/elb.health-check-flag</td>
<td>No</td>
<td>String</td>
<td>Whether to enable the ELB health check.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Disabled by default.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>● Enabling health check: Leave blank this parameter or set it to on.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>● Disabling health check: Set this parameter to off.</td>
</tr>
<tr>
<td>kubernetes.io/elb.health-check-option</td>
<td>No</td>
<td>Table 8-9 Object</td>
<td>ELB health check configuration items.</td>
</tr>
<tr>
<td>port</td>
<td>Yes</td>
<td>Integer</td>
<td>Access port that is registered on the load balancer and mapped to the cluster-internal IP address.</td>
</tr>
<tr>
<td>targetPort</td>
<td>Yes</td>
<td>String</td>
<td>Container port set on the CCE console.</td>
</tr>
</tbody>
</table>
Table 8-8 Data structure of the `elb.session-affinity-option` field

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Mandatory</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>persistence_timeout</td>
<td>Yes</td>
<td>String</td>
<td>Sticky session timeout, in minutes. This parameter is valid only when <code>elb.session-affinity-mode</code> is set to <code>SOURCE_IP</code>. Value range: 1 to 60. Default value: 60</td>
</tr>
</tbody>
</table>

Table 8-9 Data structure description of the `elb.health-check-option` field

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Mandatory</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>delay</td>
<td>No</td>
<td>String</td>
<td>Initial waiting time (in seconds) for starting the health check. Value range: 1 to 50. Default value: 5</td>
</tr>
<tr>
<td>timeout</td>
<td>No</td>
<td>String</td>
<td>Health check timeout, in seconds. Value range: 1 to 50. Default value: 10</td>
</tr>
<tr>
<td>max_retries</td>
<td>No</td>
<td>String</td>
<td>Maximum number of health check retries. Value range: 1 to 10. Default value: 3</td>
</tr>
<tr>
<td>protocol</td>
<td>No</td>
<td>String</td>
<td>Health check protocol. Default value: protocol of the associated Service Value options: TCP, UDP_CONNECT, or HTTP</td>
</tr>
<tr>
<td>path</td>
<td>No</td>
<td>String</td>
<td>Health check URL. This parameter needs to be configured when the protocol is HTTP. Default value: / The value can contain 1 to 10,000 characters.</td>
</tr>
</tbody>
</table>

Step 3  Create a workload.

`kubectl create -f nginx-deployment.yaml`

If information similar to the following is displayed, the workload is being created.

`deployment "nginx" created`
kubectl get pod

If information similar to the following is displayed, the workload is running.

<table>
<thead>
<tr>
<th>NAME</th>
<th>READY</th>
<th>STATUS</th>
<th>RESTARTS</th>
<th>AGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>etcd-0</td>
<td>0/1</td>
<td>ImagePullBackOff</td>
<td>0</td>
<td>1h</td>
</tr>
<tr>
<td>icagent-m9dkt</td>
<td>0/0</td>
<td>Running</td>
<td>0</td>
<td>3d</td>
</tr>
<tr>
<td>nginx-2601814895-c1xhw</td>
<td>1/1</td>
<td>Running</td>
<td>0</td>
<td>6s</td>
</tr>
</tbody>
</table>

Step 4  Create a Service.

kubectl create -f nginx-elb-svc.yaml

If information similar to the following is displayed, the Service has been created.

service "nginx" created

kubectl get svc

If information similar to the following is displayed, the access type has been set successfully, and the workload is accessible.

<table>
<thead>
<tr>
<th>NAME</th>
<th>TYPE</th>
<th>CLUSTER-IP</th>
<th>EXTERNAL-IP</th>
<th>PORT(S)</th>
<th>AGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>etcd-svc</td>
<td>ClusterIP</td>
<td>None</td>
<td>&lt;none&gt;</td>
<td>3120/TCP</td>
<td>1h</td>
</tr>
<tr>
<td>kubernetes</td>
<td>ClusterIP</td>
<td>10.247.0.1</td>
<td>&lt;none&gt;</td>
<td>443/TCP</td>
<td>3d</td>
</tr>
<tr>
<td>nginx</td>
<td>LoadBalancer</td>
<td>10.247.130.196</td>
<td>10.78.42.242</td>
<td>80:31540/TCP</td>
<td>51s</td>
</tr>
</tbody>
</table>

Step 5  Enter the URL in the address box of the browser, for example, 10.78.42.242:80. 10.78.42.242 indicates the IP address of the load balancer, and 80 indicates the access port displayed on the CCE console.

The Nginx is accessible.

Figure 8-12 Accessing Nginx through the LoadBalancer Service

Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

----End

Using kubectl to Create a Service (Automatically Creating a Load Balancer)

You can add a Service when creating a workload using kubectl. This section uses a Nginx workload as an example to describe how to add a LoadBalancer Service using kubectl.

Step 1  Use kubectl to connect to the cluster. For details, see Connecting to a Cluster Using kubectl.

Step 2  Create and edit the nginx-deployment.yaml and nginx-elb-svc.yaml files.

The file names are user-defined. nginx-deployment.yaml and nginx-elb-svc.yaml are merely example file names.
vi nginx-deployment.yaml

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx
spec:
  replicas: 1
  selector:
    matchLabels:
      app: nginx
  strategy:
    type: RollingUpdate
template:
  metadata:
    labels:
      app: nginx
  spec:
    containers:
    - image: nginx
      imagePullPolicy: Always
      name: nginx
      imagePullSecrets:
      - name: default-secret
```

vi nginx-elb-svc.yaml

```yaml
# NOTE

Before enabling sticky session, ensure that the following conditions are met:

- The workload protocol is TCP.
- Anti-affinity has been configured between pods of the workload. That is, all pods of the workload are deployed on different nodes. For details, see Workload-Node Anti-Affinity.

Example of a Service using a shared, public network load balancer:

```yaml
apiVersion: v1
kind: Service
metadata:
  annotations:
    kubernetes.io/elb.class: union
    kubernetes.io/session-affinity-mode: SOURCE_IP
    kubernetes.io/elb.subnet-id: ca5b861e-4e13-480a-996a-6c84c1d9538d
    kubernetes.io/elb.enterpriseID: '0'
  labels:
    app: nginx
  name: nginx
spec:
  externalTrafficPolicy: Local
  ports:
    - name: service0
      port: 80
      protocol: TCP
      targetPort: 80
  selector:
    app: nginx
  type: LoadBalancer
```
Example Service using a public network dedicated load balancer (for clusters of v1.17 and later only):

```yaml
apiVersion: v1
kind: Service
metadata:
  name: nginx
  labels:
    app: nginx
  namespace: default
annotations:
  kubernetes.io/elb.class: performance
  kubernetes.io/elb.subnet-id: ca5b861e-4e13-480a-996a-6c84c1d9538d
  kubernetes.io/elb.enterpriseID: '0'
  kubernetes.io/elb.autocreate:
    '{
      "type": "public",
      "bandwidth_name": "cce-bandwidth-1626694478577",
      "bandwidth_chargemode": "bandwidth",
      "bandwidth_size": 1,
      "bandwidth_sharetype": "PER",
      "eip_type": "5_gray",
      "available_zone": [
        "cn-south-2b",
        "cn-south-1c"
      ],
      "l4_flavor_name": "L4_flavor.elb.s1.small",
      "elb_virsubnet_ids": [
        "14567f27-8ae4-42b8-ae47-9f847a4690dd"
      ]
    }'
  kubernetes.io/elb.lb-algorithm: ROUND_ROBIN
  kubernetes.io/elb.health-check-flag: 'on'
  kubernetes.io/elb.health-check-option: '{"protocol":"TCP","delay":"5","timeout":"10","max_retries":"3"}'
spec:
  selector:
    app: nginx
  externalTrafficPolicy: Local
  ports:
  - name: cce-service-0
    targetPort: 80
    nodePort: 0
    port: 80
    protocol: TCP
    type: LoadBalancer
```

**Table 8-10 Key parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Mandatory</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| kubernetes.io/elb.class | No        | String   | Select a proper load balancer type as required. The value can be:  
- union: shared load balancer  
- performance: dedicated load balancer, which can be used only in clusters of v1.17 and later. For details, see [Differences Between Shared and Dedicated Load Balancers](#).  
Default value: union |
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Mandatory</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| kubernetes.io/elb.subnet-id|           | String       | This parameter indicates the ID of the subnet where the cluster is located. The value can contain 1 to 100 characters.  
|                            |           |              | - Mandatory when a cluster of v1.11.7-r0 or earlier is to be automatically created.               |
|                            |           |              | - Optional for clusters later than v1.11.7-r0.                                                   |
|                            |           |              | For details about how to obtain the value, see [What Is the Difference Between the VPC Subnet API and the OpenStack Neutron Subnet API](#). |
| kubernetes.io/elb.enterpriseID | No       | String       | Clusters of v1.15 and later versions support this field. In clusters earlier than v1.15, load balancers are created in the default project by default.  
|                            |           |              | This parameter indicates the ID of the enterprise project in which the ELB load balancer will be created.  
|                            |           |              | If this parameter is not specified or is set to 0, resources will be bound to the default enterprise project.  
|                            |           |              | **How to obtain:**  
<p>|                            |           |              | Log in to the management console and choose Enterprise &gt; Project Management on the top menu bar. In the list displayed, click the name of the target enterprise project, and copy the ID on the enterprise project details page.  |
| kubernetes.io/elb.session-affinity-option | No | Table 8-8 Object | Sticky session timeout.  |</p>
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Mandatory</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| kubernetes.io/elb.autocreate | Yes       | elb.autocreate      | Whether to automatically create a load balancer associated with the Service. **Example:**  
  - Automatically created public network load balancer: 
    ```json
    {
      "type": "public",
      "bandwidth_name": "cce-bandwidth-1551163379627",
      "bandwidth_chargemode": "bandwidth",
      "bandwidth_size": 5,
      "bandwidth_sharetype": "PER",
      "eip_type": "5_bgp",
      "name": "james"
    }
    ```  
  - Automatically created private network load balancer: 
    ```json
    {
      "type": "inner",
      "name": "A-location-d-test"
    }
    ```
| kubernetes.io/elb.lb-algorithm | No        | String              | This parameter indicates the load balancing algorithm of the backend server group. The default value is **ROUND_ROBIN**. Options:  
  - **ROUND_ROBIN**: weighted round robin algorithm  
  - **LEAST_CONNECTIONS**: weighted least connections algorithm  
  - **SOURCE_IP**: source IP hash algorithm  
    When the value is **SOURCE_IP**, the weights of backend servers in the server group are invalid. |
| kubernetes.io/elb.health-check-flag | No        | String              | Whether to enable the ELB health check. Disabled by default.  
  - Enabling health check: Leave blank this parameter or set it to **on**.  
  - Disabling health check: Set this parameter to **off**. |
<p>| kubernetes.io/elb.health-check-option | No        | Table 8-9 Object    | ELB health check configuration items. |</p>
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Mandatory</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| kubernetes.io/session-affinity-mode | No | String | Listeners ensure session stickiness based on IP addresses. Requests from the same IP address will be forwarded to the same backend server.  
- Disabling sticky session: Do not set this parameter.  
- Enabling sticky session: Set this parameter to **SOURCE_IP**, indicating that the sticky session is based on the source IP address. |
| kubernetes.io/elb.session-affinity-option | No | **Table 8-8** Object | Sticky session timeout. |
| kubernetes.io/hws-hostNetwork | No | String | This parameter indicates whether the workload Services use the host network.  
The host network is not used by default. The value can be **true** or **false**. |
| externalTrafficPolicy | No | String | If sticky session is enabled, add this parameter so that requests are transferred to a fixed node. If a LoadBalancer Service with this parameter set to **Local** is created, a client can access the target backend only if the client is installed on the same node as the backend. |

### Table 8-11 Data structure of the elb.autocreate field

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Mandatory</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| name      | No        | String | Name of the load balancer that is automatically created.  
Value range: a string of 1 to 64 characters, including lowercase letters, digits, and underscores (_).  
The value must start with a lowercase letter and end with a lowercase letter or digit.  
Default name: **cce-lb+service.UID** |
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Mandatory</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| type               | No        | String     | Network type of the load balancer.  
|                    |           |            | - **public**: public network load balancer  
|                    |           |            | - **inner**: private network load balancer  
| Default value: **inner** |
| bandwidth_name     | No        | String     | Bandwidth name. The default value is **cce-bandwidth-********.  
|                    |           |            | Value range: a string of 1 to 64 characters, including lowercase letters, digits, and underscores (_).  
|                    |           |            | The value must start with a lowercase letter and end with a lowercase letter or digit. |
| bandwidth_charge_mode | No    | String     | Bandwidth billing mode.  
|                    |           |            | - **bandwidth**: billed by bandwidth  
|                    |           |            | - **traffic**: billed by traffic |
| bandwidth_size     | No        | Integer    | Bandwidth size. The default value is 1 to 2000 Mbit/s. Set this parameter based on the bandwidth range allowed in your region. |
| bandwidth_sharetype | No     | String     | Bandwidth sharing mode.  
|                    |           |            | - **PER**: dedicated bandwidth |
| eip_type           | No        | String     | EIP type.  
|                    |           |            | - **5_telcom**: China Telecom  
|                    |           |            | - **5_union**: China Unicom  
|                    |           |            | - **5_bgp**: dynamic BGP  
|                    |           |            | - **5_sbgp**: static BGP  
|                    |           |            | For dedicated load balancers, set this parameter to **5_gray**. |
| available_zone     | Yes       | Array of strings | AZ where the load balancer is located.  
|                    |           |            | This parameter is available only for dedicated load balancers. |
| l4_flavor_name     | Yes       | String     | Flavor name of the layer-4 load balancer.  
|                    |           |            | This parameter is available only for dedicated load balancers. |
### Parameter Table

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Mandatory</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>l7_flavor_name</td>
<td>Yes</td>
<td>String</td>
<td>Flavor name of the layer-7 load balancer. This parameter is available only for dedicated load balancers.</td>
</tr>
<tr>
<td>elb_virsubnet_ids</td>
<td>Yes</td>
<td>Array of strings</td>
<td>Subnet where the backend server of the load balancer is located. If this parameter is left blank, the default cluster subnet is used. Load balancers occupy different number of subnet IP addresses based on their specifications. Therefore, you are not advised to use the subnet CIDR blocks of other resources (such as clusters and nodes) as the load balancer CIDR block. This parameter is available only for dedicated load balancers.</td>
</tr>
</tbody>
</table>

### Step 3 Create a workload.

```bash
cubectl create -f nginx-deployment.yaml
```

If information similar to the following is displayed, the workload is being created.

```
deployment "nginx" created
```

```bash
cubectl get po
```

If information similar to the following is displayed, the workload is running.

```
NAME                     READY     STATUS             RESTARTS   AGE
etcd-0                   0/1       ImagePullBackOff  0          1h
icagent-m9dkt            0/0       Running            0          3d
nginx-2601814895-c1xhw   1/1       Running            0          6s
```

### Step 4 Create a Service.

```bash
cubectl create -f nginx-elb-svc.yaml
```

If information similar to the following is displayed, the Service has been created.

```
service "nginx" created
```

```bash
cubectl get svc
```

If information similar to the following is displayed, the access type has been set successfully, and the workload is accessible.

```
NAME         TYPE           CLUSTER-IP       EXTERNAL-IP   PORT(S)        AGE
etcd-svc     ClusterIP      None             <none>        3120/TCP       1h
kubernetes   ClusterIP      10.247.0.1       <none>        443/TCP        3d
nginx        LoadBalancer   10.247.130.196   10.78.42.242   80:31540/TCP   51s
```

### Step 5 Enter the URL in the address box of the browser, for example, 10.78.42.242:80. 10.78.42.242 indicates the IP address of the load balancer, and 80 indicates the access port displayed on the CCE console.
The Nginx is accessible.

**Figure 8-13** Accessing Nginx through the LoadBalancer Service

---

**Welcome to nginx!**

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org. Commercial support is available at nginx.com.

Thank you for using nginx.

---

**8.3.5 DNAT**

**Scenario**

A *destination network address translation (DNAT) gateway* is situated between cluster nodes and public networks and assigned an EIP. After receiving inbound requests from public networks, the NAT gateway translates the EIP (destination address in the inbound requests) into a cluster-internal address. It appears to workload users as if all nodes running the workload share the same EIP.

DNAT provides higher reliability than EIP-based NodePort in which the EIP is bound to a single node and once the node is down, all inbound requests to the workload will not be distributed. The access address is in the format of `<EIP>:\<access port>`, for example, 10.117.117.117:80.
Notes and Constraints

Observe the following constraints when using the NAT Gateway service:

- Clusters that use the VPC network model do not allow containers to access DNAT Services whose `externalTrafficPolicy` is set to `local`.
- Multiple rules for one NAT gateway can use the same EIP, but the rules for different NAT gateways must use different EIPs.
- Each VPC can have only one NAT gateway.
- Users cannot manually add the default route in a VPC.
- Only one SNAT rule can be added to a subnet in a VPC.
- SNAT and DNAT rules are designed for different functions. If SNAT and DNAT rules use the same EIP, resource preemption will occur. An SNAT rule cannot share an EIP with a DNAT rule with `Port Type` set to `All ports`.
- DNAT rules do not support binding an EIP to a virtual IP address.
- When both the EIP and NAT Gateway services are configured for a server, data will be forwarded through the EIP.
- The custom CIDR block must be a subset of the VPC subnet CIDR blocks.
- The custom CIDR block must be a CIDR block of Direct Connect and cannot conflicts with VPC's existing subnet CIDR blocks.
- When you perform operations on underlying resources of an ECS, for example, changing its specifications, the configured NAT gateway rules become invalid. You need to delete the rules and reconfigure them.
After a Service is created, if the **affinity setting** is switched from the cluster level to the node level, the connection tracing table will not be cleared. You are advised not to modify the Service affinity setting after the Service is created. If you need to modify it, create a Service again.

**Prerequisites**

You have created a NAT gateway and an elastic IP address. The specific procedure is as follows:

**Step 1** Log in to the management console, choose **Network > NAT Gateway** from the service list, and click **Buy NAT Gateway** in the upper right corner. Configure parameters based on site requirements.

![NOTE]

When buying a NAT gateway, ensure that the NAT gateway belongs to the same VPC and subnet as the CCE cluster where the workload is running.

**Step 2** Log in to the management console, choose **Network > Elastic IP** from the service list, and click **Buy EIP** in the upper right corner. Configure parameters based on site requirements.

------

**Adding a Service When Creating a Workload**

You can set the Service when creating a workload on the CCE console. A Nginx workload is used as an example. The procedure is as follows:

**Step 1** In the **Set Application Access** step of **Creating a Deployment, Creating a StatefulSet**, or **Creating a DaemonSet**, click **Add Service** and set the following parameters:

- **Access Type**: Select **LoadBalancer (DNAT)**.
- **Service Name**: Specify a Service name, which can be the same as the workload name.
- **Service Affinity**
  - Cluster level: The IP addresses and access ports of all nodes in a cluster can access the workload associated with the Service. Service access will cause performance loss due to route redirection, and the source IP address of the client cannot be obtained.
  - Node level: Only the IP address and access port of the node where the workload is located can access the workload associated with the Service. Service access will not cause performance loss due to route redirection, and the source IP address of the client can be obtained.
- **NAT Gateway**
  - NAT gateway: Select a NAT gateway from the drop-down list box. If no NAT gateway is available, click **Create a NAT gateway** to create one. The NAT gateway must be in the same VPC as the current cluster.
  - EIP: Select an EIP for the NAT gateway from the drop-down list box. If no EIPs are available, click **Create an EIP** to create one.
  - Select I have read and agree to the NAT Gateway Usage Restrictions.
● Port Settings
  - **Protocol**: protocol used by the Service.
  - **Container Port**: port that is defined in the container image and on which the workload listens. The Nginx workload listens on port 80.
  - **Access Port**: port mapped to the container port. The port number range is 1–65535.

**Step 2** After the configuration is complete, click **OK**.

**Step 3** On the workload creation page, click **Next: Configure Advanced Settings**. On the page displayed, click **Create**.

**Step 4** After the workload is successfully created, choose **Workloads > Deployments** or **Workloads > StatefulSets** on the CCE console. Click the name of the workload to show the details of the workload. On the workload details page, click the **Services** tab and obtain the access address, for example, 10.154.78.160:2.

**Step 5** Click the access address.

**Figure 8-15** Accessing Nginx through the NAT gateway

**Welcome to nginx!**

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to [nginx.org](http://nginx.org). Commercial support is available at [nginx.com](http://nginx.com).

Thank you for using nginx.

---- End

**Adding a Service After Creating a Workload**

You can set the Service after creating a workload. This has no impact on the workload status and takes effect immediately. The procedure is as follows:

**Step 1** Log in to the CCE console. In the navigation pane, choose **Workloads > Deployments**. On the workload list, click the name of the workload for which you will create a Service.

**Step 2** On the **Services** tab page, click **Create Service**.

The parameters are the same as those in **Adding a Service When Creating a Workload**.

**Step 3** Click **Create**. A DNAT gateway (DNAT) Service will be added for the workload.

---- End

**Updating a Service**

The DNAT gateway (DNAT) Service cannot be updated for the workload.
Setting the Access Type Using kubectl

You can set the Service when creating a workload using kubectl. This section uses a Nginx workload as an example to describe how to implement intra-cluster access using kubectl.

**Step 1** Use kubectl to connect to the cluster. For details, see Connecting to a Cluster Using kubectl.

**Step 2** Create and edit the `nginx-deployment.yaml` and `nginx-nat-svc.yaml` files.

The file names are user-defined. `nginx-deployment.yaml` and `nginx-nat-svc.yaml` are merely example file names.

```yaml
vi nginx-deployment.yaml
```

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx
spec:
  replicas: 1
  selector:
    matchLabels:
      app: nginx
  strategy:
    type: RollingUpdate
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - image: nginx
        imagePullPolicy: Always
        name: nginx
        imagePullSecrets:
        - name: default-secret
```

For descriptions of the preceding fields, see Table 6-4.

```yaml
vi nginx-nat-svc.yaml
```

```yaml
apiVersion: v1
kind: Service
metadata:
  labels:
    app: nginx
  name: nginx
annotations:
  kubernetes.io/elb.class: dnat
  kubernetes.io/natgateway.id: e4a1cfcf-29df-4ab8-a4ea-c05dc860f554
spec:
  loadBalancerIP: 10.78.42.242
  ports:
  - name: service0
    port: 80
    protocol: TCP
  selector:
    app: nginx
  type: LoadBalancer
```
Table 8-12 Key parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Mandatory</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>kubernetes.io/elb.class</td>
<td>Yes</td>
<td>String</td>
<td>This parameter is set to <code>dnat</code> so CCE can work with a NAT gateway and DNAT rules can be added.</td>
</tr>
<tr>
<td>kubernetes.io/natgateway.id</td>
<td>Yes</td>
<td>String</td>
<td>ID of a NAT gateway.</td>
</tr>
<tr>
<td>loadBalancerIP</td>
<td>Yes</td>
<td>String</td>
<td>EIP ID.</td>
</tr>
<tr>
<td>port</td>
<td>Yes</td>
<td>Integer</td>
<td>Access port set on the console. The value ranges from 1 to 65535.</td>
</tr>
<tr>
<td>targetPort</td>
<td>Yes</td>
<td>String</td>
<td>Container port set on the console. The value ranges from 1 to 65535.</td>
</tr>
<tr>
<td>type</td>
<td>Yes</td>
<td>String</td>
<td>NAT gateway service type must be set to <code>LoadBalancer</code>.</td>
</tr>
</tbody>
</table>

**Step 3** Create a workload.

```
kubectl create -f nginx-deployment.yaml
```

If information similar to the following is displayed, the workload is being created.

```
deployment "nginx" created
```

```
kubectl get po
```

If information similar to the following is displayed, the workload is running.

```
NAME                     READY     STATUS             RESTARTS   AGE
etcd-0                   0/1     ImagePullBackOff 0          59m
icagent-m9dkt            0/0     Running            0          3d
nginx-2601814895-sf71t   1/1     Running            0          8s
```

**Step 4** Create a Service.

```
kubectl create -f nginx-nat-svc.yaml
```

If information similar to the following is displayed, the Service has been created.

```
service "nginx-eip" created
```

```
kubectl get svc
```

If the following information is displayed, the Service has been set successfully, and the workload is accessible.

```
NAME         TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)        AGE
etcd-svc     ClusterIP   None             <none>        3120/TCP       59m
kubernetes    ClusterIP   10.247.0.1       <none>        443/TCP        3d
nginx-nat     LoadBalancer 10.247.226.2 10.154.74.98 80:30589/TCP  5s
```

**Step 5** In the address bar of your browser, enter `10.154.74.98:80` and press Enter.
In this example, **10.154.74.98** is the elastic IP address and **80** is the port number obtained in the previous step.

---

**8.3.6 Enabling Passthrough Networking for LoadBalancer Services**

**Painpoint**

A Kubernetes cluster can publish applications running on a group of pods as Services, which provide unified layer-4 access entries. For a Loadbalancer Service, kube-proxy configures the LoadbalanceIP in **status** of the Service to the local forwarding rule of the node by default. When a pod accesses the load balancer from within the cluster, the traffic is forwarded within the cluster instead of being forwarded by the load balancer.

kube-proxy is responsible for intra-cluster forwarding. kube-proxy has two forwarding modes: iptables and IPVS. iptables is a simple polling forwarding mode. IPVS has multiple forwarding modes but it requires modifying the startup parameters of kube-proxy. Compared with iptables and IPVS, load balancers provide more flexible forwarding policies as well as health check capabilities.

**Solution**

CCE supports passthrough. You can configure the **annotation** of **kubernetes.io/elb.pass-through** for the Loadbalancer Service. Intra-cluster access to the Service load balancer address is then forwarded to backend pods by the load balancer.

**Figure 8-16 Passthrough networking illustration**

- For CCE clusters:
When a LoadBalancer Service is accessed within the cluster, the access is forwarded to the backend pods using iptables/IPVS by default.

- Configure `elb.pass-through` for LoadBalancer Services. Intra-cluster access to the Service is forwarded to the ELB load balancer first. The load balancer forwards the traffic back to the node.

- For CCE Turbo clusters:
  - When an LB Service is accessed within the cluster, the access is forwarded to the backend pods using iptables/IPVS by default.
  - Configure `elb.pass-through` for LB Services. Intra-cluster access to the Service is forwarded to the ELB first. The ELB forwards the traffic directly to the container.

Notes and Constraints

- In a CCE Turbo cluster, when a node accesses the load balancer address of the LoadBalancer Service published by a container on the same node, passthrough is not supported.
- Passthrough is not supported for clusters of v1.15 or earlier.

Procedure

This section describes how to create a Deployment using an Nginx image and create a Service with passthrough networking enabled.

**Step 1** Use the Nginx image to create a Deployment.

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx
spec:
  replicas: 2
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - image: nginx:latest
        name: container-0
        resources:
          limits:
            cpu: 100m
            memory: 200Mi
          requests:
            cpu: 100m
            memory: 200Mi
        imagePullSecrets:
        - name: default-secret
```

**Step 2** Create a LoadBalancer Service and configure `kubernetes.io/elb.pass-through` to `true`.

For details about how to create a LoadBalancer Service, see [Automatically Creating a LoadBalancer Service](#).

```yaml
apiVersion: v1
kind: Service
```
A shared load balancer named **james** is automatically created. Use `kubernetes.io/elb.subnet-id` to specify the VPC subnet where the load balancer is located. The load balancer and the cluster must be in the same VPC.

----End

**Verification**

Check the ELB load balancer corresponding to the created Service. The load balancer name is **james**. The number of ELB connections is **0**, as shown in the following figure.

Use `kubectl` to connect to the cluster, go to an Nginx container, and access the ELB address. The access is successful.

```bash
# kubectl get pod
NAME                     READY   STATUS    RESTARTS   AGE
nginx-7c4c5cc6b5-vpncx   1/1     Running   0          9m47s
nginx-7c4c5cc6b5-xj5wl   1/1     Running   0          9m47s

# kubectl exec -it nginx-7c4c5cc6b5-vpncx -- /bin/sh
# curl 120.46.141.192
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <title>Cloud Container Engine</title>
  <link rel="stylesheet" href="css/style.css">
  <script src="js/script.js"></script>
</head>
<body>
  <h1>Verify ELB Load Balancer</h1>
  <p>The ELB load balancer name is <strong>james</strong>. The number of ELB connections is <strong>0</strong>.</p>
  <p>Use <code>kubectl</code> to connect to the cluster, go to an Nginx container, and access the ELB address. The access is successful.</p>
</body>
</html>
```
8.4 Ingresses

8.4.1 Overview

Layer-7 load balancing (ingress) uses shared and dedicated load balancers and additionally supports Uniform Resource Identifier (URI) configurations and distributes access traffic to services based on URIs, compared with layer-4 load balancing. In addition, different functions are implemented based on various URIs.

The access address is in the format of `<IP address of the load balancer>:<access port>`/defined URI, for example, `10.117.117.117:80/helloworld`.

You can configure load balancers of public and private networks to implement layer-7 route forwarding on public networks and private networks (intra-VPC networks).
8.4.2 Basic Operations

Precautions

Automatically created load balancers should not be used by other resources. Otherwise, these load balancers cannot be completely deleted.

Creating an Ingress

You can set the Service when creating a workload on the CCE console. In the following example, a Nginx workload is created with an ingress configured.

**Step 1** Create a workload. For details, see Creating a Deployment, Creating a StatefulSet, or Creating a DaemonSet.
- If a NodePort Service has been created, go to **Step 3**.
- If no NodePort Service is created, go to **Step 2**.

**Step 2** (Optional) Set the NodePort Service.

1. Log in to the CCE console. In the navigation pane, choose Resource Management > Network.
2. On the Services tab page, click Create Service. In the Select Type dialog box, select NodePort.
   - **Service Name**: Specify a Service name, which can be the same as the workload name.
   - **Cluster Name**: Select the cluster for which you want to add a Service.
   - **Namespace**: Select a namespace for which you want to add a Service.
   - **Workload**: Click Select Workload, select the name of the workload for which the NodePort Service is to be configured, and click OK.
   - **Service Affinity**:
     - Cluster level: The IP addresses and access ports of all nodes in a cluster can access the workload associated with the Service.
access will cause performance loss due to route redirection, and the source IP address of the client cannot be obtained.

- Node level: Only the IP address and access port of the node where the workload is located can access the workload associated with the Service. Service access will not cause performance loss due to route redirection, and the source IP address of the client can be obtained.

  - IPv6: This function is disabled by default. After this function is enabled, the cluster IP address of the Service changes to an IPv6 address. For details, see [How Do I Create an IPv4/IPv6 Dual-Stack Cluster](#). This parameter is available only in clusters of v1.15 or later with IPv6 enabled (set during cluster creation).

- Port Settings
  - Protocol: protocol used by the Service.
  - Container Port: port on which the workload in the container image listens. The Nginx application listens on port 80.
  - Access Port: Specify a port to which the container port will be mapped when the node's private IP address is used for accessing the workload. The port number range is 30000–32767. You are advised to select **Automatically generated**.
    - **Auto generated**: The system automatically assigns a port number.
    - **Manually specified**: Specify a fixed node port. The port number ranges from 30000 to 32767. Ensure that the port is unique in a cluster.

3. Click Create. The NodePort Service is successfully set.

Step 3 Add an ingress.

1. In the navigation pane, choose **Resource Management > Network**.
2. On the **Ingresses** tab page, click **Create Ingress**.

**Figure 8-18 Creating an ingress**

- **Access Type**: Select a load balancer.
- **Ingress Name**: Specify a name of an ingress, for example, **ingress-demo**.
- **Cluster Name**: Select the cluster to which the ingress is to be added.
- **Namespace**: Select the namespace to which the ingress is to be added.
- **nginx-ingress**: This option is displayed only when **nginx-ingress** has been installed in the cluster.
After you switch on nginx-ingress, it is interconnected to provide Layer-7 access. You can configure the following parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Front-End Protocol</td>
<td>HTTP and HTTPS are supported.</td>
</tr>
<tr>
<td>External Port</td>
<td>Listening port reserved for installing the nginx-ingress add-on. The port number is 80 for HTTP and 443 for HTTPS.</td>
</tr>
<tr>
<td>Timeout</td>
<td>Timeout period that the client establishes a connection with the proxy server.</td>
</tr>
<tr>
<td>Redirected To</td>
<td>Address to which all workload content is to be redirected, for example, <a href="https://www.huaweicloud.com/">https://www.huaweicloud.com/</a>.</td>
</tr>
<tr>
<td>Custom Settings</td>
<td>Used by ingresses to modify the configuration in the nginx.conf file. For details about keys and values to be set, see Annotations.</td>
</tr>
</tbody>
</table>

- **ELB Configuration**: Ingress uses the load balancer of the ELB service to provide layer-7 network access.

**NOTICE**

1. After the ingress is created, ensure that you do not modify its load balancer settings on the ELB console because the ingress may become abnormal. Modify the settings only on the CCE console.

2. Ensure that automatically created load balancers are not used by another ingress or Service. If they are used by another ingress or Service, load balancers may still be available in ELB after they are deleted on the CCE console.

After the nginx-ingress add-on is connected, the load balancing settings configured in the add-on are automatically used, and these settings will not be displayed on the GUI.

**Elastic Load Balancer**: The selected or created load balancer must be in the same VPC as the current cluster, and it must match the load balancer type (private or public network).

- **Shared**: Resources are deployed in cluster mode and shared among load balancers.

- **Dedicated**: Resources are exclusively used, ensuring high performance.

You can create public network or private network load balancers.
### Public network
You can select an existing public network load balancer or have the system automatically create a new public network load balancer. If no shared load balancer is available, click **Create Load Balancer** to create one and then click the refresh button.

- **Enterprise Project**: Select an enterprise project in which the load balancer is created.
- **Change Configuration**: When selecting **Public network > Automatically created**, you can click **Change Configuration** to modify the name, specifications, billing mode, and bandwidth of the ELB instance to be created.

### Private network
You can select an existing private network load balancer or have the system automatically create a new private network load balancer.

- **Enterprise Project**: Select an enterprise project in which the load balancer is created.

#### Listener Configuration
Ingress configures a listener for the load balancer, which listens to requests from the load balancer and distributes traffic.

**The configuration items are not displayed if the nginx-ingress add-on is connected.**

- **External Protocol**: HTTP and HTTPS are supported.
- **External Port**: Port number that is open to the ELB service address. The port number can be specified randomly.
  
  If the nginx-ingress add-on is connected, **External Port** is not displayed and ports 80 and 443 are enabled by default.

- **Server Certificate**: If you select HTTPS, choose a secret certificate.

  **NOTE**

  - The secret certificate **ingress-test-secret.yaml** is required only when HTTPS is selected. For details on how to create a secret, see [Creating a Secret](#).
  - If there is already an HTTPS ingress for the chosen port on the load balancer, the certificate of the new HTTPS ingress must be the same as the certificate of the existing ingress. This means that a listener has only one certificate. If two certificates, each with a different ingress, are added to the same listener of the same load balancer, only the earliest certificate takes effect on the load balancer.

- **SNI**: If multiple domain names are required for external access and different domain names use independent certificates, you need to enable the Server Name Indication (SNI) function. After SNI is enabled, the client is allowed to submit the requested domain name when initiating an SSL handshake request. After receiving the SSL request, the load balancer searches for the certificate based on the domain name in the request. If the certificate corresponding to the domain name is found, the load balancer returns the certificate. Otherwise, the default certificate is returned.
NOTE
○ The SNI option is available only when HTTPS is selected.
○ The SNI option can be configured only for clusters of v1.15.11 or later.

Security Policy: combinations of different TLS versions and supported cipher suites available to HTTPS listeners.
For details about security policies, see Security Policy.

NOTE
○ Security Policy is available only when HTTPS is selected.
○ Security policies can be configured only for clusters of v1.17.9 and later.

Forwarding Policies: When the access address of a request matches the forwarding policy (a forwarding policy consists of a domain name and URL), the request is forwarded to the corresponding target Service for processing.

NOTICE
You can modify the load balancer settings, including algorithm, sticky session, and health check configurations, after you select a Service in Forwarding Policies. Do not modify these configurations on the ELB console.

Domain Name: actual domain name. Ensure that the entered domain name has been registered and archived. After the ingress is created, bind the domain name to the IP address of the automatically created load balancer (IP address of the ingress access address). If a domain name rule is configured, the domain name must always be used for access.

Rule Matching
○ Prefix matching: If the URL is set to /healthz, the URL that meets the prefix can be accessed. For example, /healthz/v1 and /healthz/v2.
○ Exact matching: Only the URL that is the same as the specified URL can be accessed. For example, if the URL is set to /healthz, only /healthz can be accessed.
○ Regular expression matching: The URL rule can be set, for example, /[A-Za-z0-9_.-]+/test. All URLs that comply with this rule can be accessed, for example, /abcA9/test and /v1-Ab/test. Two regular expression standards are supported: POSIX and Perl.

URL: access path to be registered, for example, /healthz.

Target Service: Select an existing Service or create a Service. Only NodePort Services can be selected from the list on the page. The query result has been automatically filtered. If the nginx-ingress add-on has been connected, you can click Create YAML in the upper right corner of the page to connect to a ClusterIP Service.
- **Service Access Port**: Select the access port of the target Service.

- **ELB Settings**: If multiple routes use the same Service, they are using the same Service load balancing configuration.
  - **Algorithm Type**: Three algorithms are available: weighted round robin, weighted least connections algorithm, or source IP hash.
    - **Weighted round robin**: Requests are forwarded to different servers based on their weights, which indicate server processing performance. Backend servers with higher weights receive proportionately more requests, whereas equal-weighted servers receive the same number of requests. This algorithm is often used for short connections, such as HTTP services.
    - **Weighted least connections**: In addition to the weight assigned to each server, the number of connections processed by each backend server is also considered. Requests are forwarded to the server with the lowest connections-to-weight ratio. Building on least connections, the weighted least connections algorithm assigns a weight to each server based on their processing capability. This algorithm is often used for persistent connections, such as database connections.
    - **Source IP hash**: The source IP address of each request is calculated using the hash algorithm to obtain a unique hash key, and all backend servers are numbered. The generated key allocates the client to a particular server. This enables requests from different clients to be distributed in load balancing mode and ensures that requests from the same client are forwarded to the same server. This algorithm applies to TCP connections without cookies.
  - **Sticky Session**: This function is disabled by default. After this function is enabled, you need to select a sticky session type and set the sticky session duration.
    - **Load balancer cookie**: The load balancer generates a cookie after receiving a request from the client. All subsequent requests with the cookie are routed to the same backend server.
    - **Application cookie**: The application deployed on the backend server generates a cookie after receiving the first request from the client. All subsequent requests that contain the cookie are routed to this backend server. This sticky session mode is supported for shared load balancers.
  - **Health Check**: This function is enabled by default. To enable this function, set parameters as prompted. For details about the parameters, see Configuring a Health Check.

- **Operation**: Click **Delete** to delete the configuration. Click **Add Forwarding Policies** to add multiple forwarding policies.

**Step 4** After the configuration is complete, click **Submit**.

After the ingress is created, it is displayed in the ingress list.

**Step 5** Access the `/healthz` interface of the workload, for example, workload `defaultbackend`. 
Method 1: By using a load balancer's IP address (The domain name cannot be configured for LoadBalanced Services.)

1. Obtain the access address of the /healthz interface of defaultbackend. The access address is in the format of <Load balancer's IP address>:<External port><Mapping URL>. For example, 10.154.73.151:80/healthz.

![Figure 8-19 Obtaining the access address](image)

2. Enter the URL of the /healthz interface, for example, http://10.154.73.151:80/healthz, in the address box of the browser to access the workload.

![Figure 8-20 Accessing the /healthz interface of defaultbackend](image)

Method 2: By using a domain name

The following uses the domain name ingress.com configured in the ingress as an example.

1. Obtain the domain name and access address (IP address and port number) of the ingress-demo.

![Figure 8-21 Obtaining the domain name and access address](image)

2. Configure the IP address in the access address and the domain name in the C:\Windows\System32\drivers\etc\hosts file on the local host, as shown in the following figure.
3. Enter \texttt{http://\textless Domain name\textgreater :\textless Access port\textgreater /\textless Mapping URL\textgreater} in the address box of the browser. For example, \texttt{http://ingress.com:81/james}.

\textit{----End}

\section*{Updating an Ingress}

After adding an ingress, you can update its port, domain name, and route configuration. The procedure is as follows:

\begin{quote}
\textbf{NOTE}
\end{quote}

- After the ingress is created, ensure that you do not modify its load balancer settings on the ELB console because the ingress may become abnormal. Modify the settings only on the CCE console.
- You can modify the load balancer settings, including algorithm, sticky session, and health check configurations, after you select a Service in \textit{Forwarding Policies} on the CCE console. Do not modify these configurations on the ELB console.
- Ensure that automatically created load balancers are not used by another ingress or Service. If they are used by another ingress or Service, load balancers may still be available in ELB after they are deleted on the CCE console.

\textbf{Step 1} Log in to the CCE console. In the navigation pane, choose \textit{Resource Management} > \textit{Network}. On the \textit{Ingresses} tab page, filter ingresses by cluster and namespace, and click \textit{Update} for the ingress to be updated.

\textbf{Step 2} On the \textit{Update Ingress} page, set the following parameters as follows:

The parameters are the same as those in \textit{Creating an Ingress}. 

Step 3  Click Submit. The ingress will be updated for the workload.

---End

Related Operations

The Kubernetes ingress structure does not contain the property field. Therefore, the ingress created by the API called by client-go does not contain the property field. CCE provides a solution to ensure compatibility with the Kubernetes client-go. For details about the solution, see How Can I Achieve Compatibility Between Ingress's property and Kubernetes client-go?

8.4.3 Using kubectl to Create an Ingress

Scenario

This section uses an Nginx workload as an example to describe how to create an ingress using kubectl.

Creating an Ingress - Interconnecting with an Existing Load Balancer

Step 1  Use kubectl to connect to the cluster. For details, see Connecting to a Cluster Using kubectl.

Step 2  Create the ingress-test-deployment.yaml, ingress-test-svc.yaml, ingress-test-ingress.yaml, and ingress-test-secret.yaml files.

The file names are user-defined. ingress-test-deployment.yaml, ingress-test-svc.yaml, ingress-test-ingress.yaml, and ingress-test-secret.yaml are merely example file names.

Note

- The secret certificate ingress-test-secret.yaml is required only when HTTPS is selected. For details on how to create a secret, see Creating a Secret.
- If there is already an HTTPS ingress for the chosen port on the load balancer, the certificate of the new HTTPS ingress must be the same as the certificate of the existing ingress.

vi ingress-test-deployment.yaml

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: ingress-test-deployment
spec:
  replicas: 1
  selector:
    matchLabels:
      app: ingress-test-deployment
  strategy:
    type: RollingUpdate
  template:
    metadata:
      labels:
        app: ingress-test-deployment
    spec:
      containers:
        # Third-party public image. You can obtain the address by referring to the description or use your own image.
```
vi ingress-test-svc.yaml

```yaml
apiVersion: v1
kind: Service
metadata:
  labels:
    app: ingress-test-svc
    name: ingress-test-svc
spec:
  ports:
    - name: service0
      port: 8888
      protocol: TCP
      targetPort: 8888
    - name: service1
      port: 8081
      protocol: TCP
      targetPort: 8081
  selector:
    app: ingress-test-deployment
    type: NodePort
```

Table 8-14 Key parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Mandatory</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>port</td>
<td>Yes</td>
<td>Integer</td>
<td>Access port mapped to the cluster-internal IP address. The value ranges from 1 to 65535.</td>
</tr>
<tr>
<td>protocol</td>
<td>Yes</td>
<td>String</td>
<td>IP protocol used by the port. The value can be TCP or UDP.</td>
</tr>
<tr>
<td>targetPort</td>
<td>Yes</td>
<td>String</td>
<td>Container port on which the application listens. The value ranges from 1 to 65535.</td>
</tr>
<tr>
<td>type</td>
<td>Yes</td>
<td>String</td>
<td>Uses the NodePort Service to connect to the load balancer. NodePort indicates the private IP address of a node.</td>
</tr>
</tbody>
</table>

vi ingress-test-ingress.yaml

- For clusters of v1.15 and later, the value of apiVersion is networking.k8s.io/v1beta1.
- For clusters of v1.13 or earlier, the value of apiVersion is extensions/v1beta1.

```yaml
apiVersion: networking.k8s.io/v1beta1
kind: Ingress
metadata:
  annotations:
    kubernetes.io/elb.class: union
    kubernetes.io/elb.id: f7891f9a-49f2-4ee2-b1ae-f019cd84eb4f
    kubernetes.io/elb.ip: 192.168.0.39
    kubernetes.io/elb.subnet-id: 29a0567e-96f1-4227-91cc-64f54d0b064d
    kubernetes.io/elb.port: "80"
```
```yaml
kubernetes.io/ingress.class: cce
kubernetes.io/elb.tls-ciphers-policy: tls-1-2
name: ingress-test-ingress
spec:
tls:
  - secretName: ingress-test-secret
rules:
  - http:
    paths:
      - backend:
        serviceName: ingress-test-svc
        servicePort: 8888
    property:
      ingress.beta.kubernetes.io/url-match-mode: EQUAL_TO
      path: "/healthz"
    host: ingress.com
```

**NOTE**

Security policy (kubernetes.io/elb.tls-ciphers-policy) is supported only in clusters of v1.17.11 and later.

### Table 8-15 Key parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Mandatory</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>kubernetes.io/elb.class</td>
<td>Yes</td>
<td>String</td>
<td>Select a proper load balancer type. The value can be:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• union: shared load balancer</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• performance: dedicated load balancer, which can be used only in clusters</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>of v1.17 and later. For details, see Differences Between Shared and Dedicated Load Balancers.</td>
</tr>
<tr>
<td>kubernetes.io/elb.id</td>
<td>Yes</td>
<td>String</td>
<td>This parameter indicates the ID of a load balancer. The value can contain</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1 to 100 characters.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Obtaining the value:</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>On the management console, click Service List, and choose Network &gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Elastic Load Balance. Click the name of the target load balancer. On the</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Summary tab page, find and copy the ID.</td>
</tr>
<tr>
<td>kubernetes.io/elb.ip</td>
<td>Yes</td>
<td>String</td>
<td>This parameter indicates the service address of a load balancer. The value</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>can be the public IP address of a public network load balancer or the</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>private IP address of a private network load balancer.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Mandatory</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------</td>
<td>-----------</td>
<td>------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| kubernetes.io/elb.subnet-id | -         | String | This parameter indicates the ID of the subnet where the cluster is located. The value can contain 1 to 100 characters.  
  - Mandatory when a cluster of v1.11.7-r0 or earlier is to be automatically created.  
  - Optional for clusters later than v1.11.7-r0.  
  For details on how to obtain the subnet ID, see What Is the Difference Between the VPC Subnet API and the OpenStack Neutron Subnet API? |
| kubernetes.io/elb.eip-id   | Yes       | String | This parameter indicates the EIP ID. The value can contain 1 to 100 characters.  
  **Obtaining the value:**  
  Choose Service List > Network > Elastic IP. Click the EIP name. On the EIP details page, copy the ID in Basic Information. |
| kubernetes.io/elb.enterpriseID | No       | String | Clusters of v1.15 and later versions support this field. In clusters earlier than v1.15, load balancers are created in the default project by default.  
  This parameter indicates the ID of the ELB enterprise project in which the load balancer will be created.  
  The value contains 1 to 100 characters.  
  **Obtaining the value:**  
  Log in to the management console and choose Enterprise > Project Management on the top menu bar. In the list displayed, click the name of the target enterprise project, and copy the ID on the enterprise project details page. |
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Mandatory</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| kubernetes.io/elb.tls-ciphers-policy | No        | String        | The default value is **tls-1-2**, which is the security policy used by the listener and takes effect only when the HTTPS protocol is used. Options:  
  ● tls-1-0  
  ● tls-1-1  
  ● tls-1-2  
  ● tls-1-2-strict  
For details of cipher suites for each security policy, see **Table 8-16**. |
| kubernetes.io/elb.port        | Yes       | Integer       | This parameter indicates the external port registered with the address of the LoadBalancer Service. The value ranges from 1 to 65535.         |
| kubernetes.io/ingress.class   | Yes       | String        | This parameter indicates whether to enable the nginx-ingress add-on.  
  ● **cce**: The nginx-ingress add-on is disabled. The shared load balancer is used by default.  
  ● **nginx**: The nginx-ingress add-on is enabled.  
This parameter is mandatory when an ingress is created by calling the API. |
| tls                           | No        | Array of strings | This parameter is mandatory if HTTPS is used.                                                                                              |
| secretName                    | No        | String        | This parameter is mandatory if HTTPS is used. Set this parameter to the name of the created secret.                                         |
| serviceName                   | Yes       | String        | This parameter indicates the name of the ingress-test-svc.yaml Service.                                                                    |
| servicePort                   | Yes       | Integer       | This parameter indicates the port number of the ingress-test-svc.yaml Service.                                                            |
Cloud Container Engine
User Guide

8 Networking

Parameter

Mandat
ory

Type

Description

ingress.beta.kuber
netes.io/urlmatch-mode

No

String

This parameter indicates the route
matching policy.
Default: STARTS_WITH (prefix
match)
Options:
● EQUAL_TO: exact match
● STARTS_WITH: prefix match
● REGEX: regular expression match

path

Yes

String

This parameter indicates the userdefined route path.

host

No

String

This parameter indicates the domain
name configuration. By default, this
parameter is left blank, and the
domain name needs to be fully
matched.

Table 8-16 tls_ciphers_policy parameter description
Security
Policy

TLS Version

Cipher Suite

tls-1-0

TLS 1.2


TLS 1.1
TLS 1.0
tls-1-1

TLS 1.2
TLS 1.1

tls-1-2

TLS 1.2

tls-1-2strict

TLS 1.2

Issue 01 (2021-09-08)


Copyright © Huawei Technologies Co., Ltd.

349


vi ingress-test-secret.yaml
apiVersion: v1
data:
tls.crt: LS0tL******tLS0tG==
tls.key: LS0tL******0tLS0K
kind: Secret
metadata:
  annotations:
  - description: test for ingressTLS secrets
  - name: ingress-test-secret
  - namespace: default
  type: IngressTLS

NOTE
In the preceding information, tls.crt and tls.key are only examples. Replace them with the actual files. The values of tls.crt and tls.key are the content encrypted using Base64.

Step 3 Create a workload.

kubectl create -f ingress-test-deployment.yaml

If information similar to the following is displayed, the workload is being created:
deployment/ingress-test-deployment created

kubectl get po

If information similar to the following is displayed, the workload has been created successfully:

<table>
<thead>
<tr>
<th>NAME</th>
<th>READY</th>
<th>STATUS</th>
<th>RESTARTS</th>
<th>AGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>ingress-test-1627801589-r64pk</td>
<td>1/1</td>
<td>Running</td>
<td>0</td>
<td>6s</td>
</tr>
</tbody>
</table>

Step 4 Create a secret.

kubectl create -f ingress-test-secret.yaml

If information similar to the following is displayed, the secret is being created:

secret/ingress-test-secret created

kubectl get secrets

If information similar to the following is displayed, the secret has been created successfully:

<table>
<thead>
<tr>
<th>NAME</th>
<th>TYPE</th>
<th>DATA</th>
<th>AGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>dash-dashboard</td>
<td>Opaque</td>
<td>0</td>
<td>7d</td>
</tr>
<tr>
<td>dash-dashboard-token-f2nbk</td>
<td>kubernetes.io/service-account-token</td>
<td>3</td>
<td>7d</td>
</tr>
<tr>
<td>default-secret</td>
<td>kubernetes.io/dockerconfigjson</td>
<td>1</td>
<td>8d</td>
</tr>
<tr>
<td>default-token-wfn4l</td>
<td>kubernetes.io/service-account-token</td>
<td>3</td>
<td>8d</td>
</tr>
<tr>
<td>paas.elb</td>
<td>cfe/secure-opaque</td>
<td>2</td>
<td>8d</td>
</tr>
<tr>
<td>ingress-test-secret</td>
<td>IngressTLS</td>
<td>2</td>
<td>13s</td>
</tr>
</tbody>
</table>

Step 5 Create a Service.

kubectl create -f ingress-test-svc.yaml

If information similar to the following is displayed, the Service is being created:
service/ingress-test-svc created

kubectl get svc

If information similar to the following is displayed, the Service has been created successfully:
### Creating an Ingress - Automatically Creating a Load Balancer

**Step 1** Use `kubectl` to connect to the cluster. For details, see [Connecting to a Cluster Using `kubectl`](#).

**Step 2** Create the `ingress-test-deployment.yaml`, `ingress-test-svc.yaml`, `ingress-test-ingress.yaml`, and `ingress-test-secret.yaml` files.

The file names are user-defined. `ingress-test-deployment.yaml`, `ingress-test-svc.yaml`, `ingress-test-ingress.yaml`, and `ingress-test-secret.yaml` are merely example file names.

**NOTE**
- The secret certificate `ingress-test-secret.yaml` is required only when HTTPS is selected. For details on how to create a secret, see [Creating a Secret](#).
- If there is already an HTTPS ingress for the chosen port on the load balancer, the certificate of the new HTTPS ingress must be the same as the certificate of the existing ingress.

```bash
vi ingress-test-deployment.yaml
```

---

### End

---

## Creating an Ingress - Automatically Creating a Load Balancer

**Step 1** Use `kubectl` to connect to the cluster. For details, see [Connecting to a Cluster Using `kubectl`](#).

**Step 2** Create the `ingress-test-deployment.yaml`, `ingress-test-svc.yaml`, `ingress-test-ingress.yaml`, and `ingress-test-secret.yaml` files.

The file names are user-defined. `ingress-test-deployment.yaml`, `ingress-test-svc.yaml`, `ingress-test-ingress.yaml`, and `ingress-test-secret.yaml` are merely example file names.

**NOTE**
- The secret certificate `ingress-test-secret.yaml` is required only when HTTPS is selected. For details on how to create a secret, see [Creating a Secret](#).
- If there is already an HTTPS ingress for the chosen port on the load balancer, the certificate of the new HTTPS ingress must be the same as the certificate of the existing ingress.

```bash
vi ingress-test-deployment.yaml
```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: ingress-test-deployment
spec:
  replicas: 1
  selector:
    matchLabels:
      app: ingress-test-deployment
strategy:
  type: RollingUpdate
template:
  metadata:
    labels:
      app: ingress-test-deployment
  spec:
    containers:
      # Third-party public image. You can obtain the address by referring to the description or use your own image.
      - image: nginx
        imagePullPolicy: Always
        name: nginx
        imagePullSecrets:
        - name: default-secret

vi ingress-test-svc.yaml

apiVersion: v1
kind: Service
metadata:
  name: ingress-test-svc
spec:
  ports:
    - name: service0
      port: 8888
      protocol: TCP
      targetPort: 8888
    - name: service1
      port: 8081
      protocol: TCP
      targetPort: 8081
  selector:
    app: ingress-test-deployment
  type: NodePort

```
Table 8-17 Key parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Mandatory</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>port</td>
<td>Yes</td>
<td>Integer</td>
<td>Access port mapped to the cluster-internal IP address. The value ranges from 1 to 65535.</td>
</tr>
<tr>
<td>protocol</td>
<td>Yes</td>
<td>String</td>
<td>IP protocol used by the port. The value can be TCP or UDP.</td>
</tr>
<tr>
<td>targetPort</td>
<td>Yes</td>
<td>String</td>
<td>Container port on which the application listens. The value ranges from 1 to 65535.</td>
</tr>
<tr>
<td>type</td>
<td>Yes</td>
<td>String</td>
<td>Uses the NodePort Service to connect to the load balancer. NodePort indicates the private IP address of a node.</td>
</tr>
</tbody>
</table>
```
vi ingress-test-ingress.yaml

- For clusters of v1.15 and later, the value of apiVersion is networking.k8s.io/v1beta1.
- For clusters of v1.13 or earlier, the value of apiVersion is apps/v1.

Example of using a shared public network load balancer:

```yaml
apiVersion: networking.k8s.io/v1beta1
kind: Ingress
metadata:
  annotations:
    kubernetes.io/elb.subnet-id: 29a0567e-96f1-4227-91cc-64f54d0b064d
    kubernetes.io/elb.enterpriseID: f6b9fffc-a9b7-4a50-a25e-364f587abe44
    kubernetes.io/elb.autocreate: '{"type":"public","bandwidth_name":"cce-bandwidth-1551163379627","bandwidth_chargemode":"bandwidth","bandwidth_size":5,"bandwidth_sharetype":"PER","eip_type":5_bgp,"name":"James"}
    kubernetes.io/elb.ciphers-policy: tls-1-2
    kubernetes.io/elb.port: 80
    kubernetes.io/ingress.class: cce
    kubernetes.io/elb.class: union
name: ingress-test-ingress
spec:
  tls:
    - secretName: ingress-test-secret
  rules:
    - http:
        paths:
          - serviceName: ingress-test-svc
            servicePort: 8888
          property:
            ingress.beta.kubernetes.io/url-match-mode: EQUAL_TO
            path: /healthz
        host: ingress.com
```

Example public network dedicated load balancer (for clusters of v1.17 and later only):

```yaml
apiVersion: networking.k8s.io/v1beta1
kind: Ingress
metadata:
  labels:
    zone: data
    isExternal: 'true'
  name: test2
  namespace: default
annotations:
  kubernetes.io/ingress.class: cce
  kubernetes.io/elb.port: '80'
  kubernetes.io/elb.subnet-id: '
  kubernetes.io/elb.autocreate:
    { "type": "public",
      "bandwidth_name": "cce-bandwidth-1626751177680",
      "bandwidth_chargemode": "bandwidth",
      "bandwidth_size": 1,
      "bandwidth_sharetype": "PER",
      "eip_type": "5_gray",
      "available_zone": [ "cn-south-2b"
      ],
      "l7_flavor_name": "L7_flavor.elb.s1.small",
      "elb_virsubnet_ids": [ "14567f27-8ae4-42b8-ae47-9f847a4690dd"
```


```yaml

```}

kubernetes.io/elb.enterpriseID: '0'
kubernetes.io/elb.class: performance

```
spec:
  rules:
  - host: ''
    http:
      paths:
      - backend:
        serviceName: test
        servicePort: 80
        path: "/"
    property:
      ingress.beta.kubernetes.io/url-match-mode: STARTS_WITH

Table 8-18 Key parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Mandatory</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>kubernetes.io/elb.class</td>
<td>Yes</td>
<td>String</td>
<td>Select a proper load balancer type.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>The value can be:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• union: shared load balancer</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• performance: dedicated load balancer, which can be used only in clusters of v1.17 and later. For details, see Differences Between Shared and Dedicated Load Balancers.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Default value: union</td>
</tr>
<tr>
<td>kubernetes.io/ingress.class</td>
<td>Yes</td>
<td>String</td>
<td>This parameter indicates whether to enable the nginx-ingress add-on.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• cce: The nginx-ingress add-on is disabled. The shared load balancer is used by default.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• nginx: The nginx-ingress add-on is enabled.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>This parameter is mandatory when an ingress is created by calling the API.</td>
</tr>
<tr>
<td>kubernetes.io/elb.port</td>
<td>Yes</td>
<td>Integer</td>
<td>This parameter indicates the external port registered with the address of the LoadBalancer Service.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>The value ranges from 1 to 65535.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Mandatory</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------</td>
<td>-----------</td>
<td>-------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>kubernetes.io/elb.subnet-id</td>
<td>-</td>
<td>String</td>
<td>This parameter indicates the ID of the subnet where the cluster is located. The value can contain 1 to 100 characters.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>● Mandatory when a cluster of v1.11.7-r0 or earlier is to be automatically created.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>● Optional for clusters later than v1.11.7-r0.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>For details about how to obtain the value, see What Is the Difference Between the VPC Subnet API and the OpenStack Neutron Subnet API.</td>
</tr>
<tr>
<td>kubernetes.io/elb.enterpriseID</td>
<td>Yes</td>
<td>String</td>
<td>Clusters of v1.15 and later versions support this field. In clusters earlier than v1.15, load balancers are created in the default project by default.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>This parameter indicates the ID of the ELB enterprise project in which the load balancer will be created.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>The value contains 1 to 100 characters.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>How to obtain:</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Log in to the management console and choose Enterprise &gt; Project Management on the top menu bar. Click the name of the target enterprise project, and copy the ID on the enterprise project details page.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Mandatory</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>-----------</td>
<td>---------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>kubernetes.io/elb.autocreate</td>
<td>Yes</td>
<td>elb.autocreate object</td>
<td>Whether to automatically create a load balancer associated with an ingress.</td>
</tr>
</tbody>
</table>

**Example:**
- If a public network load balancer will be automatically created, set this parameter to the following value:
  
  ```json
  {
  "type": "public",
  "bandwidth_name": "cce-bandwidth-1551163379627",
  "bandwidth_chargemode": "bandwidth",
  "bandwidth_size": 5,
  "bandwidth_sharetype": "PER",
  "eip_type": "5_bgp",
  "name": "james"
  }
  ``
- If a private network load balancer will be automatically created, set this parameter to the following value:
  
  ```json
  {
  "type": "inner",
  "name": "A-location-d-test"
  }
  ``

**Table 8-19 Data structure of the elb.autocreate field**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Mandatory</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>No</td>
<td>String</td>
<td>Name of the automatically created load balancer. Value range: a string of 1 to 64 characters, including lowercase letters, digits, and underscores (_). The value must start with a lowercase letter and end with a lowercase letter or digit. Default value: cce-lb+ingress.UID</td>
</tr>
</tbody>
</table>
| type      | No        | String | Network type of the load balancer. **public**: public network load balancer  
**inner**: private network load balancer  
Default value: inner |
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Mandatory</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bandwidth_name</td>
<td>Yes</td>
<td>String</td>
<td>Bandwidth name. The default value is <strong>cce-bandwidth-</strong>****. Value range: a string of 1 to 64 characters, including lowercase letters, digits, and underscores (_). The value must start with a lowercase letter and end with a lowercase letter or digit.</td>
</tr>
</tbody>
</table>
| bandwidth_chargemode | Yes       | String      | Bandwidth billing mode.  
- **bandwidth**: billed by bandwidth  
- **traffic**: billed by traffic |
| bandwidth_size     | Yes       | Integer     | Bandwidth size. The default value is 1 to 2000 Mbit/s. Set this parameter based on the bandwidth range allowed in your region. |
| bandwidth_sharetype | Yes       | String      | Whether the bandwidth is shared or dedicated. Options:  
- **PER**: dedicated bandwidth |
| eip_type           | Yes       | String      | EIP type.  
- **5_telcom**: China Telecom  
- **5_union**: China Unicom  
- **5_bgp**: dynamic BGP  
- **5_sbgp**: static BGP  
For dedicated load balancers, set this parameter to **5_gray**. |
| available_zone     | Yes       | Array of strings | (Mandatory) AZ where the load balancer is located. This parameter is available only for dedicated load balancers. |
| l4_flavor_name     | No        | String      | Flavor name of the layer-4 load balancer. This parameter is available only for dedicated load balancers. |
| l7_flavor_name     | Yes       | String      | (Mandatory) Flavor name of the layer-7 load balancer. This parameter is available only for dedicated load balancers. |
### ElbVirSubnetIDs

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Mandatory</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ElbVirSubnetIDs</td>
<td>No</td>
<td>Array of strings</td>
<td>Subnet where the backend server of the load balancer is located. If this parameter is left blank, the default cluster subnet is used. Load balancers occupy different number of subnet IP addresses based on their specifications. Therefore, you are not advised to use the subnet CIDR blocks of other resources (such as clusters and nodes) as the load balancer CIDR block. Default value: subnet where the cluster is located. This parameter is available only for dedicated load balancers.</td>
</tr>
</tbody>
</table>

#### vi ingress-test-secret.yaml

```yaml
apiVersion: v1
data:
tls.crt: LS0******tLS0tCg==
tls.key: LS0tL******0tLS0K
kind: Secret
metadata:
  annotations:
    description: test for ingressTLS secrets
    name: ingress-test-secret
    namespace: default
    type: IngressTLS
```

**NOTE**

In the preceding information, `tls.crt` and `tls.key` are only examples. Replace them with the actual files. The values of `tls.crt` and `tls.key` are the content encrypted using Base64.

### Step 3: Create a workload.

**kubectl create -f ingress-test-deployment.yaml**

If information similar to the following is displayed, the workload is being created:

```
deployment/ingress-test-deployment created
```

**kubectl get po**

If information similar to the following is displayed, the workload has been created successfully:

```
NAME                            READY   STATUS     RESTARTS   AGE
ingress-test-1627801589-r64pk  1/1      Running    0          6s
```

### Step 4: Create a secret.

**kubectl create -f ingress-test-secret.yaml**
If information similar to the following is displayed, the secret is being created:

```
secret/ingress-test-secret created
```

**kubectl get secrets**

If information similar to the following is displayed, the secret has been created successfully:

<table>
<thead>
<tr>
<th>NAME</th>
<th>TYPE</th>
<th>DATA</th>
<th>AGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>dash-dashboard</td>
<td>Opaque</td>
<td>0</td>
<td>7d</td>
</tr>
<tr>
<td>dash-dashboard-token-f2nbk</td>
<td>kubernetes.io/service-account-token</td>
<td>3</td>
<td>7d</td>
</tr>
<tr>
<td>default-secret</td>
<td>kubernetes.io/dockerconfigjson</td>
<td>1</td>
<td>8d</td>
</tr>
<tr>
<td>default-token-wfn4l</td>
<td>kubernetes.io/service-account-token</td>
<td>3</td>
<td>8d</td>
</tr>
<tr>
<td>paas.elb</td>
<td>cfe/secure-opaque</td>
<td>2</td>
<td>8d</td>
</tr>
<tr>
<td>ingress-test-secret</td>
<td>IngressTLS</td>
<td>2</td>
<td>13s</td>
</tr>
</tbody>
</table>

**Step 5** Create a Service.

**kubectl create -f ingress-test-svc.yaml**

If information similar to the following is displayed, the Service is being created:

```
service/ingress-test-svc created
```

**kubectl get svc**

If information similar to the following is displayed, the Service has been created successfully:

<table>
<thead>
<tr>
<th>NAME</th>
<th>TYPE</th>
<th>CLUSTER-IP</th>
<th>EXTERNAL-IP</th>
<th>PORT(S)</th>
<th>AGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>ingress-test</td>
<td>NodePort</td>
<td>10.247.189.207</td>
<td>&lt;none&gt;</td>
<td>8080:30532/TCP</td>
<td>5s</td>
</tr>
<tr>
<td>kubernetes</td>
<td>ClusterIP</td>
<td>10.247.0.1</td>
<td>&lt;none&gt;</td>
<td>443/TCP</td>
<td>3d</td>
</tr>
</tbody>
</table>

**kubectl create -f ingress-test-ingress.yaml**

If information similar to the following is displayed, the ingress is being created:

```
ingress/ingress-test-ingress created
```

**kubectl get ingress**

If information similar to the following is displayed, the ingress has been created successfully and the workload is accessible:

<table>
<thead>
<tr>
<th>NAME</th>
<th>HOSTS</th>
<th>ADDRESS</th>
<th>PORTS</th>
<th>AGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>ingress-test</td>
<td>*</td>
<td>10.154.76.63</td>
<td>80</td>
<td>10s</td>
</tr>
</tbody>
</table>

**Step 6** In the address bar of your browser, enter **http://10.154.76.63/healthz** and press **Enter**.

**10.154.76.63** indicates the IP address of the unified load balancer.
As the service logic becomes increasingly complex, many applications require network calls between modules. Traditional external firewalls or application-based firewalls cannot meet the requirements. Network policies are urgently needed between modules, service logic layers, or functional teams in a large cluster.

CCE has enhanced the Kubernetes-based network policy feature, allowing network isolation in a cluster by configuring network policies. This means that a firewall can be set between pods.

For example, to make a payment system accessible only to specified components for security purposes, you can configure network policies.

### Notes and Constraints
- Only clusters that use the tunnel network model support network policies.
- Egresses are not supported for network policies.
- Network isolation is not supported for IPv6 addresses.
- For clusters of v1.13 and v1.15 that use the tunnel network model, if the node OS kernel is CentOS, you need to upgrade the Open vSwitch version before configuring network policies. For details, see Upgrading the OS Kernel.

### Precautions
If no network policies have been configured for a workload, such as workload-1, other workloads in the same cluster can access workload-1.

### Creating a Network Policy

**Step 1** Log in to the CCE console. In the navigation pane, choose Resource Management > Network. On the Network Policies tab page, click Create Network Policy.

- **Network Policy Name**: Specify a network policy name.
- **Cluster Name**: Select a cluster to which the network policy belongs.
- **Namespace**: Select a namespace in which the network policy is applied.
- **Workload**
Click **Select Workload**. In the dialog box displayed, select a workload for which the network policy is to be created, for example, **workload-1**. Then, click **OK**.

- **Rules**: Click **Add Rule**, set the parameters listed in **Table 8-20**, and click **OK**.

**Table 8-20 Parameters for adding a rule**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Direction</td>
<td>Only <strong>Inbound</strong> is supported, indicating that the whitelisted workloads access the current workload (<strong>workload-1</strong> in this example).</td>
</tr>
<tr>
<td>Protocol</td>
<td>Select a protocol. Currently, the TCP and UDP protocols are supported. The ICMP protocol is not supported.</td>
</tr>
<tr>
<td>Destination Container Port</td>
<td>Specify a port on which the workload in the container image listens. The Nginx workload listens on port 80. If no container port is specified, all ports can be accessed by default.</td>
</tr>
</tbody>
</table>
| Whitelisted Workloads      | Select other workloads that can access the current workload. These workloads will access the current workload at the destination container port.  
  - **Namespace**: All workloads in the selected namespace(s) are added to the whitelist. That is, all workloads in the namespace(s) can access **workload-1**.  
  - **Workload**: The selected workloads can access **workload-1**. Only other workloads in the same namespace as **workload-1** can be selected. |

**Step 2** Click **Create**.

**Step 3** Repeat the preceding steps to add more network policies for the current workload when other ports need to be accessed by some workloads.

After the network policies are created, only the specified workloads or workloads in the specified namespaces can access the current workload.

----End

### 8.6 How Does CCE Communicate with Other HUAWEI CLOUD Services over an Intranet?

Common HUAWEI CLOUD services that communicate with CCE over the intranet include RDS, DMS, Kafka, RabbitMQ, VPN, and ModelArts. The following two scenarios are involved:

- In the same VPC network, CCE nodes can communicate with all services. When CCE nodes communicate with other services, check whether the security group rule in the inbound direction of the container CIDR block is
enabled on the peer end. (This restriction applies only to CCE clusters that use the VPC network model.)

- If CCE nodes and other services are in different VPCs, you can use a peering connection or VPN to connect two VPCs. Note that the two VPC CIDR blocks cannot overlap with the container CIDR block. In addition, you need to configure a return route for the peer VPC or private network. (This restriction applies only to CCE clusters that use the VPC network model.) For details, see VPC Peering Connection.

---

**NOTICE**

- This logic works for all HUAWEI CLOUD services.
- Clusters using the container tunnel network support internal communication of Services with no additional configuration required.
- Pay attention to the following points when configuring a cluster using the VPC network:
  1. The source IP address displayed on the peer end is the container IP address.
  2. Custom routing rules added on CCE enable containers to communicate with each other on nodes in a VPC.
  3. When a CCE container accesses other Services, check whether the inbound security group rule or firewall of the container CIDR block is configured on the peer end (destination end). For details, see Security Group Configuration Examples.
  4. If a VPN or VPC peering connection is used to enable communication between private networks, you need to configure a VPC peering connection route that points to the container CIDR block on the path and destination.

---

### 8.7 Configuring Containers to Access Public Networks

You can use the **NAT Gateway** service to enable container pods in a VPC to access the Internet. The NAT Gateway service provides source network address translation (SNAT), which translates private IP addresses to a public IP address by binding an elastic IP address (EIP) to the gateway, providing secure and efficient access to the Internet. **Figure 8-24** shows the SNAT architecture. The SNAT function allows the container pods in a VPC to access the Internet without being bound to an EIP. SNAT supports a large number of concurrent connections, which makes it suitable for applications involving a large number of requests and connections.
To enable a container pod to access the Internet, perform the following steps:

**Step 1**  Buy an EIP.
1. Log in to the management console.
2. Click in the upper left corner of the management console and select a region and a project.
3. Click at the upper left corner and choose Network > Elastic IP in the expanded list.
4. On the EIPs page, click Buy EIP.
5. Set parameters as required.

**NOTE**

Set Region to the region where container pods are located.
Figure 8-25 Buying an elastic IP address

Step 2 Buy a NAT gateway. For details, see Buying a NAT Gateway.

1. Log in to the management console.
2. Click in the upper left corner of the management console and select a region and a project.
3. Click at the upper left corner and choose Network > NAT Gateway in the expanded list.
4. On the displayed page, click Buy NAT Gateway.
5. Set parameters as required.

NOTE
Select the VPC and subnet that are the same as those of the namespace where container pods are located.
**Figure 8-26 Buying a NAT gateway**

![Figure 8-26 Buying a NAT gateway](image)

**Step 3** Configure an SNAT rule and bind the EIP to the subnet. For details, see Adding an SNAT Rule.

1. Log in to the management console.
2. Click 📍 in the upper left corner of the management console and select a region and a project.
3. Click 🕳️ at the upper left corner and choose Network > NAT Gateway in the expanded list.
4. On the page displayed, click the name of the NAT gateway for which you want to add the SNAT rule.
5. On the SNAT Rules tab page, click Add SNAT Rule.
6. Set parameters as required.

**NOTE**

Select the subnet that is the same as that of the namespace where container pods are located.
After the SNAT rule is configured, workloads can access public networks from the container. Public networks can be pinged from the container.

--- End
9 Storage (CSI)

9.1 Storage Basics

Volume

On-disk files in a container are ephemeral, which will be lost when the container crashes and are difficult to be shared between containers running together in a pod. The Kubernetes volume abstraction solves both of these problems. Volumes cannot be independently created, but defined in the pod spec.

All containers in a pod can access its volumes, but the volumes must have been mounted. Volumes can be mounted to any directory in a container.

The following figure shows how a storage volume is used between containers in a pod.
A volume will no longer exist if the pod to which it is mounted does not exist. However, files in the volume may outlive the volume, depending on the volume type.

**Volume Types**

Volumes can be classified into local volumes and cloud volumes.

- **Local volumes**
  CCE supports the following four types of local volumes. For details about how to use them, see [Using Local Disks as Storage Volumes](#).
  - emptyDir: an empty volume used for temporary storage
  - hostPath: mounts a directory on a host (node) to your container for reading data from the host.
  - ConfigMap: references the data stored in a ConfigMap for use by containers.
  - Secret: references the data stored in a secret for use by containers.

- **Cloud volumes**
  CCE supports the following types of cloud volumes:
  - Elastic Volume Service (EVS): [Using EVS Disks as Storage Volumes](#)
  - SFS Turbo: [Using SFS Turbo File Systems as Storage Volumes](#)
  - Object Storage Service (OBS): [Using OBS Buckets as Storage Volumes](#)
  - Scalable File Service (SFS): [Using SFS File Systems as Storage Volumes](#)

**CSI**

You can use Kubernetes Container Storage Interface (CSI) to develop plug-ins to support specific storage volumes.

CCE developed the storage add-on [everest](#) for you to use cloud storage services, such as EVS and SFS Turbo. You can install this add-on when creating a cluster. For details about CSI, see [CSI Overview](#).

**PV and PVC**

Kubernetes provides PersistentVolumes (PVs) and PersistentVolumeClaims (PVCs) to abstract details of how storage is provided from how it is consumed. You can request specific size of storage when needed, just like pods can request specific levels of resources (CPU and memory).

- A PV defines a directory for persistent storage on a host machine, for example, a mount directory of a file system.
- A PVC describes the attributes of the PV that a pod wants to use, such as the volume capacity and read/write permissions.

You can bind PVCs to PVs in a pod so that the pod can use storage resources. The following figure shows the relationship between PVs and PVCs.
PVs describes storage resources in the cluster. PVCs are requests for those resources. The following sections will describe how to use kubectl to connect to storage resources.

If you do not want to create storage resources or PVs manually, you can use StorageClasses.

**StorageClass**

Although PVs and PVCs allow you to consume abstract storage resources, you may need to configure multiple fields to create PVs and PVCs (such as the `csi` field structure in the PV), and PVs/PVCs are generally managed by the cluster administrator, which can be inconvenient when you need PVs with varying attributes for different problems.

To resolve this issue, Kubernetes supports dynamic PV provisioning to create PVs automatically. The cluster administrator can deploy a PV provisioner and define the corresponding StorageClass. In this way, developers can select the storage class to be created when creating a PVC. The PVC transfers the StorageClass to the PV provisioner, and the provisioner automatically creates a PV. In CCE, storage classes such as csi-disk, csi-nas, and csi-obs are supported. After `storageClassName` is added to a PVC, PVs can be automatically provisioned and underlying storage resources can be automatically created.

You can run the following command to query the storage classes that CCE supports. You can use the CSI plug-in provided by CCE to customize a storage class, which functions similarly as the default storage classes in CCE.

```
# kubectl get sc
NAME                PROVISIONER                      AGE                     
csi-disk            everest-csi-provisioner         17d          # Storage class for EVS disks
csi-disk-topology   everest-csi-provisioner         17d          # Storage class for EVS disks with delayed binding
csi-nas             everest-csi-provisioner         17d          # Storage class for SFS file systems
csi-obs             everest-csi-provisioner         17d          # Storage class for OBS buckets
csi-sfsturbo        everest-csi-provisioner         17d          # Storage class for SFS Turbo file systems
```
After a StorageClass is set, PVs can be automatically created and maintained. You only need to specify the StorageClass when creating a PVC, which greatly reduces the workload.

9.2 CSI Overview

In container storage, you can use different types of volumes and mount them to containers in pods as many as you want.

In CCE, container storage is backed both by Kubernetes-native objects, such as emptyDir, hostPath, secret, and ConfigMap, and by HUAWEI CLOUD storage services. These cloud storage services can be accessed via Container Storage Interface (CSI).

CCE allows you to use CDSI in your clusters by providing the everest add-on. This add-on is installed by default when the cluster version is Kubernetes 1.15 or later.

This topic will walk you through how to use CSI in CCE for container storage, and differences between CCE add-ons for CSI and FlexVolume implementation.

Cloud Services for Container Storage

CCE allows you to mount local and cloud storage volumes listed in Volume Types to your pods. Their features are described below.

Figure 9-2 Volume types supported by CCE
## Table 9-1 Detailed description of cloud storage services

<table>
<thead>
<tr>
<th>Dimension</th>
<th>EVS</th>
<th>SFS</th>
<th>OBS</th>
<th>SFS Turbo</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Definition</strong></td>
<td>EVS offers scalable block storage for cloud servers. With high reliability, high performance, and rich specifications, EVS disks can be used for distributed file systems, dev/test environments, data warehouses, and high-performance computing (HPC) applications.</td>
<td>Expandable to petabytes, SFS provides fully hosted shared file storage, highly available and stable to handle data- and bandwidth-intensive applications in HPC, media processing, file sharing, content management, and web services.</td>
<td>OBS is a stable, secure, and easy-to-use object storage service that lets you inexpensively store data of any format and size. You can use it in enterprise backup/ archiving, video on demand (VoD), video surveillance, and many other scenarios.</td>
<td>Expandable to 320 TB, SFS Turbo provides a fully hosted shared file storage, highly available and stable to support small files and applications requiring low latency and high IOPS. You can use SFS Turbo in high-traffic websites, log storage, compression/decompression, DevOps, enterprise OA, and containerized applications.</td>
</tr>
<tr>
<td><strong>Data storage logic</strong></td>
<td>Stores binary data and cannot directly store files. To store files, you need to format the file system first.</td>
<td>Stores files and sorts and displays data in the hierarchy of files and folders.</td>
<td>Stores objects. Files directly stored automatically generate the system metadata, which can also be customized by users.</td>
<td>Stores files and sorts and displays data in the hierarchy of files and folders.</td>
</tr>
<tr>
<td><strong>Access mode</strong></td>
<td>Accessible only after being mounted to ECSs or BMSs and initialized.</td>
<td>Mounted to ECSs or BMSs using network protocols. A network address must be specified or mapped to a local directory for access.</td>
<td>Accessible through the Internet or Direct Connect (DC). You need to specify the bucket address and use transmission protocols such as HTTP and HTTPS.</td>
<td>Supports the Network File System (NFS) protocol (NFSv3 only). You can seamlessly integrate existing applications and tools with SFS Turbo.</td>
</tr>
<tr>
<td>Dimension</td>
<td>EVS</td>
<td>SFS</td>
<td>OBS</td>
<td>SFS Turbo</td>
</tr>
<tr>
<td>-----------------------</td>
<td>--------------</td>
<td>--------------</td>
<td>--------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>Static provisioning</td>
<td>Supported</td>
<td>Supported</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>Dynamic provisioning</td>
<td>Supported</td>
<td>Supported</td>
<td>Supported</td>
<td>Not supported</td>
</tr>
<tr>
<td>Feature</td>
<td>Non-shared storage. Each volume can be mounted to only one node.</td>
<td>Shared storage featuring high performance and throughput</td>
<td>Shared, user-mode file system</td>
<td>Shared storage featuring high performance and bandwidth</td>
</tr>
<tr>
<td>Usage</td>
<td>HPC, enterprise core cluster applications, enterprise application systems, and dev/test</td>
<td>HPC, media processing, content management, web services, big data, and analysis applications</td>
<td>Big data analysis, static website hosting, online video on demand (VoD), gene sequencing, intelligent video surveillance, backup and archiving, and enterprise cloud boxes (web disks)</td>
<td>High-traffic websites, log storage, DevOps, and enterprise OA</td>
</tr>
<tr>
<td></td>
<td>NOTE</td>
<td>NOTE</td>
<td>NOTE</td>
<td></td>
</tr>
<tr>
<td></td>
<td>HPC apps here require high-speed and high-IOPS storage, such as industrial design and energy exploration. For more information, see Overview.</td>
<td>HPC apps here require high bandwidth and shared file storage, such as gene sequencing and image rendering. For more information, see Overview.</td>
<td>For more information, see Overview.</td>
<td>For more information, see Overview.</td>
</tr>
<tr>
<td>Capacity</td>
<td>TB</td>
<td>PB</td>
<td>EB</td>
<td>TB</td>
</tr>
<tr>
<td>Latency</td>
<td>1–2 ms</td>
<td>3-10 ms</td>
<td>10 ms</td>
<td>1–2 ms</td>
</tr>
<tr>
<td>IOPS/TPS</td>
<td>33,000 for a single disk</td>
<td>10,000 for a single file system</td>
<td>Tens of millions</td>
<td>100,000</td>
</tr>
<tr>
<td>Bandwidth</td>
<td>MB/s</td>
<td>GB/s</td>
<td>TB/s</td>
<td>GB/s</td>
</tr>
</tbody>
</table>
Notice on Using Add-ons

- To use the CSI plug-in (the everest add-on in CCE), your cluster must be using Kubernetes 1.15 or later. This add-on is installed by default when you create a cluster of v1.15 or later. The FlexVolume plug-in (the storage-driver add-on in CCE) is installed by default when you create a cluster of v1.13 or earlier.
- If your cluster is upgraded from v1.13 to v1.15, storage-driver is replaced by everest (v1.1.6 or later) for container storage. The takeover does not affect the original storage functions.
- In version 1.2.0 of the everest add-on, key authentication is optimized when OBS is used. After the everest add-on is upgraded from a version earlier than 1.2.0, you need to restart all workloads that use OBS in the cluster. Otherwise, workloads may not be able to use OBS.

Installing and Upgrading the CSI Plug-in

For details, see everest.
Differences Between CSI and FlexVolume Plug-ins

<table>
<thead>
<tr>
<th>Kubernetes Solution</th>
<th>CCE Add-on</th>
<th>Feature</th>
<th>Usage</th>
</tr>
</thead>
</table>
| CSI                 | everest    | CSI was developed as a standard for exposing arbitrary block and file storage systems to containerized workloads. Using CSI, third-party storage providers can deploy plugins exposing new storage systems in Kubernetes without having to touch the core Kubernetes code. In CCE, the everest add-on is installed by default in clusters of Kubernetes v1.15 and later to connect to HUAWEI CLOUD storage services (EVS, OBS, SFS, and SFS Turbo). The everest add-on consists of two parts:  
  - **everest-csi-controller** for storage volume creation, deletion, capacity expansion, and cloud disk snapshots  
  - **everest-csi-driver** for mounting, unmounting, and formatting storage volumes on nodes  
For details, see **everest**. |

The **everest** add-on is installed by default in clusters of **v1.15 and later**. CCE will mirror the Kubernetes community by providing continuous support for updated CSI capabilities.
FlexVolume is an out-of-tree plugin interface that has existed in Kubernetes since version 1.2 (before CSI). CCE provided FlexVolume volumes through the storage-driver add-on installed in clusters of Kubernetes v1.13 and earlier versions. This add-on connects clusters to HUAWEI CLOUD storage services (EVS, OBS, SFS, and SFS Turbo). For details, see storage-driver.

For clusters of v1.13 or earlier that have been created, the installed FlexVolume plug-in (the storage-driver add-on in CCE) can still be used. CCE stops providing update support for this add-on, and you are advised to upgrade these clusters.

### NOTE

- A cluster can use only one type of storage plug-ins.
- The FlexVolume plug-in cannot be replaced by the CSI plug-in in clusters of v1.13 or earlier. You can only upgrade these clusters. For details, see [Cluster Upgrade Between Major Versions](#).

### Checking Storage Add-ons

1. Log in to the CCE console.
2. In the navigation tree on the left, click **Add-ons**.
3. Click the **Add-on Instance** tab.
4. Select a cluster in the upper right corner. The default storage add-on installed during cluster creation is displayed.

![Checking which storage add-on was installed](image)

### 9.3 Using Local Disks as Storage Volumes

You can mount a file directory of the host where a container is located to a specified container path (the hostPath mode in Kubernetes) for persistent data.
storage. Alternatively, you can leave the source path empty (the emptyDir mode in Kubernetes), and a temporary directory of the host will be mounted to the mount point of the container for temporary storage.

**Using Local Volumes**

CCE supports four types of local volumes.

- **hostPath**: mounts a file directory of the host where the container is located to the specified mount point of the container. For example, if the container needs to access `/etc/hosts`, you can use a hostPath volume to map `/etc/hosts`.  

- **emptyDir**: stores data temporarily. An emptyDir volume is first created when a pod is assigned to a node, and exists as long as that pod is running on that node. When the pod is removed for any reason, the data in the emptyDir volume is deleted permanently.

- **ConfigMap**: A ConfigMap can be mounted as a volume, and all contents stored in its key are mounted onto the specified container directory. A ConfigMap is a type of resource that stores configuration data required by a workload. Its contents are user-defined. For details about how to create a ConfigMap, see [Creating a ConfigMap](#). For details about how to use a ConfigMap, see [Using a ConfigMap](#).

- **Secret**: You can store sensitive information, such as passwords, in secrets and mount them as files for use by pods. All contents in a secret are user-defined. For details about how to create a secret, see [Creating a Secret](#). For details about how to use a secret, see [Using a Secret](#).

The following describes how to mount these four types of volumes.

**hostPath**

You can mount a path on the host to a specified container path. A hostPath volume is usually used to **store workload logs permanently** or used by workloads that need to **access internal data structure of the Docker engine on the host**.

**Step 1** Create a workload by referring to [Creating a Deployment](#), [Creating a StatefulSet](#), or [Creating a DaemonSet](#). When adding a container, expand Data Storage, and click Add Local Volume.

**Step 2** Set parameters for adding a local volume, as listed in [Table 9-3](#).

**Table 9-3** Setting parameters for mounting a hostPath volume

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Select HostPath</td>
</tr>
</tbody>
</table>
### Parameter

**Host Path**

Path of the host to which the local volume is to be mounted, for example, `/etc/hosts`.

**NOTE**

Host Path cannot be set to the root directory `/`. Otherwise, the mounting fails. Mount paths can be as follows:

- `/opt/xxxx` (excluding `/opt/cloud`)
- `/mnt/xxxx` (excluding `/mnt/paas`)
- `/tmp/xxxx`
- `/var/xxxx` (excluding key directories such as `/var/lib`, `/var/script`, and `/var/paas`)
- `/xxxx` (it cannot conflict with the system directory, such as `bin`, `lib`, `home`, `root`, `boot`, `dev`, etc, `lost+found`, `mnt`, `proc`, `sbin`, `srv`, `tmp`, `var`, `media`, `opt`, `selinux`, `sys`, and `usr`).

Do not set this parameter to `/home/paas`, `/var/paas`, `/var/lib`, `/var/script`, `/mnt/paas`, or `/opt/cloud`. Otherwise, the system or node installation will fail.

### Add Container Path

Configure the following parameters:

1. **subPath**: Enter a subpath, for example, `tmp`.
   
   A subpath is used to mount a local disk so that the same data volume is used in a single pod. If this parameter is left blank, the root path is used by default.

2. **Container Path**: Enter the path of the container, for example, `/tmp`.
   
   This parameter indicates the container path to which a data volume will be mounted. The container path cannot be a system directory, such as `/` or `/var/run`. Otherwise, the container may not function properly. You are advised to mount the volume to an empty directory. If the directory is not empty, ensure that there are no files affecting container startup in the directory. Otherwise, such files will be replaced, resulting in failures to start the container and create the workload.

**NOTICE**

If the volume is mounted to a high-risk directory, you are advised to use an account with minimum permissions to start the container; otherwise, high-risk files on the host machine may be damaged.

3. **Permission**

   - **Read-only**: You can only read the data volumes in the path.
   
   - **Read/Write**: You can modify the data volumes in the path. Newly written data is not migrated if the container is migrated, which may cause a data loss.

Click **Add Container Path** to add multiple settings. Then, click **OK**.
emptyDir

emptyDir applies to temporary data storage, disaster recovery, and runtime data sharing. It will be deleted upon deletion or transfer of workload pods.

**Step 1** Create a workload by referring to Creating a Deployment, Creating a StatefulSet, or Creating a DaemonSet. After you add a container, expand Data Storage, and click Add Local Volume.

**Step 2** Set the local volume type to EmptyDir and set parameters for adding a local volume, as described in Table 9-4.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Select EmptyDir.</td>
</tr>
</tbody>
</table>
| Medium    | ● If you select Memory, the running speed is improved, but the storage capacity is limited by the memory size. This mode applies to scenarios where the data volume is small and the read and write efficiency is high.  
**NOTE**  
- If you select Memory, any files you write will count against your container’s memory limit. Pay attention to the memory quota. If the memory usage exceeds the threshold, OOM may occur.  
- If Memory is selected, the size of an emptyDir volume is 50% of the pod specifications and cannot be changed.  
- If Memory is not selected, emptyDir volumes will not occupy the system memory.  
● If Memory is not selected, data is stored in disks, which is applicable to a large amount of data with low requirements on read and write efficiency. |
Configure the following parameters:

1. **subPath**: Enter a subpath, for example, `tmp`. A subpath is used to mount a local disk so that the same data volume is used in a single pod. If this parameter is left blank, the root path is used by default.

2. **Container Path**: Enter the path of the container, for example, `/tmp`. This parameter indicates the container path to which a data volume will be mounted. The container path cannot be a system directory, such as `/` or `/var/run`. Otherwise, the container may not function properly. You are advised to mount the volume to an empty directory. If the directory is not empty, ensure that there are no files affecting container startup in the directory. Otherwise, such files will be replaced, resulting in failures to start the container and create the workload.

   **NOTICE**
   If the volume is mounted to a high-risk directory, you are advised to use an account with minimum permissions to start the container; otherwise, high-risk files on the host machine may be damaged.

3. **Permission**
   - **Read-only**: You can only read the data volumes in the path.
   - **Read/Write**: You can modify the data volumes in the path. Newly written data is not migrated if the container is migrated, which may cause a data loss.

Click **Add Container Path** to add multiple settings. Then, click **OK**.

---End

**ConfigMap**

The data stored in a ConfigMap can be referenced in a volume of type ConfigMap. You can mount such a volume to a specified container path. The platform supports the separation of workload codes and configuration files. ConfigMap volumes are used to store workload configuration parameters. Before that, you need to create ConfigMaps in advance. For details, see **Creating a ConfigMap**.

**Step 1** Create a workload by referring to **Creating a Deployment**, **Creating a StatefulSet**, or **Creating a DaemonSet**. After you add a container, expand **Data Storage**, and click **Add Local Volume**.

**Step 2** Set the local volume type to **ConfigMap** and set parameters for adding a local volume, as shown in **Table 9-5**.
### Table 9-5 Setting parameters for mounting a ConfigMap volume

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Select <strong>ConfigMap</strong>.</td>
</tr>
<tr>
<td>Option</td>
<td>Select the desired ConfigMap name.</td>
</tr>
<tr>
<td></td>
<td>A ConfigMap must be created in advance. For details, see <strong>Creating a ConfigMap</strong>.</td>
</tr>
<tr>
<td>Add Container Path</td>
<td>Configure the following parameters:</td>
</tr>
<tr>
<td></td>
<td>1. <strong>subPath</strong>: Enter a subpath, for example, <strong>tmp</strong>.</td>
</tr>
<tr>
<td></td>
<td>A subpath is used to mount a local disk so that the same data volume is used in a single pod. If this parameter is left blank, the root path is used by default.</td>
</tr>
<tr>
<td></td>
<td>2. <strong>Container Path</strong>: Enter the path of the container, for example, <strong>/tmp</strong>.</td>
</tr>
<tr>
<td></td>
<td>This parameter indicates the container path to which a data volume will be mounted. The container path cannot be a system directory, such as <strong>/</strong> or <strong>/var/run</strong>. Otherwise, the container may not function properly. You are advised to mount the volume to an empty directory. If the directory is not empty, ensure that there are no files affecting container startup in the directory. Otherwise, such files will be replaced, resulting in failures to start the container and create the workload.</td>
</tr>
<tr>
<td></td>
<td><strong>NOTICE</strong></td>
</tr>
<tr>
<td></td>
<td>If the volume is mounted to a high-risk directory, you are advised to use an account with minimum permissions to start the container; otherwise, high-risk files on the host machine may be damaged.</td>
</tr>
<tr>
<td></td>
<td>3. Set the permission to <strong>Read-only</strong>. Data volumes in the path are read-only.</td>
</tr>
<tr>
<td></td>
<td>Click <strong>Add Container Path</strong> to add multiple settings. Then, click <strong>OK</strong>.</td>
</tr>
</tbody>
</table>

#### Secret

You can mount a secret as a volume to the specified container path. Contents in a secret are user-defined. Before that, you need to create a secret. For details, see **Creating a Secret**.

**Step 1** Create a workload by referring to **Creating a Deployment**, **Creating a StatefulSet**, or **Creating a DaemonSet**. After you add a container, expand **Data Storage**, and click **Add Local Volume**.

**Step 2** Set the local volume type to **Secret** and set parameters for adding a local volume, as shown in **Table 9-6**.
### Table 9-6 Setting parameters for mounting a secret volume

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Secret.</td>
</tr>
<tr>
<td>Secret</td>
<td>Select the desired secret name. A secret must be created in advance. For details, see <a href="#">Creating a Secret</a>.</td>
</tr>
<tr>
<td>Add Container Path</td>
<td>Configure the following parameters:</td>
</tr>
<tr>
<td></td>
<td>1. <strong>subPath</strong>: Enter a subpath, for example, <code>tmp</code>. A subpath is used to mount a local disk so that the same data volume is used in a single pod. If this parameter is left blank, the root path is used by default.</td>
</tr>
<tr>
<td></td>
<td>2. <strong>Container Path</strong>: Enter the path of the container, for example, <code>/tmp</code>. This parameter indicates the container path to which a data volume will be mounted. The container path cannot be a system directory, such as <code>/</code> or <code>/var/run</code>. Otherwise, the container may not function properly. You are advised to mount the volume to an empty directory. If the directory is not empty, ensure that there are no files affecting container startup in the directory. Otherwise, such files will be replaced, resulting in failures to start the container and create the workload. <strong>NOTICE</strong> If the volume is mounted to a high-risk directory, you are advised to use an account with minimum permissions to start the container; otherwise, high-risk files on the host machine may be damaged.</td>
</tr>
<tr>
<td></td>
<td>3. Set the permission to <strong>Read-only</strong>. Data volumes in the path are read-only.</td>
</tr>
<tr>
<td></td>
<td>Click <strong>Add Container Path</strong> to add multiple settings. Then, click <strong>OK</strong>.</td>
</tr>
</tbody>
</table>

---

**Mounting a hostPath Volume Using `kubectl`**

You can use `kubectl` to mount a file directory of the host where the container is located to a specified mount path of the container.

**Step 1** Use `kubectl` to connect to the cluster. For details, see [Connecting to a Cluster Using `kubectl`](#).

**Step 2** Run the following commands to configure the `hostPath-pod-example.yaml` file, which is used to create a pod.

```bash
touch hostPath-pod-example.yaml
vi hostPath-pod-example.yaml
```

Mount the hostPath volume for the Deployment. The following is an example:
apiVersion: apps/v1
kind: Deployment
metadata:
  name: hostpath-pod-example
namespace: default
spec:
  replicas: 1
  selector:
    matchLabels:
      app: hostpath-pod-example
  template:
    metadata:
      labels:
        app: hostpath-pod-example
    spec:
      containers:
      - image: nginx
        name: container-0
        volumeMounts:
        - mountPath: /tmp
          name: hostpath-example
        imagePullSecrets:
        - name: default-secret
          restartPolicy: Always
        volumes:
        - name: hostpath-example
          hostPath:
            path: /tmp/test

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mountPath</td>
<td>Mount path of the container. In this example, the volume is mounted to the /tmp directory.</td>
</tr>
<tr>
<td>hostPath</td>
<td>Host path. In this example, the host path is /tmp/test.</td>
</tr>
</tbody>
</table>

**Table 9-7 Local disk storage dependency parameters**

**Note**

The `spec.template.spec.containers.volumeMounts.name` and `spec.template.spec.volumes.name` must be consistent because they have a mapping relationship.

**Step 3** Run the following command to create a pod:

```bash
kubectl create -f hostPath-pod-example.yaml
```

**Step 4** Verify the mounting.

1. Query the pod name of the workload (`hostpath-pod-example` is used as an example).
   ```bash
   kubectl get po | grep hostpath-pod-example
   ```
   **Expected outputs:**
   
   hostpath-pod-example-55c8d4dc59-md5d9   1/1     Running   0          35s

2. Create the `test1` file in the container mount path `/tmp`.
   ```bash
   kubectl exec hostpath-pod-example-55c8d4dc59-md5d9 -- touch /tmp/test1
   ```

3. Verify that the file is created in the host path `/tmp/test/`.
   ```bash
   ll /tmp/test/
   ```
   **Expected outputs:**
4. Create the test2 file in the host path /tmp/test/
   touch /tmp/test/test2

5. Verify that the file is created in the container mount path.
   kubectl exec hostpath-pod-example-55c8d4dc59-md5d9 -- ls -l /tmp
   
   Expected outputs:
   -rw-r--r-- 1 root root 0 Jun  1 08:12 test1
   -rw-r--r-- 1 root root 0 Jun  1 08:14 test2

---End

9.4 Using EVS Disks as Storage Volumes

9.4.1 Overview

To achieve persistent storage, CCE allows you to mount the storage volumes created from Elastic Volume Service (EVS) disks to a path of a container. When the container is migrated, the mounted EVS volumes are also migrated. By using EVS volumes, you can mount the remote file directory of storage system into a container so that data in the data volume is permanently preserved even when the container is deleted.

Figure 9-4 Mounting EVS volumes to CCE

Description

- **User-friendly**: Similar to formatting disks for on-site servers in traditional layouts, you can format block storage (disks) mounted to cloud servers, and create file systems on them.
- **Data isolation**: Each server uses an independent block storage device (disk).
- **Private network**: User can access data only in private networks of data centers.
- **Capacity and performance**: The capacity of a single volume is limited (TB-level), but the performance is excellent (ms-level read/write I/O latency).
• **Restriction:** EVS disks that have partitions or have non-ext4 file systems cannot be imported.

• **Use cases:** You can use such volumes for Deployments, jobs, and StatefulSets deployed with a single pod. Common use cases include high-performance computing (HPC), enterprise core cluster applications, enterprise OA systems, and dev/test.

### 9.4.2 Using EVS Volumes

#### Prerequisites
You have created a CCE cluster and installed the CSI plug-in (*everest*) in the cluster.

#### Notes and Constraints
- By default, CCE creates EVS disks billed in **pay-per-use** mode. To use EVS disks billed in **yearly/monthly** mode, see *Yearly/Monthly-Billed EVS Disks*.
- EVS disks cannot be attached across AZs and cannot be used by multiple workloads, multiple pods of the same workload, or multiple jobs.
- Data in a shared disk cannot be shared between nodes in a CCE cluster. If the same EVS disk is attached to multiple nodes, read and write conflicts and data cache conflicts may occur. When creating a Deployment, you are advised to create only one pod if you want to use EVS disks.
- When you create a StatefulSet and add a cloud storage volume, existing EVS volumes cannot be used.
- EVS disks that have partitions or have non-ext4 file systems cannot be imported.
- Container storage in CCE clusters of Kubernetes 1.13 or later version supports encryption. Currently, E2E encryption is supported only in certain regions.
- EVS volumes cannot be created in specified enterprise projects. Only the default enterprise project is supported.

#### Buying an EVS Disk

**Step 1** Log in to the CCE console. In the navigation pane, choose **Resource Management > Storage.** Click **Buy EVS Disk.**

**Step 2** Configure basic disk information. *Table 9-8* describes the parameters.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>* PVC Name</td>
<td><strong>New PVC Name:</strong> name of the PVC to be created. A storage volume is automatically created when a PVC is created. One PVC corresponds to one storage volume. The storage volume name is automatically generated when the PVC is created.</td>
</tr>
<tr>
<td>Cluster Name</td>
<td>Cluster where the EVS disk is deployed.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Namespace</td>
<td>Select the namespace where the EVS disk is deployed. If you do not need to select a namespace, retain the default value.</td>
</tr>
<tr>
<td>Volume Capacity (GB)</td>
<td>Size of the storage to be created.</td>
</tr>
</tbody>
</table>
| Access Mode   | Access permissions of user applications on storage resources (PVs).  
|                | - **ReadWriteOnce** (RWO): The volume can be mounted as read-write by a single node, and data reading and writing are supported based on a non-shared EVS volume. EVS volumes in RWO mode are supported since v1.13.10-r1. |
| Primary AZ    | AZ to which the volume belongs.                                                                                                             |
| Type          | Type of the new EVS disk.  
|                | - **High I/O**: uses serial attached SCSI (SAS) drives to store data.  
|                | - **Ultra-high I/O**: uses solid state disk (SSD) drives to store data.  
|                | Click [here](#) to learn how to select a disk type.                                                                                         |
| Storage Format | The default value is **CSI** and cannot be changed.                                                                                         |
| Encryption    | **KMS Encryption** is deselected by default.                                                                                              |
|               | After **KMS Encryption** is selected, Key Management Service (KMS), an easy-to-use and highly secure cloud service for your keys, will be used for EVS disks. If no agency has been created, click **Create Agency** and set the following parameters:  
|               | - **Agency Name**: Agencies can be used to assign permissions to trusted accounts or cloud services for a specific period of time. If no agency is created, click **Create Agency**. The agency name **EVSAccessKMS** indicates that EVS is granted the permission to access KMS. After EVS is authorized successfully, it can obtain KMS keys to encrypt and decrypt EVS systems.  
|               | - **Key Name**: After a key is created, it can be loaded and used in containerized applications. For details on how to create a key, see **Creating a CMK**.  
|               | - **Key ID**: generated by default. This function is supported only for clusters of v1.13.10 and later in certain regions.               |

**Step 3** Click **Buy Now**. Review your order, click **Submit**, and wait until the creation is successful.
The file system is displayed in the list. When its status becomes **Normal**, the file system is created successfully.

**Step 4** Click the volume name to view detailed information about the volume.

----End

### Adding an EVS Volume

**Step 1** Create a workload or job by referring to *Creating a Deployment*, *Creating a StatefulSet*, or *Creating a Job*. During creation, expand **Data Storage** after adding a container. On the **Cloud Volume** tab page, click **Add Cloud Volume**.

**Step 2** Set the storage volume type to **EVS**.

### Table 9-9 Parameters required for mounting an EVS volume

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Type</strong></td>
<td><strong>EVS</strong>: You can use EVS disks the same way you use traditional hard disks on servers. EVS disks deliver higher data reliability and I/O throughput and are easy to use. They can be used for file systems, databases, or other system software and applications that require block storage resources.</td>
</tr>
<tr>
<td></td>
<td><strong>CAUTION</strong></td>
</tr>
<tr>
<td></td>
<td>● To attach an EVS disk to a workload, you must set the number of pods to 1 when creating the workload. If multiple pods are configured, you cannot attach EVS disks.</td>
</tr>
<tr>
<td></td>
<td>● When you create a StatefulSet and add a cloud storage volume, existing EVS volumes cannot be used.</td>
</tr>
<tr>
<td></td>
<td>● EVS disks cannot be attached across AZs and cannot be used by multiple workloads, multiple pods of the same workload, or multiple tasks.</td>
</tr>
<tr>
<td><strong>Allocation Mode</strong></td>
<td>Select a created disk. If no disk is available, follow the prompts to create one.</td>
</tr>
<tr>
<td>Manual</td>
<td>For the same cluster and namespace, you can use an existing storage volume when creating a Deployment (with <strong>Allocation Mode</strong> set to <strong>Manual</strong>).</td>
</tr>
<tr>
<td></td>
<td>When creating a StatefulSet, you can only use a volume automatically allocated by the system (only <strong>Automatic</strong> is available for <strong>Allocation Mode</strong>).</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| Automatic     | If you select **Automatic**, you need to configure the following items:  
1. **Access Mode**: permissions of user applications on storage resources (PVs).  
   - **ReadWriteOnce** (RWO): The volume can be mounted as read-write by a single node, and data reading and writing are supported based on a non-shared EVS volume. EVS volumes in RWO mode are supported since v1.13.10-r1.  
2. **Availability Zone**: AZ where the storage volume is located. Only the AZ where the node is located can be selected.  
3. **Sub-Type**: Select a storage subtype.  
   - **High I/O**: uses serial attached SCSI (SAS) drives to store data.  
   - **Ultra-high I/O**: uses solid state disk (SSD) drives to store data.  
4. **Storage Capacity**: Enter the storage capacity in the unit of GB. Ensure that the storage capacity quota is not exceeded; otherwise, creation will fail.  
5. **Storage Format**: The default value is **CSI**. The container storage interface (CSI) is used to establish a set of standard storage management interfaces between Kubernetes and external storage systems to provide storage services for containers.  
6. After you select **KMS Encryption**, Key Management Service (KMS), an easy-to-use and highly secure service, will be enabled for EVS disks. This function is supported only for clusters of v1.13.10 and later in certain regions. If no agency has been created, click **Create Agency** and set the following parameters:  
   - **Agency Name**: Agencies can be used to assign permissions to trusted accounts or cloud services for a specific period of time. If no agency is created, click **Create Agency**. The agency name **EVSAccessKMS** indicates that EVS is granted the permission to access KMS. After EVS is authorized successfully, it can obtain KMS keys to encrypt and decrypt EVS systems.  
   - **Key Name**: After a key is created, it can be loaded and used in containerized applications. For details on how to create a key, see **Creating a CMK**.  
   - **Key ID**: generated by default. |
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| Add Container Path  | 1. Click **Add Container Path**.  
2. **Container Path**: Enter the container path to which the data volume is mounted.  
**NOTICE**  
- Do not mount a data volume to a system directory such as `/` or `/var/run`; this action may cause a container error to occur. You are advised to mount the volume to an empty directory. If the directory is not empty, ensure that there are no files affecting container startup in the directory. Otherwise, such files will be replaced, resulting in failures to start the container and create the workload.  
- If the volume is mounted to a high-risk directory, you are advised to use an account with minimum permissions to start the container; otherwise, high-risk files on the host machine may be damaged.  
3. Set permissions.  
   - **Read-only**: You can only read the data volumes mounted to the path.  
   - **Read/Write**: You can modify the data volumes mounted to the path. Newly written data is not migrated if the container is migrated, which may cause a data loss. |

![Figure 9-5 Adding a cloud volume](image)

**Step 3** Click **OK**.

---End

**Importing an EVS Disk**

CCE allows you to import existing EVS disks.
An EVS disk can be imported into only one namespace. If an EVS disk has been imported into a namespace, it is invisible in other namespaces and cannot be imported again. If you want to import an EVS disk that has file system (ext4) formatted, ensure that no partition has been created for the disk. Otherwise, data may be lost.

**Step 1** Log in to the CCE console. In the navigation pane, choose **Resource Management > Storage**. On the **EVS** tab page, click **Import**.

**Step 2** Select one or more EVS disks that you want to import. Then, click **OK**.

---End

**Yearly/Monthly-Billed EVS Disks**

If you want to change the billing mode of an EVS disk from pay-per-use to yearly/monthly after the disk is created or imported, perform the following operations:

**Method 1:**
1. Create an EVS disk on the CCE console. The disk will be billed on a pay-per-use basis by default.
2. On the CCE console, choose **Resource Management > Storage > EVS**, and click **More > Change Billing Mode** at the row of the target EVS disk. This change of billing mode takes some time.

**Method 2:**
1. Create a yearly/monthly-billed EVS disk with the required capacity on the cloud server console.
2. Back on the **EVS** tab page of the CCE console, click **Import**. Select the EVS disk you just created, and click **OK**.

---CAUTION---

When a yearly/monthly-billed EVS disk is deleted on CCE, it is only unbound from CCE. If you want to permanently delete the EVS disk, unbind it on CCE and then unsubscribe from it in the Billing Center.

---End---

**Unbinding an EVS Disk**

After an EVS volume is successfully created or imported, the EVS volume is automatically bound to the current cluster and cannot be used by other clusters. When the volume is unbound from the cluster, other clusters can still use the volume.

If the EVS volume has been mounted to a workload, it cannot be unbound from the cluster.

**Step 1** Log in to the CCE console. In the navigation pane, choose **Resource Management > Storage**. In the EVS disk list, click **Unbind** next to the target EVS disk.

**Step 2** Confirm the unbinding, and click **OK**.

---End---
Related Operations

After an EVS volume is created, you can perform operations described in Table 9-10.

Table 9-10 Other operations

<table>
<thead>
<tr>
<th>Operation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deleting an EVS volume</td>
<td>1. Select the EVS volume to be deleted and click <strong>Delete</strong> in the <strong>Operation</strong> column.</td>
</tr>
<tr>
<td></td>
<td>2. Follow the prompts to delete the EVS volume.</td>
</tr>
</tbody>
</table>

9.4.3 (kubectl) Automatically Creating an EVS Volume

Scenario

CCE supports creating EVS volumes through PersistentVolumeClaims (PVCs).

Prerequisites

- You have created a CCE cluster and installed the CSI plug-in (**everest**) in the cluster.
- You have configured kubectl for your cluster. For details, see Connecting to a Cluster Using kubectl.

Notes and Constraints

The following configuration example applies to clusters of Kubernetes 1.15 or later.

Procedure

Step 1 Use kubectl to connect to the cluster. For details, see Connecting to a Cluster Using kubectl.

Step 2 Run the following commands to configure the pvc-evs-auto-example.yaml file, which is used to create a PVC.

```bash
touch pvc-evs-auto-example.yaml
vi pvc-evs-auto-example.yaml
```

Example YAML file:

```yaml
apiVersion: v1
category: PersistentVolumeClaim
metadata:
  name: pvc-evs-auto-example
  namespace: default
  annotations:
    everest.io/disk-volume-type: SSD
  labels:
    failure-domain.beta.kubernetes.io/region: ap-southeast-1
    failure-domain.beta.kubernetes.io/zone:
```
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 10Gi
  storageClassName: csi-disk

Table 9-11 Key parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>everest.io/disk-volume-type</td>
<td>EVS disk type. All letters are in uppercase. Supported values: High I/O (SAS) and Ultra-high I/O (SSD)</td>
</tr>
<tr>
<td>failure-domain.beta.kubernetes.io/region</td>
<td>Region where the cluster is located. For details about the value of <code>region</code>, see Regions and Endpoints.</td>
</tr>
<tr>
<td>failure-domain.beta.kubernetes.io/zone</td>
<td>AZ where the EVS volume is created. It must be the same as the AZ planned for the workload. For details about the value of <code>zone</code>, see Regions and Endpoints.</td>
</tr>
<tr>
<td>storage</td>
<td>Storage capacity in the unit of Gi.</td>
</tr>
<tr>
<td>storageClassName</td>
<td>Name of the associated Kubernetes storage class that dynamically creates the storage volume. Name of the storage class associated with the CSI used by the v1.15 cluster (csi-disk).</td>
</tr>
<tr>
<td>accessModes</td>
<td>Read/write mode of the volume. Clusters of v1.15 support only non-shared volumes. Set this parameter to <code>ReadWriteOnce</code>.</td>
</tr>
</tbody>
</table>

Step 3  Run the following command to create the PVC.

`kubectl create -f pvc-evs-auto-example.yaml`

After the command is executed, an EVS volume is created in the region where the cluster is located. You can choose Storage > EVS on the CCE console to view the EVS volume. Alternatively, you can view the EVS disk based on the volume name on the EVS console.

----End
9.4.4 (kubectl) Creating a PV from an Existing EVS Disk

Scenario

CCE allows you to create a PersistentVolume (PV) using an existing EVS disk. After the PV is created, you can create a PersistentVolumeClaim (PVC) and bind it to the PV.

Prerequisites

- You have created a CCE cluster and installed the CSI plug-in (*everest*) in the cluster.
- You have configured kubectl for your cluster. For details, see [*Connecting to a Cluster Using kubectl*](#).

Notes and Constraints

The following configuration example applies to clusters of Kubernetes 1.15 or later.

Procedure

**Step 1** Log in to the EVS console, create an EVS disk, and record the volume ID, capacity, and disk type of the EVS disk.

**Step 2** Use kubectl to connect to the cluster. For details, see [*Connecting to a Cluster Using kubectl*](#).

**Step 3** Create two YAML files for creating the PersistentVolume (PV) and PersistentVolumeClaim (PVC). Assume that the file names are *pv-evs-example.yaml* and *pvc-evs-example.yaml*.

```
touch pv-evs-example.yaml pvc-evs-example.yaml
```

- `vi pv-evs-example.yaml`

  **Example YAML file for the PV:**

  ```yaml
  apiVersion: v1
  kind: PersistentVolume
  metadata:
    labels:
      failure-domain.beta.kubernetes.io/region: ap-southeast-1
      failure-domain.beta.kubernetes.io/zone:
  annotations:
    pv.kubernetes.io/provisioned-by: everest-csi-provisioner
  name: pv-evs-example
  spec:
    accessModes:
    - ReadWriteOnce
    capacity:
      storage: 10Gi
    claimRef:
      apiVersion: v1
      kind: PersistentVolumeClaim
      name: pvc-evs-example
      namespace: default
    csi:
      driver: disk.csi.everest.io
      fsType: ext4
  ```

  **Example YAML file for the PVC:**

  ```yaml
  apiVersion: v1
  kind: PersistentVolumeClaim
  metadata:
    name: pvc-evs-example
    namespace: default
  spec:
    accessModes:
    - ReadWriteOnce
    resources:
      requests:
        storage: 10Gi
  ```
### Table 9-12 Key parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>everest.io/disk-volume-type</td>
<td>EVS disk type. All letters are in uppercase. Supported values: High I/O (SAS) and Ultra-high I/O (SSD)</td>
</tr>
<tr>
<td>failure-domain.beta.kubernetes.io/region</td>
<td>Region where the cluster is located. For details about the value of <code>region</code>, see <a href="#">Regions and Endpoints</a>.</td>
</tr>
<tr>
<td>failure-domain.beta.kubernetes.io/zone</td>
<td>AZ where the EVS volume is created. It must be the same as the AZ planned for the workload. For details about the value of <code>zone</code>, see <a href="#">Regions and Endpoints</a>.</td>
</tr>
<tr>
<td>storage</td>
<td>EVS volume capacity in the unit of Gi.</td>
</tr>
<tr>
<td>storageClassName</td>
<td>Name of the associated Kubernetes storage class that dynamically creates the storage volume. The disk type must be <code>csi-disk</code>.</td>
</tr>
<tr>
<td>accessModes</td>
<td>Read/write mode of the volume. Clusters of v1.15 support only non-shared volumes. Set this parameter to <code>ReadWriteOnce</code>.</td>
</tr>
<tr>
<td>driver</td>
<td>Dependent storage driver for the mounting. For EVS disks, set this parameter to <code>disk.csi.everest.io</code>.</td>
</tr>
<tr>
<td>volumeHandle</td>
<td>Volume ID of the EVS disk. To obtain the volume ID, log in to the <a href="#">Cloud Server Console</a>. In the navigation pane, choose <strong>Elastic Volume Service</strong> &gt; <strong>Disks</strong>. Click the name of the target EVS disk to go to its details page. On the <strong>Summary</strong> tab page, click the copy button after <strong>ID</strong>.</td>
</tr>
<tr>
<td>everest.io/disk-mode</td>
<td>Device type of the EVS disk. The value can be <strong>SCSI</strong>.</td>
</tr>
<tr>
<td>spec.claimRef.apiVersion</td>
<td>The value is fixed at <strong>v1</strong>.</td>
</tr>
</tbody>
</table>
### Table 9-13 Key parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>everest.io/disk-volume-type</td>
<td>EVS disk type. All letters are in uppercase. Supported values: High I/O (SAS) and Ultra-high I/O (SSD)</td>
</tr>
<tr>
<td>failure-domain.beta.kubernetes.io/region</td>
<td>Region where the cluster is located. For details about the value of region, see Regions and Endpoints.</td>
</tr>
<tr>
<td>failure-domain.beta.kubernetes.io/zone</td>
<td>AZ where the EVS volume is created. It must be the same as the AZ planned for the workload. For details about the value of zone, see Regions and Endpoints.</td>
</tr>
<tr>
<td>storage</td>
<td>Requested PVC capacity, in Gi. The value must be the same as the storage size of the existing PV.</td>
</tr>
<tr>
<td>volumeName</td>
<td>Name of the PV.</td>
</tr>
</tbody>
</table>

- **vi pvc-evs-example.yaml**

**Example YAML file for the PVC:**

```yaml
apiVersion: v1
class: PersistentVolumeClaim
metadata:
  labels:
    failure-domain.beta.kubernetes.io/region: ap-southeast-1
    failure-domain.beta.kubernetes.io/zone: ap-southeast-1a
  annotations:
    everest.io/disk-volume-type: SAS
    volume.beta.kubernetes.io/storage-provisioner: everest-csi-provisioner
name: pvc-evs-example
namespace: default
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 10Gi
  volumeName: pv-evs-example
```

---

---
Step 4  Create a PV.

```bash
kubectl create -f pv-evs-example.yaml
```

Step 5  Create a PVC.

```bash
kubectl create -f pvc-evs-example.yaml
```

After the operation is successful, choose Resource Management > Storage to view the created PVC. You can also view the EVS disk by name on the EVS page.

Step 6  (Optional) Add the metadata associated with the cluster to ensure that the EVS disks associated with the mounted static PV are not deleted when the node or cluster is deleted.

⚠️ CAUTION

If you skip this step in this example or when creating a static PV or PVC, ensure that the EVS disk associated with the static PV has been unbound from the node before you delete the node.

1. Obtain the tenant token. For details, see Obtaining a User Token.
2. Obtain the EVS access address EVS_ENDPOINT. For details, see Regions and Endpoints.
3. Add the metadata associated with the cluster to the EVS disk associated with the EVS static PV. For details about the API, see Adding Metadata of an EVS Disk.

```bash
curl -X POST ${EVS_ENDPOINT}/v2/${project_id}/volumes/${volume_id}/metadata --insecure \
-d '{"metadata":{"cluster_id": "$cluster_id", "namespace": "$pvc_namespace"}}' \
-H 'Accept:application/json' -H 'Content-Type:application/json; charset=utf8' \
-H 'X-Auth-Token:${TOKEN}'
```

<table>
<thead>
<tr>
<th>Table 9-14 Key parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Parameter</strong></td>
</tr>
<tr>
<td>EVS_ENDPOINT</td>
</tr>
<tr>
<td>project_id</td>
</tr>
<tr>
<td>volume_id</td>
</tr>
<tr>
<td>Parameter</td>
</tr>
<tr>
<td>--------------</td>
</tr>
<tr>
<td>cluster_id</td>
</tr>
<tr>
<td>pvc_namespace</td>
</tr>
<tr>
<td>TOKEN</td>
</tr>
</tbody>
</table>

**Figure 9-6 Obtaining the disk ID**

![Figure 9-6 Obtaining the disk ID](image)

**Figure 9-7 Obtaining the cluster ID**

![Figure 9-7 Obtaining the cluster ID](image)

For example, run the following command:

curl -X POST https://evs.cn-north-4.myhuaweicloud.com:443/v2/060576866680d5762f52c0150e726aa7/volumes/69c9619d-174c-4c41-837e-31b892604e14/metadata --insecure \
After the request is executed, run the following command to check whether the EVS disk has been associated with the metadata of the cluster:

```bash
curl -X GET ${{EVS_ENDPOINT}}/v2/${project_id}/volumes/${volume_id}/metadata --insecure \
-H 'X-Auth-Token:${TOKEN}'
```

For example, run the following command:

```bash
curl -X GET https://evs.cn-north-4.myhuaweicloud.com/v2/060576866680d5762f52c0150e726aa7/volumes/69c9619d-174c-4c41-837e-31b892604e14/metadata --insecure \
-H 'X-Auth-Token:MIIPeAYJ***9t1c31ASaQ=='
```

The command output displays the current metadata of the EVS disk.

```
{
  "metadata": {
    "namespace": "default",
    "cluster_id": "71e8277e-80c7-11ea-925c-0255ac100442",
    "hw:passthrough": "true"
  }
}
```

----End

9.4.5 (kubectl) Creating a Pod Mounted with an EVS Volume

**Scenario**

After an EVS volume is created or imported to CCE, you can mount it to a workload.

---

**NOTICE**

EVS disks cannot be attached across AZs. Before mounting a volume, you can run the `kubectl get pvc` command to query the available PVCs in the AZ where the current cluster is located.

---

**Prerequisites**

You have created a CCE cluster and installed the CSI plug-in (`everest`) in the cluster.

**Notes and Constraints**

The following configuration example applies to clusters of Kubernetes 1.15 or later.

**Procedure**

1. **Step 1** Use `kubectl` to connect to the cluster. For details, see Connecting to a Cluster Using `kubectl`.

2. **Step 2** Run the following commands to configure the `evs-deployment-example.yaml` file, which is used to create a Deployment.
Example of mounting an EVS volume to a Deployment (PVC-based, shared volume):

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: evs-deployment-example
  namespace: default
spec:
  replicas: 1
  selector:
    matchLabels:
      app: evs-deployment-example
template:
  metadata:
    labels:
      app: evs-deployment-example
  spec:
    containers:
    - image: nginx
      name: container-0
      volumeMounts:
        - mountPath: /tmp
          name: pvc-evs-example
      imagePullSecrets:
        - name: default-secret
    restartPolicy: Always
    volumes:
    - name: pvc-evs-example
      persistentVolumeClaim:
        claimName: pvc-evs-auto-example
```

<table>
<thead>
<tr>
<th>Parent Parameter</th>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>spec.template.spec.containers.volumeMounts.name</td>
<td>Name</td>
<td>Name of the volume mounted to the container.</td>
</tr>
<tr>
<td>spec.template.spec.containers.volumeMounts.mountPath</td>
<td>MountPath</td>
<td>Mount path of the container. In this example, the volume is mounted to the /tmp directory.</td>
</tr>
<tr>
<td>spec.template.spec.volumes.name</td>
<td>Name</td>
<td>Name of the volume.</td>
</tr>
<tr>
<td>spec.template.spec.volumes.persistentVolumeClaim.claimName</td>
<td>claimName</td>
<td>Name of an existing PVC.</td>
</tr>
</tbody>
</table>

**NOTE**

`spec.template.spec.containers.volumeMounts.name` and `spec.template.spec.volumes.name` must be consistent because they have a mapping relationship.

Mounting an EVS volume to a StatefulSet (PVC template-based, non-shared volume):
Example YAML:

apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: evs-statefulset-example
  namespace: default
spec:
  replicas: 1
  selector:
    matchLabels:
      app: evs-statefulset-example
  template:
    metadata:
      labels:
        app: evs-statefulset-example
    spec:
      containers:
        - name: container-0
          image: 'nginx:latest'
          volumeMounts:
            - name: pvc-evs-auto-example
              mountPath: /tmp
          restartPolicy: Always
          imagePullSecrets:
            - name: default-secret
      volumeClaimTemplates:
        - metadata:
            name: pvc-evs-auto-example
            namespace: default
            annotations:
              everest.io/disk-volume-type: SAS
        spec:
          accessModes:
            - ReadWriteOnce
          requests:
            storage: 10Gi
          storageClassName: csi-disk
          serviceName: evs-statefulset-example-headless
          updateStrategy:
            type: RollingUpdate

Table 9-16 Key parameters

<table>
<thead>
<tr>
<th>Parent Parameter</th>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>metadata</td>
<td>name</td>
<td>Name of the created workload.</td>
</tr>
<tr>
<td>spec.template.spec.containers</td>
<td>image</td>
<td>Image of the workload.</td>
</tr>
<tr>
<td>spec.template.spec.containers.volumeMount</td>
<td>mountPath</td>
<td>Mount path of the container. In this example, the volume is mounted to the /tmp directory.</td>
</tr>
<tr>
<td>spec</td>
<td>serviceName</td>
<td>Service corresponding to the workload. For details about how to create a Service, see Creating a StatefulSet.</td>
</tr>
</tbody>
</table>
NOTE

spec.template.spec.containers.volumeMounts.name and spec.volumeClaimTemplates.metadata.name must be consistent because they have a mapping relationship.

Step 3 Run the following command to create a pod:

`kubectl create -f evs-deployment-example.yaml`

After the creation is complete, log in to the CCE console. In the navigation pane, choose Resource Management > Storage > EVS. Then, click the PVC name. On the PVC details page, you can view the binding relationship between the EVS disk and the PVC.

----End

Verifying Persistent Storage of an EVS Volume

Step 1 Query the pod and EVS files of the deployed workload (for example, evs-statefulset-example).

1. Run the following command to query the pod name of the workload:
   `kubectl get po | grep evs-statefulset-example`
   Expected outputs:
   ```
   evs-statefulset-example-0   1/1     Running   0          22h
   ```

2. Run the following command to check whether an EVS volume is mounted to the /tmp directory:
   `kubectl exec evs-statefulset-example-0 -- df tmp`
   Expected outputs:
   ```
   /dev/sda        10255636 36888  10202364   1% /tmp
   ```

Step 2 Run the following command to create a file named test in the /tmp directory:

`kubectl exec evs-statefulset-example-0 -- touch /tmp/test`

Step 3 Run the following command to view the file in the /tmp directory:

`kubectl exec evs-statefulset-example-0 -- ls -l /tmp`

Expected outputs:
```
-rw-r--r-- 1 root root     0 Jun  1 02:50 test
```
3. The test file still exists after the pod is rebuilt, indicating that the data in the EVS volume can be persistently stored.

--- End

9.5 Using SFS Turbo File Systems as Storage Volumes

9.5.1 Overview

CCE allows you to mount a volume created from an SFS Turbo file system to a container to store data persistently. Provisioned on demand and fast, SFS Turbo is suitable for DevOps, container microservices, and enterprise OA scenarios.

Figure 9-8 Mounting SFS Turbo volumes to CCE

Description

- **Standard file protocols**: You can mount file systems as volumes to servers, the same as using local directories.
- **Data sharing**: The same file system can be mounted to multiple servers, so that data can be shared.
- **Private network**: User can access data only in private networks of data centers.
- **Data isolation**: The on-cloud storage service provides exclusive cloud file storage, which delivers data isolation and ensures IOPS performance.
- **Use cases**: Deployments/StatefulSets in the ReadWriteMany mode, DaemonSets, and jobs created for high-traffic websites, log storage, DevOps, and enterprise OA applications.
9.5.2 Using SFS Turbo Volumes

Prerequisites

You have created a CCE cluster and installed the CSI plug-in (everest) in the cluster.

Notes and Constraints

Currently, SFS Turbo file systems cannot be directly created on CCE.

Importing an SFS Turbo Volume

CCE allows you to import existing SFS Turbo volumes. Currently, only HUAWEI CLOUD accounts and IAM users with the CCE Administrator permissions can import SFS Turbo volumes.

Step 1  Log in to the CCE console. In the navigation pane, choose Resource Management > Storage. On the SFS Turbo tab page, click Import.

Step 2  Select one or more SFS Turbo volumes that you want to import.

Step 3  Select the cluster and namespace to which you want to import the volumes.

Step 4  Click Next. The volumes are displayed in the list. When PVS Status becomes Bound, the volumes are imported successfully.

Adding an SFS Turbo Volume

Step 1  Create a workload or job by referring to Creating a Deployment, Creating a StatefulSet, Creating a DaemonSet, or Creating a Job. After you have added a container, choose Data Storage > Cloud Volume, and then click Add Cloud Volume.

Step 2  Set the storage volume type to SFS Turbo.

Table 9-17 Parameters for configuring an SFS Turbo volume

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Parameter Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>SFS Turbo: applicable to DevOps, containerized microservices, and enterprise office applications.</td>
</tr>
<tr>
<td>Allocation Mode</td>
<td>Manual  Select an existing SFS Turbo volume. You need to import SFS Turbo volumes in advance. For details, see Importing an SFS Turbo Volume.</td>
</tr>
</tbody>
</table>
### Parameter Description

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Add Container Path</strong></td>
<td>Configure the following parameters:</td>
</tr>
<tr>
<td></td>
<td>1. <strong>subPath</strong>: Enter the subpath of the file storage, for example, <code>/tmp</code>. This parameter specifies a subpath inside the referenced volume instead of its root. If this parameter is not specified, the root path is used. Currently, only file storage is supported. The value must be a relative path and cannot start with a slash (<code>/</code>) or <code>../</code>.</td>
</tr>
<tr>
<td></td>
<td>2. <strong>Container Path</strong>: Enter the mount path in the container, for example, <code>/tmp</code>. The mount path must not be a system directory, such as <code>/</code> and <code>/var/run</code>. Otherwise, an exception occurs. You are advised to mount the volume to an empty directory. If the directory is not empty, ensure that there are no files affecting container startup in the directory. Otherwise, such files will be replaced, resulting in failures to start the container and create the workload.</td>
</tr>
<tr>
<td></td>
<td><strong>NOTICE</strong>: If the volume is mounted to a high-risk directory, you are advised to use an account with minimum permissions to start the container; otherwise, high-risk files on the host machine may be damaged.</td>
</tr>
<tr>
<td></td>
<td>3. Set permissions.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Read-only</strong>: You can only read the data in the mounted volumes.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Read/Write</strong>: You can modify the data in the mounted volumes. Newly written data is not migrated if the container is migrated, which causes a data loss.</td>
</tr>
</tbody>
</table>

Click **Add Container Path** to add multiple settings. Then, click **OK**.

---

**Step 3**  Click OK.

---End

### Unbinding an SFS Turbo Volume

When an SFS Turbo volume is successfully imported to a cluster, the volume is bound to the cluster. The volume can also be imported to other clusters. When the volume is unbound from the cluster, other clusters can still import and use the volume.

If the SFS Turbo volume has been mounted to a workload, the volume cannot be unbound from the cluster.

**Step 1**  Log in to the CCE console. In the navigation pane, choose **Resource Management > Storage**. In the SFS Turbo volume list, click **Unbind** next to the target volume.
Step 2 In the dialog box displayed, click OK.

---End

9.5.3 (kubectl) Creating a PV from an Existing SFS Turbo File System

Scenario

CCE allows you to use an existing SFS Turbo file system to create a PersistentVolume (PV). After the creation is successful, you can create a PersistentVolumeClaim (PVC) and bind it to the PV.

Prerequisites

You have created a CCE cluster and installed the CSI plug-in (everest) in the cluster.

Notes and Constraints

The following configuration example applies to clusters of Kubernetes 1.15 or later.

Procedure

Step 1 Log in to the SFS console, create a file system, and record the file system ID, shared path, and capacity.

Step 2 Use kubectl to connect to the cluster. For details, see Connecting to a Cluster Using kubectl.

Step 3 Create two YAML files for creating the PV and PVC. Assume that the file names are pv-efs-example.yaml and pvc-efs-example.yaml.

```
touch pv-efs-example.yaml  pvc-efs-example.yaml
```

- vi pv-efs-example.yaml

  Example YAML file for the PV:

  ```yaml
  apiVersion: v1
  kind: PersistentVolume
  metadata:
    name: pv-efs-example
    annotations:
      pv.kubernetes.io/provisioned-by: everest-csi-provisioner
  spec:
    accessModes:
    - ReadWriteMany
    capacity:
      storage: 10Gi
    claimRef:
      apiVersion: v1
      kind: PersistentVolumeClaim
      name: pvc-efs-example
      namespace: default
    csi:
      driver: sfsturbo.csi.everest.io
      fsType: nfs
      volumeAttributes:
  ```
Table 9-18 Key parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>driver</td>
<td>Storage driver used to mount the volume. Set it to sfsturbo.csi.everest.io.</td>
</tr>
<tr>
<td>everest.io/share-export-location</td>
<td>Shared path of the SFS Turbo volume.</td>
</tr>
<tr>
<td>volumeHandle</td>
<td>SFS Turbo volume ID. You can obtain the ID on the SFS Turbo storage instance details page on the SFS console.</td>
</tr>
<tr>
<td>storage</td>
<td>File system size.</td>
</tr>
<tr>
<td>storageClassName</td>
<td>Name of the Kubernetes storage class. Set this field to csi-sfsturbo for SFS Turbo volumes.</td>
</tr>
<tr>
<td>spec.claimRef.apiVersion</td>
<td>The value is fixed at v1.</td>
</tr>
<tr>
<td>spec.claimRef.kind</td>
<td>The value is fixed at PersistentVolumeClaim.</td>
</tr>
<tr>
<td>spec.claimRef.name</td>
<td>The value is the same as the name of the PVC created in the next step.</td>
</tr>
<tr>
<td>spec.claimRef.namespace</td>
<td>The value is the same as the namespace of the PVC created in the next step.</td>
</tr>
</tbody>
</table>

- vi pvc-efs-example.yaml

Example YAML file for the PVC:

```yaml
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  annotations:
    volume.beta.kubernetes.io/storage-provisioner: everest-csi-provisioner
name: pvc-efs-example
namespace: default
spec:
  accessModes:
    - ReadWriteMany
  resources:
    requests:
      storage: 10Gi
  storageClassName: csi-sfsturbo
  volumeName: pv-efs-example
```

Table 9-19 Key parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>storageClassName</td>
<td>Name of the Kubernetes storage class. Set this field to csi-sfsturbo.</td>
</tr>
</tbody>
</table>
### Parameter Description

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>storage</td>
<td>Storage capacity, in the unit of Gi. The value must be the same as the storage size of the existing PV.</td>
</tr>
<tr>
<td>volumeName</td>
<td>Name of the PV.</td>
</tr>
</tbody>
</table>

**NOTE**

The VPC to which the SFS Turbo file system belongs must be the same as the VPC of the ECS VM planned for the workload. Ports 111, 445, 2049, 2051, and 20048 must be enabled in the security groups.

**Step 4** Create the PV.

```
kubectl create -f pv-efs-example.yaml
```

**Step 5** Create the PVC.

```
kubectl create -f pvc-efs-example.yaml
```

--- End

### 9.5.4 (kubectl) Creating a Deployment Mounted with an SFS Turbo Volume

**Scenario**

After an SFS Turbo volume is created or imported to CCE, you can mount the volume to a workload.

**Prerequisites**

You have created a CCE cluster and installed the CSI plug-in (*everest*) in the cluster.

**Notes and Constraints**

The following configuration example applies to clusters of Kubernetes 1.15 or later.

**Procedure**

**Step 1** Use kubectl to connect to the cluster. For details, see [Connecting to a Cluster Using kubectl](#).

**Step 2** Run the following commands to configure the `efs-deployment-example.yaml` file, which is used to create a Deployment:

- `touch efs-deployment-example.yaml`
- `vi efs-deployment-example.yaml`

Example of mounting an SFS Turbo volume to a Deployment (PVC-based, shared volume):
apiVersion: apps/v1
kind: Deployment
metadata:
  name: efs-deployment-example                                # Workload name
  namespace: default
spec:
  replicas: 1
  selector:
    matchLabels:
      app: efs-deployment-example
template:
  metadata:
    labels:
      app: efs-deployment-example
  spec:
    containers:
      - image: nginx
        name: container-0
        volumeMounts:
          - mountPath: /tmp                                # Mount path
            name: pvc-efs-example
        restartPolicy: Always
        imagePullSecrets:
          - name: default-secret
        volumes:
          - name: pvc-efs-example
            persistentVolumeClaim:
              claimName: pvc-sfs-auto-example                # PVC name

Table 9-20 Key parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Name of the created Deployment.</td>
</tr>
<tr>
<td>app</td>
<td>Name of the Deployment.</td>
</tr>
<tr>
<td>mountPath</td>
<td>Mount path of the container. In this example, the mount path is /tmp.</td>
</tr>
</tbody>
</table>

NOTE

spec.template.spec.containers.volumeMounts.name and spec.template.spec.volumes.name must be consistent because they have a mapping relationship.

Step 3  Run the following command to create a pod:

```
kubectl create -f efs-deployment-example.yaml
```

After the creation is complete, choose Storage > SFS Turbo on the CCE console and click the PVC name. On the PVC details page, you can view the binding relationship between SFS Turbo and PVC.

-----End
9.5.5 (kubectl) Creating a StatefulSet Mounted with an SFS Turbo Volume

Scenario

CCE allows you to use an existing SFS Turbo volume to create a StatefulSet.

Prerequisites

You have created a CCE cluster and installed the CSI plug-in (everest) in the cluster.

Notes and Constraints

The following configuration example applies to clusters of Kubernetes 1.15 or later.

Procedure

Step 1 Create an SFS Turbo volume and record the volume name.

Step 2 Use kubectl to connect to the cluster. For details, see Connecting to a Cluster Using kubectl.

Step 3 Create a YAML file for creating the workload. Assume that the file name is efs-statefulset-example.yaml.

`touch efs-statefulset-example.yaml`

`vi efs-statefulset-example.yaml`

Configuration example:

```yaml
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: efs-statefulset-example
  namespace: default
spec:
  replicas: 1
  selector:
    matchLabels:
      app: efs-statefulset-example
  template:
    metadata:
      labels:
        app: efs-statefulset-example
    spec:
      volumes:
        - name: pvc-efs-example
          persistentVolumeClaim:
            claimName: pvc-efs-example
      containers:
        - name: container-0
          image: 'nginx:latest'
          volumeMounts:
            - name: pvc-efs-example
              mountPath: /tmp
          restartPolicy: Always
          imagePullSecrets:
            - name: default-secret
```
Table 9-21 Key parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>replicas</td>
<td>Number of pods.</td>
</tr>
<tr>
<td>name</td>
<td>Name of the new workload.</td>
</tr>
<tr>
<td>image</td>
<td>Image used by the workload.</td>
</tr>
<tr>
<td>mountPath</td>
<td>Mount path of a container.</td>
</tr>
<tr>
<td>serviceName</td>
<td>Service corresponding to the workload. For details about how to create a</td>
</tr>
<tr>
<td></td>
<td>Service, see <a href="#">Creating a StatefulSet</a>.</td>
</tr>
<tr>
<td>claimName</td>
<td>Name of an existing PVC.</td>
</tr>
</tbody>
</table>

**NOTE**

`spec.template.spec.containers.volumeMounts.name` and `spec.template.spec.volumes.name` must be consistent because they have a mapping relationship.

**Step 4** Create the StatefulSet.

```bash
kubectl create -f efs-statefulset-example.yaml
```

--- End

### Verifying Persistent Storage of an SFS Turbo Volume

**Step 1** Query the pod and SFS Turbo volume of the deployed workload (for example, `efs-statefulset-example`).

1. Run the following command to query the pod name of the workload:

   ```bash
   kubectl get po | grep efs-statefulset-example
   ```

   Expected outputs:

   ```
   efs-statefulset-example-0   1/1     Running   0          2m5s
   ```

2. Run the following command to check whether an SFS Turbo volume is mounted to the `/tmp` directory:

   ```bash
   kubectl exec efs-statefulset-example-0 -- mount | grep /tmp
   ```

   Expected outputs:

   ```
   192.168.0.108:/ on /tmp type nfs
   (rw,relatime,vers=3,rsize=1048576,wsize=1048576,namlen=255,hard,nolock,noresvport,proto=tcp,timeo
   =600,retrans=2,sec=sys,mountaddr=192.168.0.108,mountvers=3,mountport=20048,mountproto=tcp,local
   _lock=all,addr=192.168.0.108)
   ```

**Step 2** Run the following command to create a file named `test` in the `/tmp` directory:

```bash
kubectl exec efs-statefulset-example-0 -- touch /tmp/test
```

**Step 3** Run the following command to view the file in the `/tmp` directory:

```bash
kubectl exec efs-statefulset-example-0 -- ls -l /tmp
```
Step 4 Run the following command to delete the pod named `efs-statefulset-example-0`:
```
kubectl delete po efs-statefulset-example-0
```

Step 5 Check whether the file still exists after the pod is rebuilt.

1. Run the following command to query the name of the rebuilt pod:
```
kubectl get po
```

   Expected outputs:
   
   `efs-statefulset-example-0   1/1     Running   0          2m`

2. Run the following command to view the file in the `/tmp` directory:
```
kubectl exec efs-statefulset-example-0 -- ls -l /tmp
```

   Expected outputs:
   
   `-rw-r--r-- 1 root root     0 Jun  1 02:50 test`

   The `test` file still exists after the pod is rebuilt, indicating that the data in the SFS Turbo volume can be persistently stored.

--- End

9.6 Using OBS Buckets as Storage Volumes

9.6.1 Overview

CCE allows you to mount a volume created from an Object Storage Service (OBS) bucket to a container to store data persistently. Object storage is commonly used in cloud workloads, data analysis, content analysis, and hotspot objects.

Figure 9-9 Mounting OBS volumes to CCE
Storage Class

Object storage offers three storage classes, Standard, Infrequent Access, and Archive, to satisfy different requirements for storage performance and costs.

- The Standard storage class features low access latency and high throughput. It is therefore applicable to storing a large number of hot files (frequently accessed every month) or small files (less than 1 MB). The application scenarios include big data analytics, mobile apps, hot videos, and picture processing on social media.

- The Infrequent Access storage class is ideal for storing data that is semi-frequently accessed (less than 12 times a year), with requirements for quick response. The application scenarios include file synchronization or sharing, and enterprise-level backup. It provides the same durability, access latency, and throughput as the Standard storage class but at a lower cost. However, the Infrequent Access storage class has lower availability than the Standard storage class.

- The Archive storage class is suitable for archiving data that is rarely-accessed (averagely once a year). The application scenarios include data archiving and long-term data backup. The Archive storage class is secure and durable at an affordable low cost, which can be used to replace tape libraries. However, it may take hours to restore data from the Archive storage class.

Description

- **Standard APIs**: With HTTP RESTful APIs, OBS allows you to use client tools or third-party tools to access object storage.

- **Data sharing**: Servers, embedded devices, and IoT devices can use the same path to access shared object data in OBS.

- **Public/Private networks**: OBS allows data to be accessed from public networks to meet Internet application requirements.

- **Capacity and performance**: No capacity limit; high performance (read/write I/O latency within 10 ms).

- **Use cases**: Deployments/StatefulSets in the ReadOnlyMany mode and jobs created for big data analysis, static website hosting, online video on demand (VoD), gene sequencing, intelligent video surveillance, backup and archiving, and enterprise cloud boxes (web disks). You can create object storage by using the OBS console, tools, and SDKs.

Reference

CCE clusters can also be mounted with OBS buckets of third-party tenants, including OBS parallel file systems (preferred) and OBS object buckets. For details, see [Mounting an Object Storage Bucket of a Third-Party Tenant](#).

9.6.2 Using OBS Volumes

Prerequisites

You have created a CCE cluster and installed the CSI plug-in (everest) in the cluster.
Notes and Constraints

- CCE clusters of v1.7.3-r8 and earlier do not support OBS volumes. You need to upgrade these clusters or create clusters of a later version that supports OBS.
- Kunpeng clusters do not support obsfs. Therefore, parallel file systems cannot be mounted.
- Volumes cannot be created in specified enterprise projects. Only the default enterprise project is supported.

Preparations

To mount reliable and stable OBS buckets as volumes, you must create AK/SK before you create OBS buckets.

The procedure for configuring the AK/SK is as follows:

1. Log in to the CCE console. In the navigation pane, choose Resource Management > Storage.
2. On the OBS tab page, click AK/SK in the notice.

Figure 9-10 Configuring the AK/SK

3. Click ☰, select a key file, and click Upload to upload the key file.
4. Select the corresponding workload and click Restart.

**NOTICE**

When creating an OBS volume, you must use the AK/SK. If the key file is not uploaded, the pod will fail to be started or OBS data access will be abnormal due to the volume mounting failure.

- For details about how to obtain access keys, see Obtaining Access Keys (AK and SK).
- For details about how to manage access keys, see Access Keys.

Creating an OBS Volume

**Step 1** Log in to the CCE console. In the navigation pane, choose Resource Management > Storage.

**Step 2** Click the OBS tab and click Create OBS Bucket.

**Step 3** Configure basic information, as shown in Table 9-22.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Parameter Description</th>
</tr>
</thead>
</table>
| * PVC Name        | Name of the new PVC, which is different from the volume name. The actual volume name is automatically generated when the PV is created by the PVC.  
                   | The name contains 3 to 55 characters (excluding the prefix). It must contain lowercase letters, digits, and hyphens (-), and cannot start or end with a hyphen (-).                                                                 |
| Cluster Name      | Cluster to which the OBS volume belongs.                                                                                                                                                                               |
| Namespace         | Namespace to which the volume belongs. The default value is default.                                                                                                                                                 |
| Instance Type     | Type of the storage instance created on OBS.  
                   | - **Parallel file system:** If the cluster version is v1.15 or later and the everest add-on version is 1.0.2 or later, parallel file systems that can be mounted by obsfs can be created.  
                   | - **Object bucket:** A bucket is a container for storing objects in OBS. OBS provides flat storage in the form of buckets and objects. Unlike the conventional multi-layer directory structure of file systems, all objects in a bucket are stored at the same logical layer.  
                   | **NOTE**  
                   | Parallel file systems are optimized OBS objects. You are advised to use a parallel file system instead of an object bucket to mount OBS volumes to containers.                                                                 |
| Storage Class     | This parameter is displayed when you select **Object bucket** for **Instance Type**.  
                   | This parameter indicates the storage classes supported by OBS.  
                   | - **Standard:** applicable to scenarios where a large number of hotspot files or small-sized files need to be accessed frequently (multiple times per month on average) and require fast access response.  
                   | - **Infrequent access:** applicable to scenarios where data is not frequently accessed (less than 12 times per year on average) but requires fast access response.  
                   | **NOTICE**  
                   | Retrieval of data from Infrequent Access OBS is billed separately. Click [here](#) to view OBS pricing details.                                                                 |
## Parameter Description

### Access Mode

Access permissions of user applications on storage resources (PVs).

- **ReadWriteOnce (RWO)**: The volume is mounted as read-write by a single node.
- **ReadWriteMany (RWX)**: The volume is mounted as read-write by multiple nodes.

### Storage Format

The default type is **CSI**.

The container storage interface (CSI) is used to establish a set of standard storage management interfaces between Kubernetes and external storage systems to provide storage services for containers.

### Step 4

Click **Create**.

After the OBS volume is successfully created, it is displayed in the OBS volume list. Click the PVC name to view detailed information about the OBS volume.

--- End

## Adding an OBS Volume

### Step 1

Create a workload or job by referring to **Creating a Deployment**, **Creating a StatefulSet**, **Creating a DaemonSet**, or **Creating a Job**. After you have added a container, choose **Data Storage > Cloud Volume**, and then click **Add Cloud Volume**.

### Step 2

Set **Type** to **OBS**.

### Table 9-23 OBS volume parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Type</strong></td>
<td>Select <strong>OBS</strong>. <strong>OBS</strong>: Standard and Infrequent Access OBS buckets are supported. OBS buckets are commonly used for big data analytics, cloud native applications, static website hosting, and backup/active archiving.</td>
</tr>
<tr>
<td><strong>Allocation Mode</strong></td>
<td></td>
</tr>
<tr>
<td>Manual</td>
<td><strong>Name</strong>: Select a created OBS volume. <strong>Sub-Type</strong>: class of the selected volume. The value can be <strong>Standard</strong> or <strong>Infrequent access</strong>, and you do not need to set this parameter.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>----------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Automatic</td>
<td>Type of the storage instance created on OBS.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Parallel file system</strong>: If the cluster version is v1.15 or later and the everest add-on version is 1.0.2 or later, parallel file systems that can be mounted by obsfs can be created.</td>
</tr>
<tr>
<td></td>
<td><strong>Storage Format</strong>: The default value is CSI.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Object bucket</strong>: A bucket is a container for storing objects in OBS.</td>
</tr>
<tr>
<td></td>
<td><strong>Sub-Type</strong>: Select <strong>Standard</strong> or <strong>Infrequent access</strong>.</td>
</tr>
<tr>
<td></td>
<td><strong>Storage Format</strong>: The default value is CSI.</td>
</tr>
<tr>
<td></td>
<td><strong>NOTE</strong></td>
</tr>
<tr>
<td></td>
<td>Parallel file systems are optimized OBS objects. You are advised to use <strong>parallel file systems</strong> instead of object buckets to mount OBS volumes to containers.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Add Container Path</th>
<th>Configure the following parameters:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1. <strong>Container Path</strong>: Enter the mount path in the container, for example, /tmp.</td>
</tr>
<tr>
<td></td>
<td>The mount path must not be a system directory, such as / and /var/run. Otherwise, an exception occurs. You are advised to mount the volume to an empty directory.</td>
</tr>
<tr>
<td></td>
<td>If the directory is not empty, ensure that there are no files affecting container startup in the directory. Otherwise, such files will be replaced, resulting in failures to start the container and create the workload.</td>
</tr>
<tr>
<td></td>
<td><strong>NOTICE</strong></td>
</tr>
<tr>
<td></td>
<td>If the volume is mounted to a high-risk directory, you are advised to use an account with minimum permissions to start the container; otherwise, high-risk files on the host machine may be damaged.</td>
</tr>
<tr>
<td></td>
<td>2. Set permissions.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Read-only</strong>: You can only read the data in the mounted volumes.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Read/Write</strong>: You can modify the data in the mounted volumes. Newly written data is not migrated if the container is migrated, which causes a data loss.</td>
</tr>
</tbody>
</table>

Click **Add Container Path** to add multiple settings. Then, click **OK**.

### Step 3
Click **OK**.

---

**Importing an OBS Volume**

CCE allows you to import existing OBS volumes.

**Step 1** Log in to the CCE console. In the navigation pane, choose **Resource Management** > **Storage**. On the **OBS** tab page, click **Import**.

**Step 2** Select one or more OBS volumes that you want to import.
**NOTE**

Parallel file systems are optimized OBS objects. You are advised to use parallel file systems instead of object buckets to mount OBS volumes to containers.

**Step 3** Select the target cluster and namespace.

**Step 4** Click OK.

---End

### Unbinding an OBS Volume

When an OBS volume is successfully created, the OBS volume is automatically bound to the current cluster. Other clusters can also use the OBS volume. When the volume is unbound from the cluster, other clusters can still use the volume.

If the volume has been mounted to a workload, the volume cannot be unbound from the cluster.

**Step 1** Log in to the CCE console. In the navigation pane, choose Resource Management > Storage. In the OBS volume list, click Unbind next to the target OBS volume.

**Step 2** In the dialog box displayed, click Yes.

---End

### Related Operations

After an OBS volume is created, you can perform the operation described in Table 9-24.

**Table 9-24** Other Operations

<table>
<thead>
<tr>
<th>Operation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deleting an OBS volume</td>
<td>1. Select the OBS volume to be deleted and click <strong>Delete</strong> in the <strong>Operation</strong> column.</td>
</tr>
<tr>
<td></td>
<td>2. Follow the prompts to delete the volume.</td>
</tr>
</tbody>
</table>

### 9.6.3 (kubectl) Automatically Creating an OBS Volume

**Scenario**

During the use of OBS, expected OBS buckets can be automatically created and mounted as volumes. Currently, standard and infrequent access OBS buckets are supported, which correspond to **obs-standard** and **obs-standard-ia**, respectively.

**Prerequisites**

You have created a CCE cluster and installed the CSI plug-in (**everest**) in the cluster.
Notes and Constraints

The following configuration example applies to clusters of Kubernetes 1.15 or later.

Procedure

**Step 1** Use kubectl to connect to the cluster. For details, see Connecting to a Cluster Using kubectl.

**Step 2** Run the following commands to configure the pvc-obs-auto-example.yaml file, which is used to create a PVC.

```bash
touch pvc-obs-auto-example.yaml
diff pvc-obs-auto-example.yaml
```

Example YAML file:

```yaml
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  annotations:
    everest.io/obs-volume-type: STANDARD
csi.storage.k8s.io/fstype: obsfs
name: obs-warm-provision-pvc
namespace: default
spec:
  accessModes:
  - ReadWriteMany
  resources:
    requests:
      storage: 1Gi
storageClassName: csi-obs
```

Table 9-25 Key parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>everest.io/obs-volume-type</td>
<td>OBS bucket type. Currently, standard (STANDARD) and infrequent access (WARM) buckets are supported.</td>
</tr>
<tr>
<td>name</td>
<td>Name of the PVC to be created.</td>
</tr>
<tr>
<td>accessModes</td>
<td>Only <strong>ReadWriteMany</strong> is supported. <strong>ReadWriteOnly</strong> is not supported.</td>
</tr>
<tr>
<td>storage</td>
<td>Storage capacity in the unit of Gi. For OBS buckets, this field is used only for verification (cannot be empty or 0). Its value is fixed at 1, and any value you set does not take effect for OBS buckets.</td>
</tr>
<tr>
<td>csi.storage.k8s.io/fstype</td>
<td>File type. The value can be <strong>obsfs</strong> or <strong>s3fs</strong>. If the value is <strong>s3fs</strong>, an OBS bucket is created and mounted using s3fs. If the value is <strong>obsfs</strong>, an OBS parallel file system is created and mounted using obsfs. You are advised to set this field to <strong>obsfs</strong>.</td>
</tr>
</tbody>
</table>
Step 3 Run the following command to create the PVC.

```
kubectl create -f pvc-obs-auto-example.yaml
```

After the command is executed, an OBS bucket is created in the VPC to which the cluster belongs. You can click the bucket name in Storage > OBS to view the bucket or view it on the OBS console.

---End

9.6.4 (kubectl) Creating a PV from an Existing OBS Bucket

**Scenario**

CCE allows you to use an existing OBS bucket to create a PersistentVolume (PV). You can create a PersistentVolumeClaim (PVC) and bind it to the PV.

**Prerequisites**

You have created a CCE cluster and installed the CSI plug-in (everest) in the cluster.

**Notes and Constraints**

The following configuration example applies to clusters of Kubernetes 1.15 or later.

**Procedure**

1. **Step 1** Log in to the OBS console, create an OBS bucket, and record the bucket name and storage class.

2. **Step 2** Use kubectl to connect to the cluster. For details, see Connecting to a Cluster Using kubectl.

3. **Step 3** Create two YAML files for creating the PV and PVC. Assume that the file names are `pv-obs-example.yaml` and `pvc-obs-example.yaml`.

```
touch pv-obs-example.yaml pvc-obs-example.yaml
```

- ```
  vi pv-obs-example.yaml
  ```

  **Example YAML file for the PV:**
  ```yaml
  apiVersion: v1
  kind: PersistentVolume
  metadata:
    name: pv-obs-example
  annotations:
    pv.kubernetes.io/provisioned-by: everest-csi-provisioner
  spec:
    mountOptions:
      - umask=0027
      - uid=10000,gid=10000
    accessModes:
      - ReadWriteMany
    capacity:
      storage: 1Gi
  claimRef:
    apiVersion: v1
    kind: PersistentVolumeClaim
  ```
name: pvc-obs-example
namespace: default
csi:
driver: obs.csi.everest.io
fsType: obsfs
volumeAttributes:
everest.io/obs-volume-type: STANDARD
everest.io/region: ap-southeast-1
storage.kubernetes.io/csiProvisionerIdentity: everest-csi-provisioner
volumeHandle: obs-normal-static-pv
persistentVolumeReclaimPolicy: Delete
storageClassName: csi-obs

Table 9-26 Key parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>driver</td>
<td>Storage driver used to mount the volume. Set the driver to <code>obs.csi.everest.io</code> for the OBS volume.</td>
</tr>
<tr>
<td>everest.io/obs-volume-type</td>
<td>Storage class, including <strong>STANDARD</strong> (standard bucket) and <strong>WARM</strong> (infrequent access bucket).</td>
</tr>
<tr>
<td>everest.io/region</td>
<td>Region where the OBS bucket is deployed. For details about the value of <code>region</code>, see <a href="#">Regions and Endpoints</a>.</td>
</tr>
<tr>
<td>volumeHandle</td>
<td>OBS bucket name.</td>
</tr>
<tr>
<td>storage</td>
<td>Storage capacity in the unit of Gi. The value is fixed at 1Gi.</td>
</tr>
<tr>
<td>storageClassName</td>
<td>Name of the Kubernetes storage class. Set this field to <strong>csi-obs</strong>.</td>
</tr>
<tr>
<td>fsType</td>
<td>File type. The value can be <strong>obsfs</strong> or <strong>s3fs</strong>. If the value is <strong>s3fs</strong>, an OBS bucket is created and mounted using s3fs. If the value is <strong>obsfs</strong>, an OBS parallel file system is created and mounted using obsfs. You are advised to set this field to <strong>obsfs</strong>.</td>
</tr>
<tr>
<td>spec.mountOptionss</td>
<td>Mount options. For details, see <a href="#">OBS Volume Mount Options</a>.</td>
</tr>
<tr>
<td>spec.claimRef.apiVersion</td>
<td>The value is fixed at <strong>v1</strong>.</td>
</tr>
<tr>
<td>spec.claimRef.kind</td>
<td>The value is fixed at <strong>PersistentVolumeClaim</strong>.</td>
</tr>
<tr>
<td>spec.claimRef.name</td>
<td>PVC name. The value is the same as the name of the PVC created in the next step.</td>
</tr>
<tr>
<td>spec.claimRef.namespace</td>
<td>Namespace of the PVC. The value is the same as the namespace of the PVC created in the next step.</td>
</tr>
</tbody>
</table>

- **vi pvc-obs-example.yaml**

**Example YAML file for the PVC:**

```yaml
apiVersion: v1
kind: PersistentVolumeClaim
```
Table 9-27 Key parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>volumeName</td>
<td>Name of the PV.</td>
</tr>
<tr>
<td>storage</td>
<td>Storage capacity in the unit of Gi. The value is fixed at 1Gi, which must be the same as the storage size of the existing PV.</td>
</tr>
<tr>
<td>storageClassName</td>
<td>Name of the Kubernetes storage class. Set this field to csi-obs.</td>
</tr>
<tr>
<td>everest.io/obs-volume-type</td>
<td>OBS volume type, which can be STANDARD (standard bucket) and WARM (infrequent access bucket).</td>
</tr>
<tr>
<td>csi.storage.k8s.io/fstype</td>
<td>File type, which can be obsfs or s3fs. If the value is s3fs, an OBS bucket is created and mounted using s3fs. If the value is obsfs, an OBS parallel file system is created and mounted using obsfs.</td>
</tr>
</tbody>
</table>

Step 4  Create the PV.

```bash
kubectl create -f pv-obs-example.yaml
```

Step 5  Create the PVC.

```bash
kubectl create -f pvc-obs-example.yaml
```

End

9.6.5 (kubectl) Creating a Deployment Mounted with an OBS Volume

Scenario

After an OBS volume is created or imported to CCE, you can mount the volume to a workload.
Prerequisites

You have created a CCE cluster and installed the CSI plug-in (everest) in the cluster.

Notes and Constraints

The following configuration example applies to clusters of Kubernetes 1.15 or later.

Procedure

**Step 1** Use kubectl to connect to the cluster. For details, see [Connecting to a Cluster Using kubectl](#).

**Step 2** Run the following commands to configure the `obs-deployment-example.yaml` file, which is used to create a pod.

```bash
touch obs-deployment-example.yaml
vi obs-deployment-example.yaml
```

Example of mounting an OBS volume to a Deployment (PVC-based, shared volume):

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: obs-deployment-example # Workload name
  namespace: default
spec:
  replicas: 1
  selector:
    matchLabels:
      app: obs-deployment-example
  template:
    metadata:
      labels:
        app: obs-deployment-example
    spec:
      containers:
        - image: nginx
          name: container-0
          volumeMounts:
            - mountPath: /tmp # Mount path
              name: pvc-obs-example
          restartPolicy: Always
          imagePullSecrets:
            - name: default-secret
          volumes:
            - name: pvc-obs-example
              persistentVolumeClaim:
                claimName: pvc-obs-auto-example # PVC name
```

**NOTE**

The `spec.template.spec.containers.volumeMounts.name` and `spec.template.spec.volumes.name` must be consistent because they have a mapping relationship.

**Step 3** Run the following command to create a pod:

```bash
kubectl create -f obs-deployment-example.yaml
```
After the creation is complete, choose **Storage > OBS** on the CCE console and click the PVC name. On the PVC details page, you can view the binding relationship between the OBS service and the PVC.

---End

### 9.6.6 (kubectl) Creating a StatefulSet Mounted with an OBS Volume

#### Scenario

CCE allows you to use an existing OBS volume to create a StatefulSet through a PVC.

#### Prerequisites

You have created a CCE cluster and installed the CSI plug-in (**everest**) in the cluster.

#### Notes and Constraints

The following configuration example applies to clusters of Kubernetes 1.15 or later.

#### Procedure

1. **Step 1** Create an OBS volume by referring to **Creating an OBS Volume** and obtain the PVC name.

2. **Step 2** Use **kubectl** to connect to the cluster. For details, see **Connecting to a Cluster Using kubectl**.

3. **Step 3** Create a YAML file for creating the workload. Assume that the file name is **obs-statefulset-example.yaml**.

    ```bash
touch obs-statefulset-example.yaml

vi obs-statefulset-example.yaml
```

    **Configuration example:**

    ```yaml
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: obs-statefulset-example
  namespace: default
spec:
  replicas: 1
  selector:
    matchLabels:
      app: obs-statefulset-example
  template:
    metadata:
      labels:
        app: obs-statefulset-example
    spec:
      volumes:
      - name: pvc-obs-example
        persistentVolumeClaim:
Table 9-28 Key parameters

| Parameter    | Description                                                                
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>replicas</td>
<td>Number of pods.</td>
</tr>
<tr>
<td>name</td>
<td>Name of the new workload.</td>
</tr>
<tr>
<td>image</td>
<td>Image used by the workload.</td>
</tr>
<tr>
<td>mountPath</td>
<td>Mount path of a container.</td>
</tr>
<tr>
<td>serviceName</td>
<td>Service corresponding to the workload. For details about how to create a Service, see Creating a StatefulSet.</td>
</tr>
<tr>
<td>claimName</td>
<td>Name of an existing PVC.</td>
</tr>
</tbody>
</table>

Example of mounting an OBS volume to a StatefulSet (PVC template-based, dedicated volume):

Example YAML:

```yaml
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: obs-statefulset-example
  namespace: default
spec:
  replicas: 1
  selector:
    matchLabels:
      app: obs-statefulset-example
  template:
    metadata:
      labels:
        app: obs-statefulset-example
    spec:
      containers:
        - name: container-0
          image: 'nginx:latest'
          volumeMounts:
            - name: pvc-obs-auto-example
              mountPath: /tmp
          restartPolicy: Always
          imagePullSecrets:
            - name: default-secret
        - name: pvc-obs-auto-example
          volumeMounts:
            - name: pvc-obs-auto-example
              mountPath: /tmp
          restartPolicy: Always
          imagePullSecrets:
            - name: default-secret
  volumeClaimTemplates:
    - metadata:
        name: pvc-obs-auto-example
        namespace: default
        annotations:
          everest.io/obs-volume-type: STANDARD
```
Step 4 Create a StatefulSet.

```
kubectl create -f obs-statefulset-example.yaml
```

----End

### Verifying Persistent Storage of an OBS Volume

#### Step 1
Query the pod and OBS volume of the deployed workload (for example, `obs-statefulset-example`).

1. Run the following command to query the pod name of the workload:

   ```bash
   kubectl get po | grep obs-statefulset-example
   ```

   **Expected outputs:**
   
   ```
   obs-statefulset-example-0   1/1     Running   0          2m5s
   ```

2. Run the following command to check whether an OBS volume is mounted to the `/tmp` directory:

   ```bash
   kubectl exec obs-statefulset-example-0 -- mount|grep /tmp
   ```

   **Expected outputs:**
   
   ```
   s3fs on /tmp type fuse.s3fs (rw,nosuid,nodev,relatime,user_id=0,group_id=0,allow_other)
   ```

#### Step 2
Run the following command to create a file named `test` in the `/tmp` directory:

```bash
kubectl exec obs-statefulset-example-0 -- touch /tmp/test
```

#### Step 3
Run the following command to view the file in the `/tmp` directory:

```bash
kubectl exec obs-statefulset-example-0 -- ls -l /tmp
```

**Expected outputs:**

```
-rw-r--r-- 1 root root     0 Jun  1 02:50 test
```

#### Step 4
Run the following command to delete the pod named `obs-statefulset-example-0`:

```bash
kubectl delete po obs-statefulset-example-0
```

#### Step 5
Check whether the file still exists after the pod is rebuilt.

1. Run the following command to query the name of the rebuilt pod:

   ```bash
   kubectl get po
   ```

   **Expected outputs:**
   
   ```
   obs-statefulset-example-0   1/1     Running   0          2m
   ```

2. Run the following command to view the file in the `/tmp` directory:

   ```bash
   kubectl exec obs-statefulset-example-0 -- ls -l /tmp
   ```

   **Expected outputs:**
   
   ```
   -rw-r--r-- 1 root root     0 Jun  1 02:50 test
   ```

3. The `test` file still exists after the pod is rebuilt, indicating that the data in the OBS volume can be persistently stored.

----End
9.6.7 Using a Custom AK/SK to Mount an OBS Volume

Scenario

Before using an OBS volume, you need to upload an access key (AK/SK) to the OBS console. For details, see Preparations.

The key you uploaded is used by default when mounting an OBS volume. That is, all IAM users under your account will use the same key to mount OBS buckets, and they have the same permissions on buckets. This setting does not allow you to configure differentiated permissions for different IAM users.

You can solve this issue by using Everest 1.2.8 and later versions to use custom access keys for different IAM users. For details, see How Can I Control Access Permissions for OBS.

Prerequisites

- The everest add-on must be version 1.2.8 or later.
- The cluster version must be 1.15.11 or later.

Notes and Constraints

Custom access keys cannot be configured for secure containers.

Disabling Auto Key Mounting

To prevent IAM users from performing unauthorized operations, you are advised to disable the automatic mounting of access keys by enabling the disable_auto_mount_secret parameter in the everest add-on. In this way, the access keys uploaded on the console will not be used when creating OBS volumes.

**NOTE**

- When enabling disable-auto-mount-secret, ensure that no OBS volume exists in the cluster. A workload mounted with an OBS volume, when scaled or restarted, will fail to remount the OBS volume because it needs to specify the access key but is prohibited by disable-auto-mount-secret.
- If disable-auto-mount-secret is set to true, an access key must be specified when a PV or PVC is created. Otherwise, the OBS volume fails to be mounted.

**kubectl edit ds everest-csi-driver -nkube-system**

Search for disable-auto-mount-secret and set it to true.
Run `:wq` to save the settings and exit. Wait until the pod is restarted.

### Creating a Secret Using an Access Key

**Step 1**  Obtain an access key.

For details, see [Obtaining Access Keys (AK and SK)](#).

**Step 2**  Encode the keys using Base64. (Assume that the AK is `xxx` and the SK is `yyy`.)

```
echo -n xxx|base64
```

```
echo -n yyy|base64
```

Record the encoded AK and SK.

**Step 3**  Create a YAML file for the secret, for example, `test-user.yaml`.

```yaml
apiVersion: v1
data:
  access.key: WE5WWVhVNUMyTjNGTkFXSkExQUE=
  secret.key: Nnk4emJyZ0FiUnlrcHBvMnJ2b2kxSTk0ekRnd1pYOVh3akxnQVJxUw==
kind: Secret
metadata:
  name: test-user
  namespace: default
  type: cfe/secure-opaque
```

Specifically:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>access.key</td>
<td>Base64-encoded AK.</td>
</tr>
<tr>
<td>secret.key</td>
<td>Base64-encoded SK.</td>
</tr>
<tr>
<td>name</td>
<td>Secret name.</td>
</tr>
<tr>
<td>namespace</td>
<td>Namespace of the secret.</td>
</tr>
<tr>
<td>type</td>
<td>Secret type. The value must be <code>cfe/secure-opaque</code>. When this type is used, the data entered by users is automatically encrypted.</td>
</tr>
</tbody>
</table>

**Step 4**  Create the secret.

```
kubectl create -f test-user.yaml
```

----End

### Mounting a Secret When Statically Creating an OBS Volume

After a secret is created using the AK/SK, you can associate the secret with the PV to be created and then use the AK/SK in the secret to mount an OBS volume. An example is provided as follows:

**Step 1**  Create a YAML file for the PV, for example, `pv-example.yaml`.

The following uses a parallel file system as an example to describe how to create a PV.
apiVersion: v1
kind: PersistentVolume
metadata:
  name: pv-obs-example
  annotations:
    pv.kubernetes.io/provisioned-by: everest-csi-provisioner
spec:
  accessModes:
  - ReadWriteMany
  capacity:
    storage: 1Gi
  csi:
    nodePublishSecretRef:
      name: test-user
      namespace: default
    driver: obs.csi.everest.io
    fsType: obsfs
    volumeAttributes:
      everest.io/obs-volume-type: STANDARD
      everest.io/region: ap-southeast-1
      storage.kubernetes.io/csiProvisionerIdentity: everest-csi-provisioner
      volumeHandle: obs-normal-static-pv
  persistentVolumeReclaimPolicy: Delete

---

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>nodePublishSecre-tRef</td>
<td>Secret specified during the mounting.</td>
</tr>
<tr>
<td></td>
<td>• name: name of the secret</td>
</tr>
<tr>
<td></td>
<td>• namespace: namespace of the secret</td>
</tr>
</tbody>
</table>

**Step 2** Create the PV.

```bash
kubectl create -f pv-example.yaml
```

After the PV is created, you can create a PVC (the secret does not need to be specified for the PVC) and bind it to the PV. Then, create a workload and associate it with the PVC.

----End

### Mounting a Secret When Dynamically Creating an OBS Volume

When dynamically creating an OBS volume, you can use the following method to specify a secret:

**Step 1** Create a YAML file for the PVC, for example, `pvc-example.yaml`.

```yaml
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  annotations:
    csi.storage.k8s.io/node-publish-secret-name: test-user
    csi.storage.k8s.io/node-publish-secret-namespace: default
    everest.io/obs-volume-type: STANDARD
    csi.storage.k8s.io/fsType: obsfs
  name: obs-secret
  namespace: default
spec:
  accessModes:
  - ReadWriteMany
  resources:
    requests:
      storage: 1Gi
  storageClassName: csi-obs
```
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>csi.storage.k8s.io/node-publish-secret-name</td>
<td>Name of the secret</td>
</tr>
<tr>
<td>csi.storage.k8s.io/node-publish-secret-namespace</td>
<td>Namespace of the secret</td>
</tr>
</tbody>
</table>

**Step 2** Create the PVC.

```bash
kubectl create -f pvc-example.yaml
```

After the PVC is created, you can create a workload and associate it with the PVC to create volumes.

----End

**Verification**

You can use a secret of an IAM user to mount an OBS volume. Assume that a workload named `obs-secret` is created, the mount path in the container is `/temp`, and the IAM user has the CCE `ReadOnlyAccess` and `Tenant Guest` permissions.

1. Query the name of the workload pod.

   ```bash
   kubectl get po | grep obs-secret
   ```

   **Expected outputs:**

   ```
   obs-secret-5cd558f76f-vxslv  1/1     Running   0          3m22s
   ```

2. Query the objects in the mount path. In this example, the query is successful.

   ```bash
   kubectl exec obs-secret-5cd558f76f-vxslv -- ls -l /temp/
   ```

3. Write data into the mount path. In this example, the write operation fails.

   ```bash
   kubectl exec obs-secret-5cd558f76f-vxslv -- touch /temp/test
   ```

   **Expected outputs:**

   ```
   touch: setting times of '/temp/test': No such file or directory
   command terminated with exit code 1
   ```

4. Set the read/write permissions for the IAM user who mounted the OBS volume by referring to the bucket policy configuration.

5. Write data into the mount path again. In this example, the write operation succeeded.

   ```bash
   kubectl exec obs-secret-5cd558f76f-vxslv -- touch /temp/test
   ```

6. Check the mount path in the container to see whether the data is successfully written.

   ```bash
   kubectl exec obs-secret-5cd558f76f-vxslv -- ls -l /temp/
   ```

   **Expected outputs:**
9.7 Using SFS File Systems as Storage Volumes

9.7.1 Overview

CCE allows you to mount a volume created from a Scalable File Service (SFS) file system to a container to store data persistently. SFS volumes are commonly used in ReadWriteMany scenarios, such as media processing, content management, big data analysis, and workload process analysis.

Figure 9-11 Mounting SFS volumes to CCE

Description

- **Standard file protocols**: You can mount file systems as volumes to servers, the same as using local directories.
- **Data sharing**: The same file system can be mounted to multiple servers, so that data can be shared.
- **Private network**: User can access data only in private networks of data centers.
- **Capacity and performance**: The capacity of a single file system is high (PB level) and the performance is excellent (ms-level read/write I/O latency).
- **Use cases**: Deployments/StatefulSets in the ReadWriteMany mode and jobs created for high-performance computing (HPC), media processing, content management, web services, big data analysis, and workload process analysis.

For details, see **SFS Service Overview**.
9.7.2 Using SFS Volumes

Prerequisites
You have created a CCE cluster and installed the CSI plug-in (everest) in the cluster.

Notes and Constraints
- Container storage in CCE clusters of Kubernetes 1.13 or later version supports encryption. Currently, E2E encryption is supported only in certain regions.
- Volumes cannot be created in specified enterprise projects. Only the default enterprise project is supported.

Creating an SFS Volume

Step 1  Log in to the CCE console. In the navigation pane, choose Resource Management > Storage.

Step 2  On the SFS tab, click Create SFS File System.

Step 3  Configure basic information, as shown in Table 9-29.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Parameter Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>* PVC Name</td>
<td>Name of the new PVC, which is different from the volume name. The actual volume name is automatically generated when the PV is created by the PVC.</td>
</tr>
<tr>
<td>Cluster Name</td>
<td>Cluster to which the file system volume belongs.</td>
</tr>
<tr>
<td>Namespace</td>
<td>Namespace in which the volume is created.</td>
</tr>
<tr>
<td>Total Capacity</td>
<td>The total capacity is the capacity of a single volume. Fees are charged by actual usage.</td>
</tr>
</tbody>
</table>
| Access Mode | Access permissions of user applications on storage resources (PVs).
  - **ReadWriteOnce** (RWO): The SFS volume can be mounted as read-write only by a single node.
  - **ReadWriteMany** (RWX): The SFS volume can be mounted as read-write by multiple nodes. |
| Storage Format | The default value is CSI and cannot be changed. |
**Parameter** | **Parameter Description**
--- | ---
Encryption | **KMS Encryption** is deselected by default. After **KMS Encryption** is selected, **Key Management Service (KMS)**, an easy-to-use and highly secure key service, will be used for **SFS file systems**. If no agency has been created, click **Create Agency** and set the following parameters:
  - **Agency Name**: Agencies can be used to assign permissions to trusted accounts or cloud services for a specific period of time. If no agency is created, click **Create Agency**. The agency name **SFSAccessKMS** indicates that **SFS** is granted the permission to access **KMS**. After **SFS** is authorized successfully, it can obtain **KMS** keys to encrypt and decrypt file systems.
  - **Key Name**: After a key is created, it can be loaded and used in containerized applications.
  - **Key ID**: generated by default.
This function is supported only for clusters of v1.13.10 and later in certain regions.

---

**Step 4**  Click **Create**.

The volume is displayed in the list. When **PVS Status** becomes **Bound**, the volume is created successfully.

**Step 5**  Click the volume name to view detailed information about the volume.

----End

**Adding an SFS Volume**

**Step 1**  Create a workload or job by referring to **Creating a Deployment**, **Creating a StatefulSet**, **Creating a DaemonSet**, or **Creating a Job**. During creation, expand **Data Storage** after adding a container. On the **Cloud Volume** tab page, click **Add Cloud Volume**.

**Step 2**  Set the storage class to **SFS**.

**Table 9-30** Parameters for mounting an SFS volume

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Parameter Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td><strong>File Storage (NFS)</strong>: This type applies to a wide range of scenarios, including media processing, content management, big data, and application analysis.</td>
</tr>
<tr>
<td>Allocation Mode</td>
<td></td>
</tr>
</tbody>
</table>
### Parameter Description

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manual</td>
<td></td>
</tr>
</tbody>
</table>
| ● **Name**: Select a created file system. You need to create a file system in advance. For details about how to create a file system, see [Creating an SFS Volume](#).  
● **Sub-Type**: subtype of the created file storage.  
● **Storage Capacity**: This field is one of the PVC attributes. If the storage capacity has been expanded on the IaaS side, it is normal that the capacity values are inconsistent. The PVC capacity is the same as the storage entity capacity only after end-to-end container storage capacity expansion is supported for CCE clusters of v1.13. |

| Automatic | An SFS volume is created automatically. You need to enter the storage capacity.  
● **Sub-Type**: Select **NFS**.  
● **Storage Capacity**: Specify the total storage capacity, in GB. Ensure that the storage capacity quota is not exceeded; otherwise, creation will fail.  
● **Storage Format**: The default value is **CSI**. The container storage interface (CSI) is used to establish a set of standard storage management interfaces between Kubernetes and external storage systems to provide storage services for containers.  
● After you select **KMS Encryption**, Key Management Service (KMS), an easy-to-use and highly secure service, will be enabled for file systems. This function is supported only for clusters of v1.13.10 and later in certain regions. If no agency has been created, click **Create Agency** and set the following parameters:  
  - **Agency Name**: Agencies can be used to assign permissions to trusted accounts or cloud services for a specific period of time. If no agency is created, click **Create Agency**. The agency name **SFSAccessKMS** indicates that SFS is granted the permission to access KMS. After SFS is authorized successfully, it can obtain KMS keys to encrypt and decrypt file systems.  
  - **Key Name**: After a key is created, it can be loaded and used in containerized applications.  
  - **Key ID**: generated by default. | |
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Parameter Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add Container Path</td>
<td>Configure the following parameters:</td>
</tr>
<tr>
<td></td>
<td>1. <strong>subPath</strong>: Enter the subpath of the file storage, for example, /tmp. If this parameter is not specified, the root path of the data volume is used by default. Currently, only file storage is supported. The value must be a relative path and cannot start with a slash (/) or ../.</td>
</tr>
<tr>
<td></td>
<td>2. <strong>Container Path</strong>: Enter the path of the container, for example, /tmp. The container path must not be a system directory, such as / and /var/run. Otherwise, an exception occurs. You are advised to mount the volume to an empty directory. If the directory is not empty, ensure that there are no files affecting container startup in the directory. Otherwise, such files will be replaced, resulting in failures to start the container and create the workload.</td>
</tr>
<tr>
<td></td>
<td><strong>NOTICE</strong></td>
</tr>
<tr>
<td></td>
<td>If the volume is mounted to a high-risk directory, you are advised to use an account with minimum permissions to start the container; otherwise, high-risk files on the host machine may be damaged.</td>
</tr>
<tr>
<td></td>
<td>3. Set permissions.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Read-only</strong>: You can only read the data volumes mounted to the path.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Read/Write</strong>: You can modify the data volumes mounted to the path. Newly written data is not migrated if the container is migrated, which may cause a data loss.</td>
</tr>
</tbody>
</table>

Click **Add Container Path** to add multiple settings. Then, click **OK**.

**Step 3** Click **OK**.

----End

**Importing an SFS Volume**

CCE allows you to import existing SFS volumes.

**Step 1** Log in to the CCE console. In the navigation pane, choose **Resource Management** > **Storage**. On the **SFS** tab page, click **Import**.

**Step 2** Select one or more SFS volumes that you want to attach.

**Step 3** Select the target cluster and namespace. Then, click **OK**.

----End

**Unbinding an SFS Volume**

When an SFS volume is successfully created or imported, the volume is automatically bound to the current cluster. Other clusters can also use the volume.
When the SFS volume is unbound from the cluster, other clusters can still import and use the volume.

If the SFS volume has been attached to a workload, the volume cannot be unbound from the cluster.

Step 1 Log in to the CCE console. In the navigation pane, choose Resource Management > Storage. In the SFS volume list, click Unbind next to the target volume.

Step 2 Confirm the unbinding, and click OK.

---End

Related Operations

After an SFS volume is created, you can perform the operation described in Table 9-31.

Table 9-31 Other operations

<table>
<thead>
<tr>
<th>Operation</th>
<th>Description</th>
</tr>
</thead>
</table>
| Deleting an SFS volume     | 1. Select the SFS volume to be deleted and click Delete in the Operation column.  
                            | 2. Follow the prompts to delete the EVS disk. |
| Importing an SFS volume    | CCE allows you to import existing SFS volumes.  
                            | 1. On the SFS tab page, click Import.  
                            | 2. Select one or more SFS volumes that you want to attach.  
                            | 3. Select the target cluster and namespace.  
                            | 4. Click Yes. |

9.7.3 (kubectl) Automatically Creating an SFS Volume

Scenario

CCE supports creating SFS volumes through PersistentVolumeClaims (PVCs).

Prerequisites

You have created a CCE cluster and installed the CSI plug-in (everest) in the cluster.

Notes and Constraints

The following configuration example applies to clusters of Kubernetes 1.15 or later.
Procedure

Step 1  Use kubectl to connect to the cluster. For details, see Connecting to a Cluster Using kubectl.

Step 2  Run the following commands to configure the pvc-sfs-auto-example.yaml file, which is used to create a PVC.

```
touch pvc-sfs-auto-example.yaml
vi pvc-sfs-auto-example.yaml
```

Example YAML file:
```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name:  pvc-sfs-auto-example
  namespace: default
spec:
  accessModes:
  - ReadWriteMany
  resources:
    requests:
      storage: 10Gi
  storageClassName: csi-nas
```

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>storageClassName</td>
<td>Name of the Kubernetes storage class. Set this field to csi-nas.</td>
</tr>
<tr>
<td>name</td>
<td>Name of the PVC to be created.</td>
</tr>
<tr>
<td>accessModes</td>
<td>Only ReadWriteMany is supported. ReadWriteOnly is not supported.</td>
</tr>
<tr>
<td>storage</td>
<td>Storage capacity in the unit of Gi.</td>
</tr>
</tbody>
</table>

Step 3  Run the following command to create a PVC.

```
kubectl create -f pvc-sfs-auto-example.yaml
```

After the command is executed, a file system is created in the VPC to which the cluster belongs. Choose Storage > SFS on the CCE console or switch to the SFS console to view the file system.

----End

9.7.4 (kubectl) Creating a PV from an Existing SFS File System

Scenario

CCE allows you to use an existing file system to create a PersistentVolume (PV). After the creation is successful, create the corresponding PersistentVolumeClaim (PVC) and bind it to the PV.
Prerequisites

You have created a CCE cluster and installed the CSI plug-in (everest) in the cluster.

Notes and Constraints

The following configuration example applies to clusters of Kubernetes 1.15 or later.

Procedure

Step 1 Log in to the SFS console, create a file system, and record the file system ID, shared path, and capacity.

Step 2 Use kubectl to connect to the cluster. For details, see Connecting to a Cluster Using kubectl.

Step 3 Create two YAML files for creating the PV and PVC. Assume that the file names are `pv-sfs-example.yaml` and `pvc-sfs-example.yaml`.

```
touch pv-sfs-example.yaml pvc-sfs-example.yaml
```

- vi pv-sfs-example.yaml

<table>
<thead>
<tr>
<th>Example YAML file for the PV:</th>
</tr>
</thead>
<tbody>
<tr>
<td>apiVersion: v1</td>
</tr>
<tr>
<td>kind: PersistentVolume</td>
</tr>
<tr>
<td>metadata:</td>
</tr>
<tr>
<td>name: pv-sfs-example</td>
</tr>
<tr>
<td>annotations:</td>
</tr>
<tr>
<td>pv.kubernetes.io/provisioned-by: everest-csi-provisioner</td>
</tr>
<tr>
<td>spec:</td>
</tr>
<tr>
<td>mountOptions:</td>
</tr>
<tr>
<td>- hard</td>
</tr>
<tr>
<td>- timeo=600</td>
</tr>
<tr>
<td>- nolock</td>
</tr>
<tr>
<td>accessModes:</td>
</tr>
<tr>
<td>- ReadWriteMany</td>
</tr>
<tr>
<td>capacity:</td>
</tr>
<tr>
<td>storage: 10Gi</td>
</tr>
<tr>
<td>claimRef:</td>
</tr>
<tr>
<td>apiVersion: v1</td>
</tr>
<tr>
<td>kind: PersistentVolumeClaim</td>
</tr>
<tr>
<td>name: pvc-sfs-example</td>
</tr>
<tr>
<td>namespace: default</td>
</tr>
<tr>
<td>csi:</td>
</tr>
<tr>
<td>driver: nas.csi.everest.io</td>
</tr>
<tr>
<td>fsType: nfs</td>
</tr>
<tr>
<td>volumeAttributes:</td>
</tr>
<tr>
<td>everest.io/share-export-location: sfs-nas01.cn-north-4.myhuaweicloud.com:/share-436304e8</td>
</tr>
<tr>
<td>storage.kubernetes.io/csiProvisionerIdentity: everest-csi-provisioner</td>
</tr>
<tr>
<td>volumeHandle: 682f00bb-ace0-41d8-9b3e-913c9aa6b695</td>
</tr>
<tr>
<td>persistentVolumeReclaimPolicy: Delete</td>
</tr>
<tr>
<td>storageClassName: csi-nas</td>
</tr>
</tbody>
</table>

Table 9-33 Key parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>driver</td>
<td>Storage driver used to mount the volume. Set the driver to nas.csi.everest.io for the file system.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>everest.io/share-export-location</td>
<td>Shared path of the file system. On the management console, choose <strong>Service List &gt; Storage &gt; Scalable File Service</strong>. You can obtain the shared path of the file system from the <strong>Mount Address</strong> column, as shown in <strong>Figure 9-12</strong>.</td>
</tr>
<tr>
<td>volumeHandle</td>
<td>File system ID. On the management console, choose <strong>Service List &gt; Storage &gt; Scalable File Service</strong>. In the SFS file system list, click the name of the target file system and copy the content following <strong>ID</strong> on the page displayed.</td>
</tr>
<tr>
<td>storage</td>
<td>File system size.</td>
</tr>
<tr>
<td>storageClassName</td>
<td>Name of the Kubernetes storage class. Set this field to <strong>csi-nas</strong>.</td>
</tr>
<tr>
<td>spec.mountOptions</td>
<td>Mount options. If not specified, the following configurations are used by default. For details, see <strong>SFS Volume Mount Options</strong>. mountOptions: - vers=3 - timeo=600 - nolock - hard</td>
</tr>
<tr>
<td>spec.claimRef.apiVersion</td>
<td>The value is fixed at <strong>v1</strong>.</td>
</tr>
<tr>
<td>spec.claimRef.kind</td>
<td>The value is fixed at <strong>PersistentVolumeClaim</strong>.</td>
</tr>
<tr>
<td>spec.claimRef.name</td>
<td>PVC name. The value is the same as the name of the PVC created in the next step.</td>
</tr>
<tr>
<td>spec.claimRef.namespace</td>
<td>Namespace of the PVC. The value is the same as the namespace of the PVC created in the next step.</td>
</tr>
</tbody>
</table>

**vi pvc-sfs-example.yaml**

**Example YAML file for the PVC:**
```yaml
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  annotations:
    volume.beta.kubernetes.io/storage-provisioner: everest-csi-provisioner
name: pvc-sfs-example
namespace: default
spec:
  accessModes:
    - ReadWriteMany
  resources:
    requests:
      storage: 10Gi
```
storageClassName: csi-nas
volumeName: pv-sfs-example

Table 9-34 Key parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>storageClassName</td>
<td>Set this field to csi-nas. The value must be the same as that of the existing PV.</td>
</tr>
<tr>
<td>storage</td>
<td>Storage capacity, in the unit of Gi. The value must be the same as the storage size of the existing PV.</td>
</tr>
<tr>
<td>volumeName</td>
<td>Name of the PV.</td>
</tr>
</tbody>
</table>

Figure 9-12 SFS - file system sharing paths

![SFS - file system sharing paths]

NOTE
The VPC to which the file system belongs must be the same as the VPC of the ECS VM to which the workload is planned.

Step 4 Create a PV.

`kubectl create -f pv-sfs-example.yaml`

Step 5 Create a PVC.

`kubectl create -f pvc-sfs-example.yaml`

----End

9.7.5 (kubectl) Creating a Deployment Mounted with an SFS Volume

Scenario
After an SFS volume is created or imported to CCE, you can mount the volume to a workload.

Prerequisites
You have created a CCE cluster and installed the CSI plug-in (everest) in the cluster.

Notes and Constraints
The following configuration example applies to clusters of Kubernetes 1.15 or later.
Procedure

Step 1  Use kubectl to connect to the cluster. For details, see Connecting to a Cluster Using kubectl.

Step 2  Run the following commands to configure the `sfs-deployment-example.yaml` file, which is used to create a pod.

```
touch sfs-deployment-example.yaml
vi sfs-deployment-example.yaml
```

Example of mounting an SFS volume to a Deployment (PVC-based, shared volume):

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: sfs-deployment-example       # Workload name
  namespace: default
spec:
  replicas: 1
  selector:
    matchLabels:
      app: sfs-deployment-example
  template:
    metadata:
      labels:
        app: sfs-deployment-example
    spec:
      containers:
      - image: nginx
        name: container-0
        volumeMounts:
          - mountPath: /tmp          # Mount path
            name: pvc-sfs-example
        imagePullSecrets:
        - name: default-secret
        restartPolicy: Always
        volumes:
        - name: pvc-sfs-example
          persistentVolumeClaim:
            claimName: pvc-sfs-auto-example # PVC name
```

**NOTE**

`spec.template.spec.containers.volumeMounts.name` and `spec.template.spec.volumes.name` must be consistent because they have a mapping relationship.

Step 3  Run the following command to create a pod:

```
kubectl create -f sfs-deployment-example.yaml
```

After the creation is complete, log in to the CCE console. In the navigation pane, choose Resource Management > Storage > SFS. Then, click the PVC name. On the PVC details page, you can view the binding relationship between SFS and PVC.

----End
9.7.6 (kubectl) Creating a StatefulSet Mounted with an SFS Volume

Scenario

CCE allows you to use an existing SGS volume to create a StatefulSet (by using a PVC).

Prerequisites

You have created a CCE cluster and installed the CSI plug-in (*everest*) in the cluster.

Notes and Constraints

The following configuration example applies to clusters of Kubernetes 1.15 or later.

Procedure

**Step 1** Create an SFS volume by referring to Creating an SFS Volume and record the volume name.

**Step 2** Use kubectl to connect to the cluster. For details, see Connecting to a Cluster Using kubectl.

**Step 3** Create a YAML file for creating the workload. Assume that the file name is `sfs-statefulset-example.yaml`.

```
touch sfs-statefulset-example.yaml

vi sfs-statefulset-example.yaml
```

Configuration example:

```yaml
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: sfs-statefulset-example
  namespace: default
spec:
  replicas: 1
  selector:
    matchLabels:
      app: sfs-statefulset-example
  template:
    metadata:
      labels:
        app: sfs-statefulset-example
    spec:
      volumes:
      - name: pvc-sfs-example
        persistentVolumeClaim:
          claimName: pvc-sfs-example
      containers:
      - name: container-0
        image: 'nginx:latest'
        volumeMounts:
        - name: pvc-sfs-example
          mountPath: /tmp
```
restartPolicy: Always
imagePullSecrets:
  - name: default-secret
serviceName: sfs-statefulset-example-headless
updateStrategy:
  type: RollingUpdate

Table 9-35 Key parameters

<table>
<thead>
<tr>
<th>Parent Parameter</th>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>spec</td>
<td>replicas</td>
<td>Number of pods.</td>
</tr>
<tr>
<td>metadata</td>
<td>name</td>
<td>Name of the new workload.</td>
</tr>
<tr>
<td>spec.template.spec.containers</td>
<td>image</td>
<td>Image used by the workload.</td>
</tr>
<tr>
<td>spec.template.spec.containers.volumeMounts</td>
<td>mountPath</td>
<td>Mount path of a container.</td>
</tr>
<tr>
<td>spec</td>
<td>serviceName</td>
<td>Service corresponding to the workload. For details about how to create a Service, see Creating a StatefulSet.</td>
</tr>
<tr>
<td>spec.template.spec.persistentVolumeClaim</td>
<td>claimName</td>
<td>Name of an existing PVC.</td>
</tr>
</tbody>
</table>

Example of mounting an SFS volume to a StatefulSet (PVC template-based, dedicated volume):

Example YAML file:
```yaml
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: sfs-statefulset-example
  namespace: default
spec:
  replicas: 1
  selector:
    matchLabels:
      app: sfs-statefulset-example
template:
  metadata:
    labels:
      app: sfs-statefulset-example
  spec:
    containers:
    - name: container-0
      image: nginx:latest
      volumeMounts:
      - name: pvc-sfs-auto-example
        mountPath: /tmp
    restartPolicy: Always
    imagePullSecrets:
    - name: default-secret
    volumeClaimTemplates:
    - metadata:
      name: pvc-sfs-auto-example
      namespace: default
    spec:
```

Cloud Container Engine
User Guide
9 Storage (CSI)

Issue 01 (2021-09-08)  Copyright © Huawei Technologies Co., Ltd.  441
accessModes:
- ReadWriteMany
resources:
requests:
storage: 10Gi
storageClassName: csi-nas
serviceName: sfs-statefulset-example-headless
updateStrategy:
type: RollingUpdate

NOTE
spec.template.spec.containers.volumeMounts.name and spec.template.spec.volumes.name must be consistent because they have a mapping relationship.

Step 4 Create a StatefulSet.

kubectl create -f sfs-statefulset-example.yaml

----End

Verifying Persistent Storage of an SFS Volume

Step 1 Query the pod and SFS volume of the deployed workload (for example, sfs-statefulset-example).

1. Run the following command to query the pod name of the workload:
kubectl get po | grep sfs-statefulset-example

   Expected outputs:
sfs-statefulset-example-0 1/1 Running 0 2m5s

2. Run the following command to check whether an SFS volume is mounted to the /tmp directory:
kubectl exec sfs-statefulset-example-0 -- mount|grep /tmp

   Expected outputs:
sfs-nas01.cn-north-4.myhuaweicloud.com:/share-c56b9aa4 on /tmp type nfs (rw,relatime,vers=3,rsize=1048576,wsize=1048576,namlen=255,hard,nolock,noresvport,proto=tcp,timeo =600,retrans=2,sec=sys,mountaddr=10.79.96.32,mountvers=3,mountproto=tcp,local_lock=all,addr=10.79.96.32)

Step 2 Run the following command to create a file named test in the /tmp directory:
kubectl exec sfs-statefulset-example-0 -- touch /tmp/test

Step 3 Run the following command to view the file in the /tmp directory:
kubectl exec sfs-statefulset-example-0 -- ls -l /tmp

   Expected outputs:
-rw-r--r-- 1 root root 0 Jun 1 02:50 test

Step 4 Run the following command to delete the pod named sfs-statefulset-example-0:
kubectl delete po sfs-statefulset-example-0

Step 5 Check whether the data file in the SFS volume still exists after the pod is rebuilt.

1. Run the following command to query the name of the rebuilt pod:
kubectl get po

   Expected outputs:
sfs-statefulset-example-0 1/1 Running 0 2m

2. Run the following command to view the file in the /tmp directory:
kubectl exec sfs-statefulset-example-0 -- ls -l /tmp

Expected outputs:
-rw-r--r-- 1 root root     0 Jun  1 02:50 test

3. The test file still exists after the pod is rebuilt, indicating that the data in the SFS volume can be persistently stored.

---End

## 9.8 Setting Mount Options

### Scenario

You can mount cloud storage volumes to your containers and use these volumes as local directories.

This section describes how to set mount options when mounting SFS and OBS volumes. You can set mount options in a PV and bind the PV to a PVC. Alternatively, set mount options in a StorageClass and use the StorageClass to create a PVC. In this way, PVs can be dynamically created and inherit mount options configured in the StorageClass by default.

### SFS Volume Mount Options

The everest add-on in CCE presets the options described in Table 9-36 for mounting SFS volumes. You can set other mount options if needed. For details, see [Mounting an NFS File System to ECSs (Linux)](https://example.com).

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>vers=3</td>
<td>File system version. Currently, only NFSv3 is supported, Value: 3</td>
</tr>
<tr>
<td>nolock</td>
<td>Whether to lock files on the server using the NLM protocol. If nolock is selected, the lock is valid for applications on one host. For applications on another host, the lock is invalid.</td>
</tr>
<tr>
<td>timeo=600</td>
<td>Waiting time before the NFS client retransmits a request. The unit is 0.1 second. Recommended value: 600</td>
</tr>
<tr>
<td>hard/soft</td>
<td>Mounting mode.</td>
</tr>
<tr>
<td></td>
<td>• hard: If the NFS request times out, the client keeps resending the request until the request is successful.</td>
</tr>
<tr>
<td></td>
<td>• soft: If the NFS request times out, the client returns an error to the invoking program.</td>
</tr>
</tbody>
</table>

The default value is hard.
OBS Volume Mount Options

When mounting file storage, the everest add-on presets the options described in Table 9-37 and Table 9-38 by default. The options in Table 9-37 are mandatory. You can set other mount options if needed. For details, see Mounting a Parallel File System.

Table 9-37 Mandatory mount options configured by default

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>use_ino</td>
<td>If enabled, obsfs allocates the <code>inode</code> number. Enabled by default in read/write mode.</td>
</tr>
<tr>
<td>big_writes</td>
<td>If configured, the maximum size of the cache can be modified.</td>
</tr>
<tr>
<td>nonempty</td>
<td>Allows non-empty mount paths.</td>
</tr>
<tr>
<td>allow_other</td>
<td>Allows other users to access the parallel file system.</td>
</tr>
<tr>
<td>no_check_certificate</td>
<td>Disables server certificate verification.</td>
</tr>
<tr>
<td>enable_noobj_cache</td>
<td>Enables cache entries for objects that do not exist, which can improve performance. Enabled by default in object bucket read/write mode.</td>
</tr>
<tr>
<td>sigv2</td>
<td>Specifies the signature version. Used by default in object buckets.</td>
</tr>
</tbody>
</table>

Table 9-38 Optional mount options configured by default

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>max_write=131072</td>
<td>If specified, obsfs allocates the <code>inode</code> number. Enabled by default in read/write mode.</td>
</tr>
<tr>
<td>ssl_verify_hostname=0</td>
<td>Disables verifying the SSL certificate based on the host name.</td>
</tr>
<tr>
<td>max_background=100</td>
<td>Allows setting the maximum number of waiting requests in the background. Used by default in parallel file systems.</td>
</tr>
<tr>
<td>public_bucket=1</td>
<td>If set to 1, public buckets are mounted anonymously. Enabled by default in object bucket read/write mode.</td>
</tr>
</tbody>
</table>

You can log in to the node to which the pod is scheduled and view all mount options used for mounting the OBS volume in the process details.

- Object bucket: ps -ef | grep s3fs
  root     22142     1  0 Jun03 ?        00:00:00 /usr/bin/s3fs pvc-82fe2cbe-3838-43a2-8af8-f994e402fb9d /mnt/paas/kubernetes/kubelet/pods/0b13ff68-4c8e-4a1c-b15c-724fd4d64389/volumes/kubernetes.io~csi/pvc-82fe2cbe-3838-43a2-8af8-f994e402fb9d/mount -o url=https://{{endpoint}}:443
**Prerequisites**

- The everest add-on version must be **1.2.8 or later**.
- The add-on identifies the mount options and transfers them to the underlying storage resources, which determine whether the specified options are valid.

**Notes and Constraints**

Mount options cannot be configured for secure containers.

**Setting Mount Options in a PV**

You can use the `mountOptions` field to set mount options in a PV. The options you can configure in `mountOptions` are listed in **SFS Volume Mount Options** and **OBS Volume Mount Options**.

```yaml
apiVersion: v1
kind: PersistentVolume
metadata:
  name: pv-obs-example
annotations:
  pv.kubernetes.io/provisioned-by: everest-csi-provisioner
spec:
  mountOptions:
  - umask=0027
  - uid=10000,gid=10000
accessModes:
- ReadWriteMany
capacity:
  storage: 1Gi
claimRef:
apiVersion: v1
kind: PersistentVolumeClaim
name: pvc-obs-example
namespace: default
csi:
  driver: obs.csi.everest.io
  fsType: obsfs
volumeAttributes:
  everest.io/obs-volume-type: STANDARD
  everest.io/region: ap-southeast-1
  storage.kubernetes.io/csiProvisionerIdentity: everest-csi-provisioner
volumeHandle: obs-normal-static-pv
persistentVolumeReclaimPolicy: Delete
storageClassName: csi-obs
```

After a PV is created, you can create a PVC and bind it to the PV, and then mount the PV to the container in the workload.
Setting Mount Options in a StorageClass

You can use the `mountOptions` field to set mount options in a StorageClass. The options you can configure in `mountOptions` are listed in **SFS Volume Mount Options** and **OBS Volume Mount Options**.

```yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: csi-obs-mount-option
mountOptions:
  - umask=0027
  - uid=10000,gid=1000
parameters:
  csi.storage.k8s.io/csi-driver-name: obs.csi.everest.io
csi.storage.k8s.io/fsType: s3fs
everest.io/obs-volume-type: STANDARD
provisioner: everest-csi-provisioner
reclaimPolicy: Delete
volumeBindingMode: Immediate
```

After the StorageClass is configured, you can use it to create a PVC. By default, the dynamically created PVs inherit the mount options set in the StorageClass.

9.9 Snapshots and Backups

CCE works with EVS to support snapshots. A snapshot is a complete copy or image of EVS disk data at a certain point of time, which can be used for data DR.

You can create snapshots to rapidly save the disk data at specified time points. In addition, you can use snapshots to create new disks so that the created disks will contain the snapshot data in the beginning.

**Notes**

- The snapshot function is available only for clusters of v1.15 or later and requires the CSI-based everest add-on.
- The subtype (common I/O, high I/O, or ultra-high I/O), disk mode (SCSI or VBD), data encryption, sharing status, and capacity of an EVS disk created from a snapshot must be the same as those of the disk associated with the snapshot. These attributes cannot be modified after being queried or set.

**Application Scenario**

The snapshot feature helps address your following needs:

- **Routine data backup**
  You can create snapshots for EVS disks regularly and use snapshots to recover your data in case that data loss or data inconsistency occurred due to misoperations, viruses, or attacks.

- **Rapid data restoration**
  You can create a snapshot or multiple snapshots before an OS change, application software upgrade, or a service data migration. If an exception occurs during the upgrade or migration, service data can be rapidly restored to the time point when the snapshot was created.

  For example, a fault occurred on system disk A of ECS A, and therefore ECS A cannot be started. Because system disk A is already faulty, the data on system
disk A cannot be restored by rolling back snapshots. In this case, you can use an existing snapshot of system disk A to create EVS disk B and attach it to ECS B that is running properly. Then, ECS B can read data from system disk A using EVS disk B.

**NOTE**

The snapshot capability provided by CCE is the same as the CSI snapshot function provided by the Kubernetes community. EVS disks can be created only based on snapshots, and snapshots cannot be rolled back to source EVS disks.

- **Rapid deployment of multiple services**

  You can use a snapshot to create multiple EVS disks containing the same initial data, and these disks can be used as data resources for various services, for example, data mining, report query, and development and testing. This method protects the initial data and creates disks rapidly, meeting the diversified service data requirements.

**Operation Overview**

1. You can create a snapshot to rapidly save the disk data at specified time points.
2. You can create an EVS disk from a snapshot so that the EVS disk contains the snapshot data in the beginning. When data is lost, the snapshot can be used to restore the data to point in time when the snapshot was taken.
3. When a snapshot is no longer needed, delete it to release the virtual resources.

**Creating a Snapshot**

You can create EVS snapshots to save the disk data at specific time points.

**NOTE**

Snapshots can be created only for EVS disks that are available or in use. A maximum of seven snapshots can be created for a single EVS disk. For details about how to create an EVS disk, see Using EVS Disks as Storage Volumes.

Snapshots data of encrypted disks is stored encrypted, and that of non-encrypted disks is stored non-encrypted.

**Procedure**

**Step 1** Log in to the CCE console. In the navigation pane, choose Resource Management > Storage.

**Step 2** On the Snapshot and backup tab page, click Create Snapshot.

**Step 3** On the Create Snapshot page displayed, configure parameters listed in Table 9-39.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Snapshot Name</td>
<td>Name of the snapshot to be created. The name starts with <code>cce-disksnap-</code> by default, for example, <code>cce-disksnap-01</code>.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Cluster Name</td>
<td>Cluster where the snapshot is deployed.</td>
</tr>
<tr>
<td>Namespace</td>
<td>Namespace with which the snapshot is associated.</td>
</tr>
<tr>
<td>Select Storage</td>
<td>Select the EVS disk for which the snapshot is to be created. The disk must</td>
</tr>
<tr>
<td></td>
<td>be available or in use. For details about how to create a disk, see Using</td>
</tr>
<tr>
<td></td>
<td>EVS Disks as Storage Volumes.</td>
</tr>
</tbody>
</table>

**Step 4** Click **Create**.

**Step 5** Click **Back to Snapshot List**. When the snapshot status changes to Normal, the snapshot is successfully created.

Click ^ on the left of the snapshot name to view the snapshot status and details.

----End

**Creating a Disk from a Snapshot**

On the snapshot list page, you can select a snapshot to create an EVS disk.

In this mode, pay attention to the following constraints:

- The disk type, disk mode, and encryption setting of the created disk are consistent with those of the snapshot's source EVS disk.
- A snapshot can be used to create a maximum of 128 EVS disks.
- Batch disk creation is not supported, and the quantity parameter must be set to 1.

**Procedure**

**Step 1** Log in to the CCE console. In the navigation pane, choose **Resource Management > Storage**.

**Step 2** Click the **Snapshot and backup** tab. In the snapshot list, locate the target snapshot and click **Create Data Volume** in the **Operation** column.

**Step 3** Set the parameters of the EVS disk. For details, see Using EVS Disks as Storage Volumes.

When you use a snapshot to create an EVS disk, the capacity must be greater than or equal to the snapshot size. In the condition that you do not specify the disk capacity, if the snapshot size is smaller than 10 GB, the default capacity 10 GB will be used as the disk capacity; if the snapshot size is greater than 10 GB, the disk capacity will be consistent with the snapshot size.

**Step 4** Click **Buy Now**.

**Step 5** Pay for the order as prompted and click **Submit**.

**Step 6** On the page indicating that the task is submitted successfully, click **Go to Storage**. On the **EVS** tab page, view the status of the EVS disk.
When the EVS disk status changes to Available, the EVS disk is created successfully.

----End

Deleting a Snapshot

If a snapshot is no longer used, you can release the virtual resources by deleting the snapshot from the system. Snapshot deletion has the following constraints:

- A snapshot can be deleted only when its status is Available or Error.
- If an EVS disk is deleted, all the snapshots created for this disk will also be deleted.

Procedure

Step 1 Log in to the CCE console. In the navigation pane, choose Resource Management > Storage.

Step 2 Click the Snapshot and backup tab. In the snapshot list, locate the target snapshot and click Delete in the Operation column.

Step 3 (Optional) If multiple snapshots are to be deleted, select ☑ in front of each snapshot and click Delete in the upper area of the list.

Step 4 In the dialog box displayed, enter DELETE, and click Yes.

----End

Related Operations

For details about EVS snapshots, see Snapshot FAQs.

Reference Documents

- Volume Snapshots
10 Storage (FlexVolume)

10.1 FlexVolume Overview

In container storage, you can use different types of volumes and mount them to containers in pods as many as you want.

In CCE, container storage is backed both by Kubernetes-native objects, such as emptyDir, hostPath, secret, and ConfigMap, and by HUAWEI CLOUD storage services.

CCE clusters of 1.13 and earlier versions use the storage-driver add-on to connect to HUAWEI CLOUD storage services to support Kubernetes FlexVolume driver for container storage. The FlexVolume driver has been deprecated in favor of the Container Storage Interface (CSI). **The everest add-on for CSI is installed in CCE clusters of 1.15 and later versions by default.** For details, see CSI Overview.

**NOTE**

- In CCE clusters earlier than Kubernetes 1.13, end-to-end capacity expansion of container storage is not supported, and the PVC capacity is inconsistent with the storage capacity.
- **In a cluster of v1.13 or earlier,** when an upgrade or bug fix is available for storage functionalities, you only need to install or upgrade the storage-driver add-on. Upgrading the cluster or creating a cluster is not required.

Notes and Constraints

- For clusters created in CCE, Kubernetes v1.15.11 is a transitional version in which the FlexVolume plug-in (storage-driver) is compatible with the CSI plug-in (everest). Clusters of v1.17 and later versions do not support FlexVolume any more. You need to use the everest add-on. For details about CSI and FlexVolume, see Differences Between CSI and FlexVolume Plug-ins.
- The FlexVolume plug-in will be maintained by Kubernetes developers, but new functionality will only be added to CSI. You are advised not to create storage that connects to the FlexVolume plug-in (storage-driver) in CCE any more. Otherwise, the storage resources may not function normally.
Differences Between CSI and FlexVolume Plug-ins

Table 10-1 CSI and FlexVolume

<table>
<thead>
<tr>
<th>Kubernetes Solution</th>
<th>CCE Add-on</th>
<th>Feature</th>
<th>Usage</th>
</tr>
</thead>
</table>
| CSI                 | everest    | CSI was developed as a standard for exposing arbitrary block and file storage systems to containerized workloads. Using CSI, third-party storage providers can deploy plugins exposing new storage systems in Kubernetes without having to touch the core Kubernetes code. In CCE, the everest add-on is installed by default in clusters of Kubernetes v1.15 and later to connect to HUAWEI CLOUD storage services (EVS, OBS, SFS, and SFS Turbo). The everest add-on consists of two parts:  
  • **everest-csi-controller** for storage volume creation, deletion, capacity expansion, and cloud disk snapshots  
  • **everest-csi-driver** for mounting, unmounting, and formatting storage volumes on nodes  
For details, see **everest**. | The everest add-on is installed by default in clusters of **v1.15 and later**. CCE will mirror the Kubernetes community by providing continuous support for updated CSI capabilities. |
**Kubernetes Solution**

**CCE Add-on**

**Feature**

FlexVolume is an out-of-tree plugin interface that has existed in Kubernetes since version 1.2 (before CSI). CCE provided FlexVolume volumes through the storage-driver add-on installed in clusters of Kubernetes v1.13 and earlier versions. This add-on connects clusters to HUAWEI CLOUD storage services (EVS, OBS, SFS, and SFS Turbo).

For details, see **storage-driver**.

**Usage**

For clusters of v1.13 or earlier that have been created, the installed FlexVolume plug-in (the storage-driver add-on in CCE) can still be used. CCE stops providing update support for this add-on, and you are advised to upgrade these clusters.

---

**NOTE**

- A cluster can use only one type of storage plug-ins.
- The FlexVolume plug-in cannot be replaced by the CSI plug-in in clusters of v1.13 or earlier. You can only upgrade these clusters. For details, see [Cluster Upgrade Between Major Versions](#).

**Notice on Using Add-ons**

- To use the CSI plug-in (the **everest** add-on in CCE), your cluster must be using **Kubernetes 1.15 or later**. This add-on is installed by default when you create a cluster of v1.15 or later. The FlexVolume plug-in (the **storage-driver** add-on in CCE) is installed by default when you create a cluster of v1.13 or earlier.
- If your cluster is upgraded from v1.13 to v1.15, **storage-driver** is replaced by everest (v1.1.6 or later) for container storage. The takeover does not affect the original storage functions.
- In version 1.2.0 of the everest add-on, **key authentication** is optimized when OBS is used. After the everest add-on is upgraded from a version earlier than 1.2.0, you need to restart all workloads that use OBS in the cluster. Otherwise, workloads may not be able to use OBS.

**Checking Storage Add-ons**

**Step 1** Log in to the CCE console.

**Step 2** In the navigation tree on the left, click **Add-ons**.

**Step 3** Click the **Add-on Instance** tab.

**Step 4** Select a cluster in the upper right corner. The default storage add-on installed during cluster creation is displayed, as shown below.
10.2 How Do I Change the Storage Class Used by a Cluster of v1.15 from FlexVolume to CSI Everest?

For clusters of v1.15.11-r1 and later, the CSI everest add-on has taken over all functions of the fuxi FlexVolume driver (the storage-driver add-on) for container storage management. In versions later than 1.17.9-r0, the fuxi FlexVolume driver (storage-driver) is no longer supported.

To migrate your storage volumes, create a static PV to associate with the original underlying storage, and then create a PVC to associate with this static PV. When you upgrade your application, mount the new PVC to the original mounting path to migrate the storage volumes.

WARNING

Services will be interrupted during the migration. Therefore, properly plan the migration and back up data.

Procedure

Step 1  (Optional) Back up data to prevent data loss in case of exceptions.

Step 2  Configure a YAML file of the PV in the CSI format according to the PV in the FlexVolume format and associate the PV with the existing storage.

To be specific, run the following commands to configure the pv-example.yaml file, which is used to create a PV.

touch pv-example.yaml

vi pv-example.yaml

Configuration example of a PV for an EVS volume:

```
apiVersion: v1
kind: PersistentVolume
metadata:
  labels:
    failure-domain.beta.kubernetes.io/region: ap-southeast-1
    failure-domain.beta.kubernetes.io/zone: <zone name>
  annotations:
    pv.kubernetes.io/provisioned-by: everest-csi-provisioner
    name: pv-evs-example
spec:
```

Figure 10-1 Checking which storage add-on was installed
accessModes:  
- ReadWriteOnce  
capacity:  
  storage: 10Gi  
csi:  
  driver: disk.csi.everest.io  
  fsType: ext4  
  volumeAttributes:  
    everest.io/disk-mode: SCSI  
    everest.io/disk-volume-type: SAS  
    storage.kubernetes.io/csiProvisionerIdentity: everest-csi-provisioner  
    volumeHandle: 0992dbda-6340-470e-a74e-4f0db288ed82  
persistentVolumeReclaimPolicy: Delete  
storageClassName: csi-disk

Pay attention to the fields in bold and red. The parameters are described as follows:

**Table 10-2** EVS volume configuration parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>failure-domain.beta.kubernetes.io/region</td>
<td>Region where the EVS disk is located. Use the same value as that of the FlexVolume PV.</td>
</tr>
<tr>
<td>failure-domain.beta.kubernetes.io/zone</td>
<td>AZ where the EVS disk is located. Use the same value as that of the FlexVolume PV.</td>
</tr>
<tr>
<td>name</td>
<td>Name of the PV, which must be unique in the cluster.</td>
</tr>
<tr>
<td>storage</td>
<td>EVS volume capacity in the unit of Gi. Use the value of <code>spec.capacity.storage</code> of the FlexVolume PV.</td>
</tr>
<tr>
<td>driver</td>
<td>Storage driver used to attach the volume. Set the driver to <code>disk.csi.everest.io</code> for the EVS volume.</td>
</tr>
<tr>
<td>volumeHandle</td>
<td>Volume ID of the EVS disk. Use the value of <code>spec.flexVolume.options.volumeId</code> of the FlexVolume PV.</td>
</tr>
<tr>
<td>everest.io/disk-mode</td>
<td>EVS disk mode. Use the value of <code>spec.flexVolume.options.disk-mode</code> of the FlexVolume PV.</td>
</tr>
<tr>
<td>everest.io/disk-volume-type</td>
<td>EVS disk type. Currently, high I/O (SAS) and ultra-high I/O (SSD) are supported. Use the value of <code>kubernetes.io/volumetype</code> in the storage class corresponding to <code>spec.storageClassName</code> of the FlexVolume PV.</td>
</tr>
<tr>
<td>storageClassName</td>
<td>Name of the Kubernetes storage class associated with the storage volume. Set this field to <code>csi-disk</code> for EVS disks.</td>
</tr>
</tbody>
</table>

**Configuration example of a PV for an SFS volume:**

```yaml
apiVersion: v1
kind: PersistentVolume
metadata:
```
name: pv-sfs-example
annotations:
  pv.kubernetes.io/provisioned-by: everest-csi-provisioner
spec:
  accessModes:
  - ReadWriteMany
  capacity:
    storage: 10Gi
csi:
  driver: nas.csi.everest.io
  fsType: nfs
  volumeAttributes:
    everest.io/share-export-location: sfs-nas01.cn-north-4.myhuaweicloud.com:/share-436304e8
    storage.kubernetes.io/csiProvisionerIdentity: everest-csi-provisioner
    volumeHandle: 6b2f00bb-ace0-41d8-9b3e-913c9aa6b695
    persistentVolumeReclaimPolicy: Delete
    storageClassName: csi-nas

Pay attention to the fields in bold and red. The parameters are described as follows:

**Table 10-3 SFS volume configuration parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Name of the PV, which must be unique in the cluster.</td>
</tr>
<tr>
<td>storage</td>
<td>File storage size in the unit of Gi. Use the value of <code>spec.capacity.storage</code> of the FlexVolume PV.</td>
</tr>
<tr>
<td>driver</td>
<td>Storage driver used to attach the volume. Set the driver to <code>nas.csi.everest.io</code> for the file system.</td>
</tr>
<tr>
<td>everest.io/share-export-location</td>
<td>Shared path of the file system. Use the value of <code>spec.flexVolume.options.deviceMountPath</code> of the FlexVolume PV.</td>
</tr>
<tr>
<td>volumeHandle</td>
<td>File system ID. Use the value of <code>spec.flexVolume.options.volumeID</code> of the FlexVolume PV.</td>
</tr>
<tr>
<td>storageClassName</td>
<td>Name of the Kubernetes storage class. Set this field to <code>csi-nas</code>.</td>
</tr>
</tbody>
</table>

**Configuration example of a PV for an OBS volume:**

apiVersion: v1
kind: PersistentVolume
metadata:
  name: pv-obs-example
  annotations:
    pv.kubernetes.io/provisioned-by: everest-csi-provisioner
spec:
  accessModes:
  - ReadWriteMany
  capacity:
    storage: 1Gi
csi:
  driver: obs.csi.everest.io
  fsType: s3fs
  volumeAttributes:
    everest.io/obs-volume-type: STANDARD
Pay attention to the fields in bold and red. The parameters are described as follows:

**Table 10-4** OBS volume configuration parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Name of the PV, which must be unique in the cluster.</td>
</tr>
<tr>
<td>storage</td>
<td>Storage capacity in the unit of Gi. Set this parameter to the fixed value 1Gi.</td>
</tr>
<tr>
<td>driver</td>
<td>Storage driver used to attach the volume. Set the driver to <code>obs.csi.everest.io</code> for the OBS volume.</td>
</tr>
<tr>
<td>fsType</td>
<td>File type. Value options are <code>obsfs</code> or <code>s3fs</code>. If the value is <code>s3fs</code>, an OBS bucket is created and mounted using s3fs. If the value is <code>obsfs</code>, an OBS parallel file system is created and mounted using obsfs. Set this parameter according to the value of <code>spec.flexVolume.options.posix</code> of the FlexVolume PV. If the value of <code>spec.flexVolume.options.posix</code> is <code>true</code>, set this parameter to <code>obsfs</code>. If the value is <code>false</code>, set this parameter to <code>s3fs</code>.</td>
</tr>
<tr>
<td>everest.io/obs-volume-type</td>
<td>Storage class, including <code>STANDARD</code> (standard bucket) and <code>WARM</code> (infrequent access bucket). Set this parameter according to the value of <code>spec.flexVolume.options.storage_class</code> of the FlexVolume PV. If the value of <code>spec.flexVolume.options.storage_class</code> is <code>standard</code>, set this parameter to <code>STANDARD</code>. If the value is <code>standard_ia</code>, set this parameter to <code>WARM</code>.</td>
</tr>
<tr>
<td>everest.io/region</td>
<td>Region where the OBS bucket is located. Use the value of <code>spec.flexVolume.options.region</code> of the FlexVolume PV.</td>
</tr>
<tr>
<td>volumeHandle</td>
<td>OBS bucket name. Use the value of <code>spec.flexVolume.options.volumeID</code> of the FlexVolume PV.</td>
</tr>
<tr>
<td>storageClassName</td>
<td>Name of the Kubernetes storage class. Set this field to <code>csi-obs</code>.</td>
</tr>
</tbody>
</table>

**Configuration example of a PV for an SFS Turbo volume:**

```yaml
apiVersion: v1
kind: PersistentVolume
metadata:
  name: pv-efs-example
  annotations:
    pv.kubernetes.io/provisioned-by: everest-csi-provisioner
spec:
  accessModes:...
Pay attention to the fields in bold and red. The parameters are described as follows:

Table 10-5 SFS Turbo volume configuration parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Name of the PV, which must be unique in the cluster.</td>
</tr>
<tr>
<td>storage</td>
<td>File system size. Use the value of spec.capacity.storage of the FlexVolume PV.</td>
</tr>
<tr>
<td>driver</td>
<td>Storage driver used to attach the volume. Set it to sfsturbo.csi.everest.io.</td>
</tr>
<tr>
<td>everest.io/share-export-location</td>
<td>Shared path of the SFS Turbo volume. Use the value of spec.flexVolume.options.deviceMountPath of the FlexVolume PV.</td>
</tr>
<tr>
<td>volumeHandle</td>
<td>SFS Turbo volume ID. Use the value of spec.flexVolume.options.volumeID of the FlexVolume PV.</td>
</tr>
<tr>
<td>storageClassName</td>
<td>Name of the Kubernetes storage class. Set this field to csi-sfsturbo for SFS Turbo volumes.</td>
</tr>
</tbody>
</table>

Step 3  Configure a YAML file of the PVC in the CSI format according to the PVC in the FlexVolume format and associate the PVC with the PV created in Step 2.

To be specific, run the following commands to configure the pvc-example.yaml file, which is used to create a PVC.

```
touch pvc-example.yaml
vi pvc-example.yaml
```

Configuration example of a PVC for an EVS volume:

```
apiVersion: v1
category: PersistentVolumeClaim
metadata:
  labels:
    failure-domain.beta.kubernetes.io/region: ap-southeast-1
    failure-domain.beta.kubernetes.io/zone: <zone name>
  annotations:
    everest.io/disk-volume-type: SAS
    volume.beta.kubernetes.io/storage-provisioner: everest-csi-provisioner
name: pvc-evs-example
namespace: default
```
spec:
  accessModes:
  - ReadWriteOnce
resources:
  requests:
    storage: 10Gi
volumeName: pv-evs-example
storageClassName: csi-disk

Pay attention to the fields in bold and red. The parameters are described as follows:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>failure-domain.beta.kubernetes.io/region</td>
<td>Region where the cluster is located. Use the same value as that of the FlexVolume PVC.</td>
</tr>
<tr>
<td>failure-domain.beta.kubernetes.io/zone</td>
<td>AZ where the EVS disk is deployed. Use the same value as that of the FlexVolume PVC.</td>
</tr>
<tr>
<td>everest.io/disk-volume-type</td>
<td>Storage class of the EVS disk. The value can be SAS or SSD. Set this parameter to the same value as that of the PV created in Step 2.</td>
</tr>
<tr>
<td>name</td>
<td>PVC name, which must be unique in the namespace. The value must be unique in the namespace. (If the PVC is dynamically created by a stateful application, the value of this parameter must be the same as the name of the FlexVolume PVC.)</td>
</tr>
<tr>
<td>namespace</td>
<td>Namespace to which the PVC belongs. Use the same value as that of the FlexVolume PVC.</td>
</tr>
<tr>
<td>storage</td>
<td>Requested capacity of the PVC, which must be the same as the storage size of the existing PV.</td>
</tr>
<tr>
<td>volumeName</td>
<td>Name of the PV. Set this parameter to the name of the static PV in Step 2.</td>
</tr>
<tr>
<td>storageClassName</td>
<td>Name of the Kubernetes storage class. Set this field to csi-disk for EVS disks.</td>
</tr>
</tbody>
</table>

Configuration example of a PVC for an SFS volume:

```yaml
apiVersion: v1
type: PersistentVolumeClaim
metadata:
  annotations:
    volume.beta.kubernetes.io/storage-provisioner: everest-csi-provisioner
  name: pvc-sfs-example
  namespace: default
spec:
  accessModes:
  - ReadWriteMany
```
Pay attention to the fields in bold and red. The parameters are described as follows:

**Table 10-7 PVC configuration parameters for an SFS volume**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>PVC name, which must be unique in the namespace. The value must be unique in the namespace. (If the PVC is dynamically created by a stateful application, the value of this parameter must be the same as the name of the FlexVolume PVC.)</td>
</tr>
<tr>
<td>namespace</td>
<td>Namespace to which the PVC belongs. Use the same value as that of the FlexVolume PVC.</td>
</tr>
<tr>
<td>storage</td>
<td>Storage capacity, in the unit of Gi. The value must be the same as the storage size of the existing PV.</td>
</tr>
<tr>
<td>storageClassName</td>
<td>Set this field to <strong>csi-nas</strong>.</td>
</tr>
<tr>
<td>volumeName</td>
<td>Name of the PV. Set this parameter to the name of the static PV in <strong>Step 2</strong>.</td>
</tr>
</tbody>
</table>

**Configuration example of a PVC for an OBS volume:**

```yaml
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  annotations:
    volume.beta.kubernetes.io/storage-provisioner: everest-csi-provisioner
everest.io/obs-volume-type: STANDARD
  csi.storage.k8s.io/fstype: s3fs
name: pvc-obs-example
namespace: default
spec:
  accessModes:
  - ReadWriteMany
  resources:
    requests:
      storage: 1Gi
    storageClassName: csi-obs
    volumeName: pv-obs-example
```

Pay attention to the fields in bold and red. The parameters are described as follows:
Table 10-8 PVC configuration parameters for an OBS volume

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>everest.io/obs-volume-type</td>
<td>OBS volume type, which can be <strong>STANDARD</strong> (standard bucket) and <strong>WARM</strong> (infrequent access bucket). Set this parameter to the same value as that of the PV created in <strong>Step 2</strong>.</td>
</tr>
<tr>
<td>csi.storage.k8s.io/fstype</td>
<td>File type, which can be <strong>obsfs</strong> or <strong>s3fs</strong>. The value must be the same as that of <strong>fsType</strong> of the static OBS volume PV.</td>
</tr>
<tr>
<td>name</td>
<td>PVC name, which must be unique in the namespace. The value must be unique in the namespace. (If the PVC is dynamically created by a stateful application, the value of this parameter must be the same as the name of the FlexVolume PVC.)</td>
</tr>
<tr>
<td>namespace</td>
<td>Namespace to which the PVC belongs. Use the same value as that of the FlexVolume PVC.</td>
</tr>
<tr>
<td>storage</td>
<td>Storage capacity in the unit of Gi. Set this parameter to the fixed value <strong>1Gi</strong>.</td>
</tr>
<tr>
<td>storageClassName</td>
<td>Name of the Kubernetes storage class. Set this field to <strong>csi-obs</strong>.</td>
</tr>
<tr>
<td>volumeName</td>
<td>Name of the PV. Set this parameter to the name of the static PV created in <strong>Step 2</strong>.</td>
</tr>
</tbody>
</table>

Configuration example of a PVC for an SFS Turbo volume:

```yaml
apiVersion: v1
category: PersistentVolumeClaim
metadata:
  annotations:
    volume.beta.kubernetes.io/storage-provisioner: everest-csi-provisioner
name: pvc-efs-example
namespace: default
spec:
  accessModes:
  - ReadWriteMany
  resources:
    requests:
      storage: 10Gi
  storageClassName: csi-sfsturbo
  volumeName: pv-efs-example
```

Pay attention to the fields in bold and red. The parameters are described as follows:
Table 10-9 PVC configuration parameters for an SFS Turbo volume

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>PVC name, which must be unique in the namespace. The value must be unique in the namespace. (If the PVC is dynamically created by a stateful application, the value of this parameter must be the same as the name of the FlexVolume PVC.)</td>
</tr>
<tr>
<td>namespace</td>
<td>Namespace to which the PVC belongs. Use the same value as that of the FlexVolume PVC.</td>
</tr>
<tr>
<td>storageClassName</td>
<td>Name of the Kubernetes storage class. Set this field to csi-sfsturbo.</td>
</tr>
<tr>
<td>storage</td>
<td>Storage capacity, in the unit of Gi. The value must be the same as the storage size of the existing PV.</td>
</tr>
<tr>
<td>volumeName</td>
<td>Name of the PV. Set this parameter to the name of the static PV created in Step 2.</td>
</tr>
</tbody>
</table>

**Step 4**  Upgrade the workload to use a new PVC.

**For Deployments**
1. Run the `kubectl create -f` commands to create a PV and PVC.
   ```
kubectl create -f pv-example.yaml
kubectl create -f pvc-example.yaml
```

   **NOTE**
   Replace the example file name `pvc-example.yaml` in the preceding commands with the names of the YAML files configured in Step 2 and Step 3.

2. Go to the CCE console. On the workload upgrade page, click Upgrade > Advanced Settings > Data Storage > Cloud Storage.

3. Uninstall the old storage and add the PVC in the CSI format. Retain the original mounting path in the container.
4. Click Submit.
5. Wait until the pods are running.

**For StatefulSets that use existing storage**
1. Run the `kubectl create -f` commands to create a PV and PVC.
   ```
kubectl create -f pv-example.yaml
kubectl create -f pvc-example.yaml
```
**NOTE**

Replace the example file name `pvc-example.yaml` in the preceding commands with the names of the YAML files configured in Step 2 and Step 3.

2. Run the `kubectl edit` command to edit the StatefulSet and use the newly created PVC.

```bash
kubectl edit sts sts-example -n xxx
```

**NOTE**

Replace `sts-example` in the preceding command with the actual name of the StatefulSet to upgrade. `xxx` indicates the namespace to which the StatefulSet belongs.

3. Wait until the pods are running.

**NOTE**

The current console does not support the operation of adding new cloud storage for StatefulSets. Use the kubectl commands to replace the storage with the newly created PVC.

For StatefulSets that use dynamically allocated storage

1. Back up the PV and PVC in the flexVolume format used by the StatefulSet.

   ```bash
   kubectl get pvc xxx -n {namespaces} -oyaml > pvc-backup.yaml
   kubectl get pv xxx -n {namespaces} -oyaml > pv-backup.yaml
   ```

2. Change the number of pods to 0.

3. On the storage page, disassociate the flexVolume PVC used by the StatefulSet.

4. Run the `kubectl create -f` commands to create a PV and PVC.

   ```bash
   kubectl create -f pv-example.yaml
   kubectl create -f pvc-example.yaml
   ```

   **NOTE**

   Replace the example file name `pvc-example.yaml` in the preceding commands with the names of the YAML files configured in Step 2 and Step 3.

5. Change the number of pods back to the original value and wait until the pods are running.
The dynamic allocation of storage for StatefulSets is achieved by using `volumeClaimTemplates`. This field cannot be modified by Kubernetes. Therefore, data cannot be migrated by using a new PVC.

The PVC naming rule of the `volumeClaimTemplates` is fixed. When a PVC that meets the naming rule exists, this PVC is used.

Therefore, disassociate the original PVC first, and then create a PVC with the same name in the CSI format.

6. (Optional) Recreate the stateful application to ensure that a CSI PVC is used when the application is scaled out. Otherwise, FlexVolume PVCs are used in scaling out.

- Run the following command to obtain the YAML file of the StatefulSet:

  ```bash
  kubectl get sts xxx -n {namespaces} -oyaml > sts.yaml
  ```

- Run the following command to back up the YAML file of the StatefulSet:

  ```bash
  cp sts.yaml sts-backup.yaml
  ```

- Modify the definition of `volumeClaimTemplates` in the YAML file of the StatefulSet.

  `vi sts.yaml`

Configuration example of `volumeClaimTemplates` for an EVS volume:

```yaml
volumeClaimTemplates:
- metadata:
  name: pvc-161070049798261342
  namespace: default
  creationTimestamp: null
  annotations:
  everest.io/disk-volume-type: SAS
spec:
  accessModes:
  - ReadWriteOnce
  resources:
  requests:
    storage: 10Gi
  storageClassName: csi-disk
```

The parameter value must be the same as the PVC of the EVS volume created in Step 3.

Configuration example of `volumeClaimTemplates` for an SFS volume:

```yaml
volumeClaimTemplates:
- metadata:
  name: pvc-161063441560279697
  namespace: default
  creationTimestamp: null
spec:
  accessModes:
  -ReadWriteMany
  resources:
  requests:
    storage: 10Gi
  storageClassName: csi-nas
```

The parameter value must be the same as the PVC of the SFS volume created in Step 3.
Configuration example of `volumeClaimTemplates` for an OBS volume:

```
volumeClaimTemplates:
  - metadata:
      name: pvc-161070100417416148
      namespace: default
      creationTimestamp: null
      annotations:
        csi.storage.k8s.io/fstype: s3fs
        everest.io/obs-volume-type: STANDARD
      spec:
        accessModes:
          - ReadWriteMany
        resources:
          requests:
            storage: 1Gi
        storageClassName: csi-obs
```

The parameter value must be the same as the PVC of the OBS volume created in Step 3.

- Delete the StatefulSet.

```
kubectl delete sts xxx -n {namespaces}
```

- Create the StatefulSet.

```
kubectl create -f sts.yaml
```

**Step 5** Check service functions.

1. Check whether the application is running properly.
2. Checking whether the data storage is normal.

**NOTE**

If a rollback is required, perform **Step 4**. Select the PVC in FlexVolume format and upgrade the application.

**Step 6** Uninstall the PVC in the FlexVolume format.

If the application functions normally, unbind the PVC in the FlexVolume format on the storage management page.

**Figure 10-2 Unbinding a volume**

You can also run the `kubectl` command to delete the PVC and PV of the FlexVolume format.
10.3 Using EVS Disks as Storage Volumes

10.3.1 Overview

To achieve persistent storage, CCE allows you to mount the storage volumes created from Elastic Volume Service (EVS) disks to a path of a container. When the container is migrated, the mounted EVS volumes are also migrated. By using EVS volumes, you can mount the remote file directory of storage system into a container so that data in the data volume is permanently preserved even when the container is deleted.

Figure 10-3 Mounting EVS volumes to CCE

Description

- **User-friendly**: Similar to formatting disks for on-site servers in traditional layouts, you can format block storage (disks) mounted to cloud servers, and create file systems on them.
- **Data isolation**: Each server uses an independent block storage device (disk).
- **Private network**: User can access data only in private networks of data centers.
Capacity and performance: The capacity of a single volume is limited (TB-level), but the performance is excellent (ms-level read/write I/O latency).

Restriction: EVS disks that have partitions or have non-ext4 file systems cannot be imported.

Use cases: You can use such volumes for Deployments, jobs, and StatefulSets deployed with a single pod. Common use cases include high-performance computing (HPC), enterprise core cluster applications, enterprise OA systems, and dev/test.

10.3.2 Using EVS Volumes

Prerequisites

You have created a CCE cluster and installed the FlexVolume plug-in (storage-driver) in the cluster.

Notes and Constraints

- By default, CCE creates EVS disks billed in pay-per-use mode. To use EVS disks billed in yearly/monthly mode, see Yearly/Monthly-Billed EVS Disks.
- EVS disks cannot be attached across AZs and cannot be used by multiple workloads, multiple pods of the same workload, or multiple jobs.
- Data in a shared disk cannot be shared between nodes in a CCE cluster. If the same EVS disk is attached to multiple nodes, read and write conflicts and data cache conflicts may occur. When creating a Deployment, you are advised to create only one pod if you want to use EVS disks.
- When you create a StatefulSet and add a cloud storage volume, existing EVS volumes cannot be used.
- EVS disks that have partitions or have non-ext4 file systems cannot be imported.
- Container storage in CCE clusters of Kubernetes 1.13 or later version supports encryption. Currently, E2E encryption is supported only in certain regions.
- Volumes cannot be created in specified enterprise projects. Only the default enterprise project is supported.
- The following operations apply to clusters of Kubernetes 1.13 or earlier.

Buying an EVS Disk

**Step 1** Log in to the CCE console. In the navigation pane, choose Resource Management > Storage. Click Buy EVS Disk.

**Step 2** Configure basic disk information. Table 10-10 describes the parameters.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>* PVC Name</td>
<td><strong>New PVC Name</strong>: name of the PVC to be created. A storage volume is automatically created when a PVC is created. One PVC corresponds to one storage volume. The storage volume name is automatically generated when the PVC is created.</td>
</tr>
<tr>
<td>Cluster Name</td>
<td>Cluster where the EVS disk is deployed.</td>
</tr>
<tr>
<td>Namespace</td>
<td>Namespace where the EVS disk is deployed. You can retain the default value or specify one.</td>
</tr>
<tr>
<td>Volume Capacity (GB)</td>
<td>Size of the storage to be created.</td>
</tr>
<tr>
<td>Access Mode</td>
<td>Access permissions of user applications on storage resources (PVs).</td>
</tr>
<tr>
<td></td>
<td>- <strong>ReadWriteOnce</strong> (RWO): The volume can be mounted as read-write by a single node, and data reading and writing are supported based on a non-shared EVS volume. EVS volumes in RWO mode are supported since v1.13.10-r1.</td>
</tr>
<tr>
<td>AZ</td>
<td>AZ to which the disk belongs.</td>
</tr>
<tr>
<td>Type</td>
<td>Type of the new EVS disk.</td>
</tr>
<tr>
<td></td>
<td>- <strong>High I/O</strong>: uses serial attached SCSI (SAS) drives to store data.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Ultra-high I/O</strong>: uses solid state disk (SSD) drives to store data.</td>
</tr>
<tr>
<td>Encryption</td>
<td><strong>KMS Encryption</strong> is deselected by default.</td>
</tr>
<tr>
<td></td>
<td>After <strong>KMS Encryption</strong> is selected, Key Management Service (KMS), an easy-to-use and highly secure cloud service for your keys, will be used for EVS disks. If no agency has been created, click <strong>Create Agency</strong> and set the following parameters:</td>
</tr>
<tr>
<td></td>
<td>- <strong>Agency Name</strong>: Agencies can be used to assign permissions to trusted accounts or cloud services for a specific period of time. If no agency is created, click <strong>Create Agency</strong>. The agency name <strong>EVSAccessKMS</strong> indicates that EVS is granted the permission to access KMS. After EVS is authorized successfully, it can obtain KMS keys to encrypt and decrypt EVS systems.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Key Name</strong>: After a key is created, it can be loaded and used in containerized applications. For details on how to create a key, see <strong>Creating a CMK</strong>.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Key ID</strong>: generated by default.</td>
</tr>
<tr>
<td></td>
<td>This function is supported only for clusters of v1.13.10 and later in certain regions.</td>
</tr>
</tbody>
</table>
Step 3 Click Buy Now. Review your order, click Submit, and wait until the creation is successful.

The file system is displayed in the list. When its status becomes Normal, the file system is created successfully.

Step 4 Click the volume name to view detailed information about the volume.

---End

Adding an EVS Volume

Step 1 Create a workload or job by referring to Creating a Deployment, Creating a StatefulSet, or Creating a Job. During creation, expand Data Storage after adding a container. On the Cloud Volume tab page, click Add Cloud Volume.

Step 2 Set the storage volume type to EVS.

Table 10-11 Parameters required for mounting an EVS volume

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| Type      | **EVS**: You can use EVS disks the same way you use traditional hard disks on servers. EVS disks deliver higher data reliability and I/O throughput and are easy to use. They can be used for file systems, databases, or other system software and applications that require block storage resources. **CAUTION**  
  - To attach an EVS disk to a workload, you must set the number of pods to 1 when creating the workload. If multiple pods are created, you cannot attach EVS disks.  
  - When you create a StatefulSet and add a cloud storage volume, existing EVS volumes cannot be used.  
  - EVS disks cannot be attached across AZs and cannot be used by multiple workloads, multiple pods of the same workload, or multiple jobs. |

| Allocation Mode | Manual | Select a created disk. If no disk is available, follow the prompts to create one.  
  For the same cluster and namespace, you can use an existing storage volume when creating a Deployment (with Allocation Mode set to Manual).  
  When creating a StatefulSet, you can only use a volume automatically allocated by the system (only Automatic is available for Allocation Mode). |
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Automatic</td>
<td>If you select <strong>Automatic</strong>, you need to configure the following items:</td>
</tr>
<tr>
<td></td>
<td>1. <strong>Access Mode</strong>: permissions of user applications on storage resources (PVs).</td>
</tr>
<tr>
<td></td>
<td>- <strong>ReadWriteOnce</strong> (RWO): A non-shared EVS volume is mounted as read-write to a pod by a single node. EVS volumes in RWO mode are supported since v1.13.10-r1.</td>
</tr>
<tr>
<td></td>
<td>- <strong>ReadWriteMany</strong> (RWX): A shared EVS volume is mounted as read-write to a pod. The use of this mode is limited in clusters earlier than v1.15.6 because shared EVS volumes cannot ensure cross-node data read/write consistency for containers. You need to ensure that only one pod is allowed to read and write data to the volume.</td>
</tr>
<tr>
<td></td>
<td>2. <strong>Availability Zone</strong>: AZ where the storage volume is located. Only the AZ where the worker node is located can be selected.</td>
</tr>
<tr>
<td></td>
<td>3. <strong>Sub-Type</strong>: Select a storage subtype.</td>
</tr>
<tr>
<td></td>
<td>- <strong>High I/O</strong>: uses serial attached SCSI (SAS) drives to store data.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Ultra-high I/O</strong>: uses solid state disk (SSD) drives to store data.</td>
</tr>
<tr>
<td></td>
<td>4. <strong>Storage Capacity</strong>: Enter the storage capacity in the unit of GB. Ensure that the storage capacity quota is not exceeded; otherwise, creation will fail.</td>
</tr>
<tr>
<td></td>
<td>5. After you select <strong>KMS Encryption</strong>, Key Management Service (KMS), an easy-to-use and highly secure service, will be enabled for EVS disks. This function is supported only for clusters of v1.13.10 and later in certain regions. If no agency has been created, click <strong>Create Agency</strong> and set the following parameters:</td>
</tr>
<tr>
<td></td>
<td>- <strong>Agency Name</strong>: Agencies can be used to assign permissions to trusted accounts or cloud services for a specific period of time. If no agency is created, click <strong>Create Agency</strong>. The agency name <strong>EVSAccessKMS</strong> indicates that EVS is granted the permission to access KMS. After EVS is authorized successfully, it can obtain KMS keys to encrypt and decrypt EVS systems.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Key Name</strong>: After a key is created, it can be loaded and used in containerized applications. For details on how to create a key, see <strong>Creating a CMK</strong>.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Key ID</strong>: generated by default.</td>
</tr>
</tbody>
</table>
Parameter | Description
---|---
Add Container Path | 1. Click Add Container Path.
2. **Container Path**: Enter the container path to which the volume is mounted.

**NOTICE**
- Do not mount the volume to a system directory such as / or /var/run; this action may cause container errors. You are advised to mount the volume to an empty directory. If the directory is not empty, ensure that there are no files affecting container startup in the directory. Otherwise, such files will be replaced, resulting in failures to start the container and create the workload.
- If the volume is mounted to a high-risk directory, you are advised to use an account with minimum permissions to start the container; otherwise, high-risk files on the host machine may be damaged.

3. **Set permissions.**
   - **Read-only**: You can only read the data in the mounted volumes.
   - **Read/Write**: You can modify the data in the mounted volumes. Newly written data is not migrated if the container is migrated, which causes a data loss.

**Figure 10-4 Adding a cloud volume**

Step 3  Click **OK**.

----End

**Importing an EVS Disk**

CCE allows you to import existing EVS disks.
**NOTE**

An EVS disk can be imported into only one namespace. If an EVS disk has been imported into a namespace, it is invisible in other namespaces and cannot be imported again. If you want to import an EVS disk that has file system (ext4) formatted, ensure that no partition has been created for the disk. Otherwise, data may be lost.

**Step 1** Log in to the CCE console. In the navigation pane, choose Resource Management > Storage. On the EVS tab page, click Import.

**Step 2** Select one or more EVS disks that you want to import. Then, click OK.

--- End

### Yearly/Monthly-Billed EVS Disks

If you want to change the billing mode of an EVS disk from pay-per-use to yearly/monthly after the disk is created or imported, perform the following operations:

**Method 1:**

1. Create an EVS disk on the CCE console. The disk will be billed on a pay-per-use basis by default.
2. On the CCE console, choose Resource Management > Storage > EVS, and click More > Change Billing Mode at the row of the target EVS disk. This change of billing mode takes some time.

**Method 2:**

1. Create a yearly/monthly-billed EVS disk with the required capacity on the cloud server console.
2. Back on the EVS tab page of the CCE console, click Import. Select the EVS disk you just created, and click OK.

--- CAUTION ---

When a yearly/monthly-billed EVS disk is deleted on CCE, it is only unbound from CCE. If you want to permanently delete the EVS disk, unbind it on CCE and then unsubscribe from it in the Billing Center.

### Unbinding an EVS Disk

After an EVS volume is successfully created or imported, the EVS volume is automatically bound to the current cluster and cannot be used by other clusters. When the volume is unbound from the cluster, other clusters can still use the volume.

If the EVS volume has been mounted to a workload, it cannot be unbound from the cluster.

**Step 1** Log in to the CCE console. In the navigation pane, choose Resource Management > Storage. In the EVS disk list, click Unbind next to the target EVS disk.

**Step 2** Confirm the unbinding, and click OK.

--- End
Related Operations

After an EVS volume is created, you can perform operations described in Table 10-12.

<table>
<thead>
<tr>
<th>Operation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deleting an EVS volume</td>
<td>1. Select the EVS volume to be deleted and click Delete in the Operation column.</td>
</tr>
<tr>
<td></td>
<td>2. Follow the prompts to delete the EVS volume.</td>
</tr>
</tbody>
</table>

10.3.3 (kubectl) Automatically Creating an EVS Disk

Scenario

CCE supports creating EVS volumes through PersistentVolumeClaims (PVCs).

Prerequisites

You have created a CCE cluster and installed the FlexVolume plug-in (storage-driver) in the cluster.

Notes and Constraints

The following configuration example applies to clusters of Kubernetes 1.13 or earlier.

Procedure

Step 1 Use kubectl to connect to the cluster. For details, see Connecting to a Cluster Using kubectl.

Step 2 Run the following commands to configure the pvc-evs-auto-example.yaml file, which is used to create a PVC.

```
touch pvc-evs-auto-example.yaml
vi pvc-evs-auto-example.yaml
```

Example YAML file for clusters of v1.9, v1.11, and v1.13:

```yaml
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: pvc-evs-auto-example
namespace: default
annotations:
  volume.beta.kubernetes.io/storage-class: sas
labels:
  failure-domain.beta.kubernetes.io/region: ap-southeast-1
spec:
  accessModes:
  - ReadWriteMany
```
Table 10-13 Key parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>volume.beta.kubernetes.io/storage-class</td>
<td>EVS disk type. The value is in lowercase. Supported values: High I/O (SAS) and Ultra-high I/O (SSD)</td>
</tr>
<tr>
<td>failure-domain.beta.kubernetes.io/region</td>
<td>Region where the cluster is located. For details about the value of <code>region</code>, see Regions and Endpoints.</td>
</tr>
<tr>
<td>failure-domain.beta.kubernetes.io/zone</td>
<td>AZ where the EVS volume is created. It must be the same as the AZ planned for the workload. For details about the value of <code>zone</code>, see Regions and Endpoints.</td>
</tr>
<tr>
<td>storage</td>
<td>Storage capacity in the unit of Gi.</td>
</tr>
<tr>
<td>accessModes</td>
<td>Read/write mode of the volume. You can set this parameter to <code>ReadWriteMany</code> (shared volume) and <code>ReadWriteOnly</code> (non-shared volume).</td>
</tr>
</tbody>
</table>

Step 3 Run the following command to create a PVC.

```
kubectl create -f pvc-evs-auto-example.yaml
```

After the command is executed, an EVS disk is created in the region where the cluster is located. Choose Storage > EVS to view the EVS disk. Alternatively, you can view the EVS disk based on the volume name on the EVS console.

----End

10.3.4 (kubectl) Creating a PV from an Existing EVS Disk

Scenario

CCE allows you to create a PersistentVolume (PV) using an existing EVS disk. After the PV is created, you can create a PersistentVolumeClaim (PVC) and bind it to the PV.

Prerequisites

You have created a CCE cluster and installed the FlexVolume plug-in (`storage-driver`) in the cluster.
Notes and Constraints

The following configuration example applies to clusters of Kubernetes 1.13 or earlier.

Procedure

Step 1 Log in to the EVS console, create an EVS disk, and record the volume ID, capacity, and disk type of the EVS disk.

Step 2 Use kubectl to connect to the cluster. For details, see Connecting to a Cluster Using kubectl.

Step 3 Create two YAML files for creating the PersistentVolume (PV) and PersistentVolumeClaim (PVC). Assume that the file names are pv-evs-example.yaml and pvc-evs-example.yaml.

```
touch pv-evs-example.yaml
```

<table>
<thead>
<tr>
<th>Kubernetes Version</th>
<th>Description</th>
<th>YAML Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.11.7 ≤ K8s version ≤ 1.13</td>
<td>Clusters from v1.11.7 to v1.13</td>
<td>Example YAML</td>
</tr>
<tr>
<td>1.11 ≤ K8s version &lt; 1.11.7</td>
<td>Clusters from v1.11 to v1.11.7</td>
<td>Example YAML</td>
</tr>
<tr>
<td>K8s version = 1.9</td>
<td>Clusters of v1.9</td>
<td>Example YAML</td>
</tr>
</tbody>
</table>

Clusters from v1.11.7 to v1.13

- **Example YAML file for the PV:**

  ```yaml
  apiVersion: v1
  kind: PersistentVolume
  metadata:
    labels:
      failure-domain.beta.kubernetes.io/region: ap-southeast-1
      failure-domain.beta.kubernetes.io/zone: ap-southeast-1a
    annotations:
      pv.kubernetes.io/provisioned-by: flexvolume-huawei.com/fuxivol
  name: pv-evs-example
  spec:
    accessModes:
    - ReadWriteMany
    capacity:
      storage: 10Gi
    claimRef:
      apiVersion: v1
      kind: PersistentVolumeClaim
      name: pvc-evs-example
      namespace: default
    flexVolume:
      driver: huawei.com/fuxivol
      fsType: ext4
      options:
        - disk-mode: SCSI
        - fsType: ext4
        - volumeID: 0992dbda-6340-470e-a74e-4f0db288ed82
        - persistentVolumeReclaimPolicy: Delete
        - storageClassName: sas
  ```
### Table 10-14 Key parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>failure-domain.beta.kubernetes.io/region</td>
<td>Region where the cluster is located. For details about the value of <strong>region</strong>, see <em>Regions and Endpoints</em>.</td>
</tr>
<tr>
<td>failure-domain.beta.kubernetes.io/zone</td>
<td>AZ where the EVS volume is created. It must be the same as the AZ planned for the workload. For details about the value of <strong>zone</strong>, see <em>Regions and Endpoints</em>.</td>
</tr>
<tr>
<td>storage</td>
<td>EVS volume capacity in the unit of Gi.</td>
</tr>
<tr>
<td>storageClassName</td>
<td>EVS disk type. Supported values: High I/O (SAS) and Ultra-high I/O (SSD)</td>
</tr>
<tr>
<td>driver</td>
<td>Storage driver. For EVS disks, set this parameter to <em>huawei.com/fuxivol</em>.</td>
</tr>
<tr>
<td>volumeID</td>
<td>Volume ID of the EVS disk. To obtain the volume ID, log in to the CCE console, choose <em>Resource Management &gt; Storage</em>, click the PVC name in the <strong>EVS</strong> tab page, and copy the PVC ID on the PVC details page.</td>
</tr>
<tr>
<td>disk-mode</td>
<td>Device type of the EVS disk. The value is <strong>VBD</strong> or <strong>SCSI</strong>. For CCE clusters earlier than v1.11.7, you do not need to set this field. The value defaults to <strong>VBD</strong>. This field is mandatory for CCE clusters from v1.11.7 to v1.13 that use Linux x86. As the EVS volumes dynamically provisioned by a PVC are created from SCSI EVS disks, you are advised to choose <strong>SCSI</strong> when manually creating volumes (static PVs). Volumes in the VBD mode can still be used after cluster upgrades.</td>
</tr>
<tr>
<td>spec.claimRef.apiVersion</td>
<td>The value is fixed at <strong>v1</strong>.</td>
</tr>
<tr>
<td>spec.claimRef.kind</td>
<td>The value is fixed at <strong>PersistentVolumeClaim</strong>.</td>
</tr>
<tr>
<td>spec.claimRef.name</td>
<td>PVC name. The value is the same as the name of the PVC created in the next step.</td>
</tr>
</tbody>
</table>
### Example YAML file for the PVC:
```yaml
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  annotations:
    volume.beta.kubernetes.io/storage-class: sas
    volume.beta.kubernetes.io/storage-provisioner: flexvolume-huawei.com/fuxivol
labels:
  failure-domain.beta.kubernetes.io/region: ap-southeast-1
  failure-domain.beta.kubernetes.io/zone: ap-southeast-1a
name: pvc-evs-example
namespace: default
spec:
  accessModes:
  - ReadWriteMany
  resources:
    requests:
      storage: 10Gi
volumeName: pv-evs-example
```

### Table 10-15 Key parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>spec.claimRef.namespace</code></td>
<td>Namespace of the PVC. The value is the same as the namespace of the PVC created in the next step.</td>
</tr>
</tbody>
</table>

#### Clusters from v1.11 to v1.11.7

- **Example YAML file for the PV:**
  ```yaml
  apiVersion: v1
  kind: PersistentVolume
  ```
metadata:
  labels:
    - failure-domain.beta.kubernetes.io/region: ap-southeast-1
    - failure-domain.beta.kubernetes.io/zone: pv-evs-example
spec:
  accessModes:
    - ReadWriteMany
  capacity:
    storage: 10Gi
  flexVolume:
    driver: huawei.com/fuxivol
    fsType: ext4
    options:
      - fsType: ext4
      - volumeID: 0992dbda-6340-470e-a74e-4f0db88ed82
  persistentVolumeReclaimPolicy: Delete
  storageClassName: sas

Table 10-16 Key parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>failure-domain.beta.kubernetes.io/region</td>
<td>Region where the cluster is located. For details about the value of region, see Regions and Endpoints.</td>
</tr>
<tr>
<td>failure-domain.beta.kubernetes.io/zone</td>
<td>AZ where the EVS volume is created. It must be the same as the AZ planned for the workload. For details about the value of zone, see Regions and Endpoints.</td>
</tr>
<tr>
<td>storage</td>
<td>EVS volume capacity in the unit of Gi.</td>
</tr>
<tr>
<td>storageClassName</td>
<td>EVS disk type. Supported values: High I/O (SAS) and Ultra-high I/O (SSD)</td>
</tr>
<tr>
<td>driver</td>
<td>Storage driver. For EVS disks, set this parameter to huawei.com/fuxivol.</td>
</tr>
<tr>
<td>volumeID</td>
<td>Volume ID of the EVS disk. To obtain the volume ID, log in to the CCE console, choose Resource Management &gt; Storage, click the PVC name in the EVS tab page, and copy the PVC ID on the PVC details page.</td>
</tr>
</tbody>
</table>
disk-mode

Device type of the EVS disk. The value is **VBD** or **SCSI**.
For CCE clusters earlier than v1.11.7, you do not need to set this field. The default value is **VBD**.
This field is mandatory for CCE clusters from v1.11.7 to v1.13 that use Linux x86.
As the EVS volumes dynamically provisioned by a PVC are created from SCSI EVS disks, you are advised to choose **SCSI** when manually creating volumes (static PVs). Volumes in the VBD mode can still be used after cluster upgrades.

---

**Example YAML file for the PVC:**
```yaml
apiVersion: v1
class: PersistentVolumeClaim
metadata:
  annotations:
    volume.beta.kubernetes.io/storage-class: sas
    volume.beta.kubernetes.io/storage-provisioner: flexvolume-huawei.com/fuxivol
  labels:
    failure-domain.beta.kubernetes.io/region: ap-southeast-1
    failure-domain.beta.kubernetes.io/zone: ap-southeast-1a
name: pvc-evs-example
namespace: default
spec:
  accessModes:
    - ReadWriteMany
  resources:
    requests:
      storage: 10Gi
  volumeName: pv-evs-example
```

---

**Table 10-17 Key parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>volume.beta.kubernetes.io/storage-class</code></td>
<td>Storage class. The value can be <strong>sas</strong> or <strong>ssd</strong>. The value must be the same as that of the existing PV.</td>
</tr>
<tr>
<td><code>volume.beta.kubernetes.io/storage-provisioner</code></td>
<td>The field must be set to <strong>flexvolume-huawei.com/fuxivol</strong>.</td>
</tr>
<tr>
<td><code>failure-domain.beta.kubernetes.io/region</code></td>
<td>Region where the cluster is located. For details about the value of <strong>region</strong>, see <strong>Regions and Endpoints</strong>.</td>
</tr>
<tr>
<td><code>failure-domain.beta.kubernetes.io/zone</code></td>
<td>AZ where the EVS volume is created. It must be the same as the AZ planned for the workload. For details about the value of <strong>zone</strong>, see <strong>Regions and Endpoints</strong>.</td>
</tr>
</tbody>
</table>
### Parameter Description

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>storage</td>
<td>Requested capacity in the PVC, in Gi. The value must be the same as the</td>
</tr>
<tr>
<td></td>
<td>storage size of the existing PV.</td>
</tr>
<tr>
<td>volumeName</td>
<td>Name of the PV.</td>
</tr>
</tbody>
</table>

### Clusters of v1.9

- **Example YAML file for the PV:**
  ```yaml
  apiVersion: v1
  kind: PersistentVolume
  metadata:
    labels:
      failure-domain.beta.kubernetes.io/region: ap-southeast-1
      failure-domain.beta.kubernetes.io/zone: name: pv-evs-example
      namespace: default
  spec:
    accessModes:
      - ReadWriteMany
    capacity:
      storage: 10Gi
    flexVolume:
      driver: huawei.com/fuxivol
      fsType: ext4
      options:
        fsType: ext4
        kubernetes.io/namespace: default
        volumeID: 0992dbda-6340-470e-a74e-4f0db288ed82
      persistentVolumeReclaimPolicy: Delete
      storageClassName: sas
  ```

### Table 10-18 Key parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>failure-domain.beta.kubernetes.io/region</td>
<td>Region where the cluster is located. For details about the value of region, see Regions and Endpoints.</td>
</tr>
<tr>
<td>failure-domain.beta.kubernetes.io/zone</td>
<td>AZ where the EVS volume is created. It must be the same as the AZ planned for the workload. For details about the value of zone, see Regions and Endpoints.</td>
</tr>
<tr>
<td>storage</td>
<td>EVS volume capacity in the unit of Gi.</td>
</tr>
<tr>
<td>storageClassName</td>
<td>EVS disk type. Supported values: High I/O (SAS) and Ultra-high I/O (SSD)</td>
</tr>
<tr>
<td>driver</td>
<td>Storage driver. For EVS disks, set this parameter to huawei.com/fuxivol.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>volumeID</td>
<td>Volume ID of the EVS disk. To obtain the volume ID, log in to the CCE console, choose <strong>Resource Management &gt; Storage</strong>, click the PVC name in the EVS tab page, and copy the PVC ID on the PVC details page.</td>
</tr>
<tr>
<td>disk-mode</td>
<td>Device type of the EVS disk. The value is <strong>VBD</strong> or <strong>SCSI</strong>. For CCE clusters earlier than v1.11.7, you do not need to set this field. The default value is <strong>VBD</strong>. This field is mandatory for CCE clusters from v1.11.7 to v1.13 that use Linux x86. As the EVS volumes dynamically provisioned by a PVC are created from SCSI EVS disks, you are advised to choose <strong>SCSI</strong> when manually creating volumes (static PVs). Volumes in the VBD mode can still be used after cluster upgrades.</td>
</tr>
</tbody>
</table>

**Example YAML file for the PVC:**
```yaml
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  annotations:
    volume.beta.kubernetes.io/storage-class: sas
    volume.beta.kubernetes.io/storage-provisioner: flexvolume-huawei.com/fuxivol
  labels:
    failure-domain.beta.kubernetes.io/region: ap-southeast-1
    failure-domain.beta.kubernetes.io/zone:
  name: pvc-evs-example
  namespace: default
spec:
  accessModes:
  - ReadWriteMany
  resources:
    requests:
      storage: 10Gi
  volumeName: pv-evs-example
  volumeNamespace: default
```

**Table 10-19 Key parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>volume.beta.kubernetes.io/storage-class</td>
<td>Storage class, which must be the same as that of the existing PV.</td>
</tr>
<tr>
<td>volume.beta.kubernetes.io/storage-provisioner</td>
<td>The field must be set to <strong>flexvolume-huawei.com/fuxivol</strong>.</td>
</tr>
</tbody>
</table>
Step 4 Create the PV.

`kubectl create -f pv-evs-example.yaml`

Step 5 Create the PVC.

`kubectl create -f pvc-evs-example.yaml`

After the operation is successful, choose Resource Management > Storage to view the created PVC. You can also view the EVS disk by name on the EVS console.

Step 6 (Optional) Add the metadata associated with the cluster to ensure that the EVS disk associated with the mounted static PV is not deleted when the node or cluster is deleted.

**CAUTION**

If you skip this step in this example or when creating a static PV or PVC, ensure that the EVS disk associated with the static PV has been unbound from the node before you delete the node.

1. Obtain the tenant token. For details, see Obtaining a User Token.
2. Obtain the EVS access address EVS_ENDPOINT. For details, see Regions and Endpoints.
3. Add the metadata associated with the cluster to the EVS disk backing the static PV.

   ```bash
   curl -X POST ${EVS_ENDPOINT}/v2/${project_id}/volumes/${volume_id}/metadata --insecure \
   -d '{"metadata":{"cluster_id": "${cluster_id}", "namespace": "$pvc_namespace"}}' \
   -H 'Accept:application/json' -H 'Content-Type:application/json;charset=utf8' \
   -H 'X-Auth-Token:${TOKEN}'
   ```
Table 10-20 Key parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EVS_ENDPOINT</td>
<td>EVS access address. Set this parameter to the value obtained in Step 6.2.</td>
</tr>
<tr>
<td>project_id</td>
<td>Project ID. You can click the login user in the upper right corner of the console page, select My Credentials from the drop-down list, and view the project ID on the Projects tab page.</td>
</tr>
<tr>
<td>volume_id</td>
<td>ID of the associated EVS disk. Set this parameter to the value of volume_id in Step 2. You can also log in to the EVS console, click the name of the EVS disk to be imported, and obtain the ID from Summary on the disk details page, as shown in Figure 10-5.</td>
</tr>
<tr>
<td>cluster_id</td>
<td>ID of the cluster where the EVS PV is to be created. On the CCE console, choose Resource Management &gt; Clusters. Click the name of the cluster to be associated. On the cluster details page, obtain the cluster ID, as shown in Figure 10-6.</td>
</tr>
<tr>
<td>pvc_namespace</td>
<td>Namespace where the PVC is to be bound.</td>
</tr>
<tr>
<td>TOKEN</td>
<td>User token. Set this parameter to the value obtained in Step 6.1.</td>
</tr>
</tbody>
</table>

Figure 10-5 Obtaining the disk ID

![Figure 10-5 Obtaining the disk ID](image-url)
For example, run the following commands:

curl -X POST https://evs.cn-north-4.myhuaweicloud.com:443/v2/060576866680d5762f52c0150e726aa7/volumes/69c9619d-174c-4c41-837e-31b892604e14/metadata --insecure 
-d '{"metadata":{"cluster_id": "71e8277e-80c7-11ea-925c-0255ac100442", "namespace": "default"}}' 
-H 'Accept:application/json' -H 'Content-Type:application/json;charset=utf8' 
-H 'X-Auth-Token:MIIPe******IsIm1ldG

After the request is executed, run the following commands to check whether the EVS disk has been associated with the metadata of the cluster:

curl -X GET ${EVS_ENDPOINT}/v2/${project_id}/volumes/${volume_id}/metadata --insecure 
-H 'X-Auth-Token:${TOKEN}'

For example, run the following commands:

curl -X GET https://evs.cn-north-4.myhuaweicloud.com/v2/060576866680d5762f52c0150e726aa7/volumes/69c9619d-174c-4c41-837e-31b892604e14/metadata --insecure 
-H 'X-Auth-Token:MIIPeAYJ***9t1c31ASaQ=='

The command output displays the current metadata of the EVS disk.

```json
{
  "metadata": {
    "namespace": "default",
    "cluster_id": "71e8277e-80c7-11ea-925c-0255ac100442",
    "hw:passthrough": "true"
  }
}
```

10.3.5 (kubectl) Creating a Pod Mounted with an EVS Volume

Scenario

After an EVS volume is created or imported to CCE, you can mount it to a workload.

---

**NOTICE**

EVS disks cannot be attached across AZs. Before mounting a volume, you can run the `kubectl get pvc` command to query the available PVCs in the AZ where the current cluster is located.
Prerequisites

You have created a CCE cluster and installed the FlexVolume plug-in (storage-driver) in the cluster.

Notes and Constraints

The following configuration example applies to clusters of Kubernetes 1.13 or earlier.

Procedure

**Step 1** Use kubectl to connect to the cluster. For details, see Connecting to a Cluster Using kubectl.

**Step 2** Run the following commands to configure the evs-deployment-example.yaml file, which is used to create a Deployment.

```
touch evs-deployment-example.yaml
vi evs-deployment-example.yaml
```

Example of mounting an EVS volume to a Deployment (PVC-based, shared volume):

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: evs-deployment-example
  namespace: default
spec:
  replicas: 1
  selector:
    matchLabels:
      app: evs-deployment-example
  template:
    metadata:
      labels:
        app: evs-deployment-example
    spec:
      containers:
      - image: nginx
        name: container-0
        volumeMounts:
          - mountPath: /tmp
            name: pvc-evs-example
        imagePullSecrets:
          - name: default-secret
        restartPolicy: Always
        volumes:
          - name: pvc-evs-example
            persistentVolumeClaim:
              claimName: pvc-evs-auto-example
```

<table>
<thead>
<tr>
<th>Table 10-21 Key parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Parent Parameter</strong></td>
</tr>
<tr>
<td>spec.template.spec.containers.volumeMounts</td>
</tr>
<tr>
<td>Parent Parameter</td>
</tr>
<tr>
<td>------------------</td>
</tr>
<tr>
<td>spec.template.spec.containers.volumeMounts</td>
</tr>
<tr>
<td>spec.template.spec.volumes</td>
</tr>
<tr>
<td>spec.template.spec.volumes.persistentVolume-Claim</td>
</tr>
</tbody>
</table>

**NOTE**

`spec.template.spec.containers.volumeMounts.name` and `spec.template.spec.volumes.name` must be consistent because they have a mapping relationship.

Mounting an EVS volume to a StatefulSet (PVC template-based, non-shared volume):

**Example YAML:**

```yaml
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: deploy-evs-sas-in
spec:
  replicas: 1
  selector:
    matchLabels:
      app: deploy-evs-sata-in
  template:
    metadata:
      labels:
        app: deploy-evs-sata-in
    failure-domain.beta.kubernetes.io/region: ap-southeast-1
    failure-domain.beta.kubernetes.io/zone: ap-southeast-1a
  spec:
    containers:
      - name: container-0
        image: 'nginx:1.12-alpine-perl'
        volumeMounts:
          - name: bs-sas-mountoptionpvc
            mountPath: /tmp
        imagePullSecrets:
          - name: default-secret
        volumeClaimTemplates:
          - metadata:
              labels:
                app: deploy-evs-sata-in
              failure-domain.beta.kubernetes.io/region: ap-southeast-1
              failure-domain.beta.kubernetes.io/zone: ap-southeast-1a
            spec:
              accessModes:
                - ReadWriteMany
              resources:
                requests:
                  storage: 10Gi
              annotations:
                volume.beta.kubernetes.io/storage-class: sas
                volume.beta.kubernetes.io/storage-provisioner: flexvolume-huawei.com/fuxivol
              spec:
                accessModes:
                  - ReadWriteMany
                resources:
                  requests:
                    storage: 10Gi
              serviceName: wwww
```
### Table 10-22 Key parameters

<table>
<thead>
<tr>
<th>Parent Parameter</th>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>metadata</td>
<td>name</td>
<td>Name of the created workload.</td>
</tr>
<tr>
<td>spec.template.spec.containers</td>
<td>image</td>
<td>Image of the workload.</td>
</tr>
<tr>
<td>spec.template.spec.containers.volume</td>
<td>mountPath</td>
<td>Mount path of the container. In this example, the volume is mounted to the /tmp directory.</td>
</tr>
<tr>
<td>spec.serviceName</td>
<td>serviceName</td>
<td>Service corresponding to the workload. For details about how to create a Service, see <a href="#">Creating a StatefulSet</a>.</td>
</tr>
</tbody>
</table>

**NOTE**

spec.template.spec.containers.volumeMounts.name and spec.volumeClaimTemplates.metadata.name must be consistent because they have a mapping relationship.

**Step 3** Run the following command to create the pod:

```
kubectl create -f evs-deployment-example.yaml
```

After the creation is complete, log in to the CCE console. In the navigation pane, choose **Resource Management** > **Storage** > **EVS**. Then, click the PVC name. On the PVC details page, you can view the binding relationship between the EVS volume and the PVC.

----End

### 10.4 Using SFS Turbo File Systems as Storage Volumes

#### 10.4.1 Overview

CCE allows you to mount a volume created from an SFS Turbo file system to a container to store data persistently. Provisioned on demand and fast, SFS Turbo is suitable for DevOps, container microservices, and enterprise OA scenarios.
**Description**

- **Standard file protocols**: You can mount file systems as volumes to servers, the same as using local directories.
- **Data sharing**: The same file system can be mounted to multiple servers, so that data can be shared.
- **Private network**: User can access data only in private networks of data centers.
- **Data isolation**: The on-cloud storage service provides exclusive cloud file storage, which delivers data isolation and ensures IOPS performance.
- **Use cases**: Deployments/StatefulSets in the ReadWriteMany mode, DaemonSets, and jobs created for high-traffic websites, log storage, DevOps, and enterprise OA applications

**10.4.2 Using SFS Turbo Volumes**

**Prerequisites**

You have created a CCE cluster and installed the FlexVolume plug-in (*storage-driver*) in the cluster.

**Notes and Constraints**

- Currently, SFS Turbo file systems cannot be directly created on CCE.
- The following operations apply to clusters of Kubernetes 1.13 or earlier.

**Importing an SFS Turbo Volume**

CCE allows you to import existing SFS Turbo volumes. Currently, only HUAWEI CLOUD accounts and IAM users with the CCE Administrator permissions can import SFS Turbo volumes.
Step 1 Log in to the CCE console. In the navigation pane, choose Resource Management > Storage. On the SFS Turbo tab page, click Import.

Step 2 Select one or more SFS Turbo volumes that you want to import.

Step 3 Select the cluster and namespace to which you want to import the volumes.

Step 4 Click OK. The volumes are displayed in the list. When PVS Status becomes Bound, the volumes are imported successfully.

---End

Adding an SFS Turbo Volume

Step 1 Create a workload or job by referring to Creating a Deployment, Creating a StatefulSet, Creating a DaemonSet, or Creating a Job. After you have added a container, choose Data Storage > Cloud Volume, and then click Add Cloud Volume.

Step 2 Set the storage volume type to SFS Turbo.

Table 10-23 Parameters for configuring an SFS Turbo volume

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>SFS Turbo: applicable to DevOps, containerized microservices, and enterprise OA applications.</td>
</tr>
<tr>
<td>Allocation Mode</td>
<td>Manual Select an existing SFS Turbo volume. You need to import SFS Turbo volumes in advance. For details, see Importing an SFS Turbo Volume.</td>
</tr>
</tbody>
</table>
### Parameter Description

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| Add Container Path         | Configure the following parameters:  
   1. **subPath**: Enter the subpath of the file storage, for example, `/tmp`. This parameter specifies a subpath inside the referenced volume instead of its root. If this parameter is not specified, the root path is used. Currently, only file storage is supported. The value must be a relative path and cannot start with a slash (/) or ../.  
   2. **Container Path**: Enter the mount path in the container, for example, `/tmp`. The mount path must not be a system directory, such as `/` and `/var/run`. Otherwise, an exception occurs. You are advised to mount the volume to an empty directory. If the directory is not empty, ensure that there are no files affecting container startup in the directory. Otherwise, such files will be replaced, resulting in failures to start the container and create the workload.  
   **NOTICE** If the volume is mounted to a high-risk directory, you are advised to use an account with minimum permissions to start the container; otherwise, high-risk files on the host machine may be damaged.  
   3. Set permissions.  
      - **Read-only**: You can only read the data in the mounted volumes.  
      - **Read/Write**: You can modify the data in the mounted volumes. Newly written data is not migrated if the container is migrated, which causes a data loss. |
  
Click **Add Container Path** to add multiple settings. Then, click **OK**.

### Step 3
Click **OK**.

-----End

### Unbinding an SFS Turbo Volume

When an SFS Turbo volume is successfully imported to a cluster, the volume is bound to the cluster. The volume can also be imported to other clusters. When the volume is unbound from the cluster, other clusters can still import and use the volume.

If the SFS Turbo volume has been mounted to a workload, the volume cannot be unbound from the cluster.

**Step 1** Log in to the CCE console. In the navigation pane, choose **Resource Management** > **Storage**. In the SFS Turbo volume list, click **Unbind** next to the target volume.
Step 2  In the dialog box displayed, click **OK**.

---End

10.4.3 (kubectl) Creating a PV from an Existing SFS Turbo File System

Scenario

CCE allows you to use an existing SFS Turbo file system to create a PersistentVolume (PV). After the creation is successful, you can create a PersistentVolumeClaim (PVC) and bind it to the PV.

Prerequisites

You have created a CCE cluster and installed the FlexVolume plug-in (**storage-driver**) in the cluster.

Notes and Constraints

The following configuration example applies to clusters of Kubernetes 1.13 or earlier.

Procedure

Step 1  Log in to the SFS console, create a file system, and record the file system ID, shared path, and capacity.

Step 2  Use kubectl to connect to the cluster. For details, see **Connecting to a Cluster Using kubectl**.

Step 3  Create two YAML files for creating the PV and PVC. Assume that the file names are **pv-efs-example.yaml** and **pvc-efs-example.yaml**.

```
touch pv-efs-example.yaml  pvc-efs-example.yaml
```

- **Example YAML file for the PV:**

```yaml
apiVersion: v1
driver: huawei.com/fuxiefs
driver: efs

options:
  deviceMountPath: 192.168.0.169:/
  fsType: efs
```

volumeID: 8962a2a2-a583-4b7f-bb74-fe76712d8414
persistentVolumeReclaimPolicy: Delete
storageClassName: efs-standard

Table 10-24 Key parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>driver</td>
<td>Storage driver used to mount the volume. Set it to huawei.com/fuxiefs.</td>
</tr>
<tr>
<td>deviceMountPath</td>
<td>Shared path of the SFS Turbo volume.</td>
</tr>
<tr>
<td>volumeID</td>
<td>SFS Turbo volume ID.</td>
</tr>
<tr>
<td></td>
<td>To obtain the ID, log in to the CCE console, choose Resource Management &gt; Storage, click the PVC name in the SFS Turbo tab page, and copy the PVC ID on the PVC details page.</td>
</tr>
<tr>
<td>storage</td>
<td>File system size.</td>
</tr>
<tr>
<td>storageClassName</td>
<td>Volume type supported by SFS Turbo. The value can be efs-standard and efs-performance.</td>
</tr>
<tr>
<td></td>
<td>Currently, SFS Turbo does not support dynamic creation; therefore, this parameter is not used for now.</td>
</tr>
<tr>
<td>spec.claimRef.apiVersion</td>
<td>The value is fixed at v1.</td>
</tr>
<tr>
<td>spec.claimRef.kind</td>
<td>The value is fixed at PersistentVolumeClaim.</td>
</tr>
<tr>
<td>spec.claimRef.name</td>
<td>The value is the same as the name of the PVC created in the next step.</td>
</tr>
<tr>
<td>spec.claimRef.namespace</td>
<td>The value is the same as the namespace of the PVC created in the next step.</td>
</tr>
</tbody>
</table>

- **Example YAML file for the PVC:**

```yaml
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  annotations:
    volume.beta.kubernetes.io/storage-class: efs-standard
    volume.beta.kubernetes.io/storage-provisioner: flexvolume-huawei.com/fuxiefs
  name: pvc-efs-example
  namespace: default
spec:
  accessModes:
    - ReadWriteMany
  resources:
    requests:
      storage: 100Gi
  volumeName: pv-efs-example
```
### Table 10-25 Key parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>volume.beta.kubernetes.io/storage-class</code></td>
<td>Read/write mode supported by SFS Turbo. The value can be <code>efs-standard</code> or <code>efs-performance</code>. The value must be the same as that of the existing PV.</td>
</tr>
<tr>
<td><code>volume.beta.kubernetes.io/storage-provisioner</code></td>
<td>The field must be set to <code>flexvolume-huawei.com/fuxiefs</code>.</td>
</tr>
<tr>
<td><code>storage</code></td>
<td>Storage capacity, in the unit of Gi. The value must be the same as the storage size of the existing PV.</td>
</tr>
<tr>
<td><code>volumeName</code></td>
<td>Name of the PV.</td>
</tr>
</tbody>
</table>

**NOTE**

The VPC to which the SFS Turbo file system belongs must be the same as the VPC of the ECS VM planned for the workload. Ports 111, 445, 2049, 2051, and 20048 must be enabled in the security groups.

**Step 4**  Create the PV.

`kubectl create -f pv-efs-example.yaml`

**Step 5**  Create the PVC.

`kubectl create -f pvc-efs-example.yaml`

----End

### 10.4.4 (kubectl) Creating a Deployment Mounted with an SFS Turbo Volume

**Scenario**

After an SFS Turbo volume is created or imported to CCE, you can mount the volume to a workload.

**Prerequisites**

You have created a CCE cluster and installed the FlexVolume plug-in (storage-driver) in the cluster.

**Notes and Constraints**

The following configuration example applies to clusters of Kubernetes 1.13 or earlier.
**Procedure**

**Step 1** Use kubectl to connect to the cluster. For details, see [Connecting to a Cluster Using kubectl](#).

**Step 2** Run the following commands to configure the `efs-deployment-example.yaml` file, which is used to create a Deployment:

```bash
touch efs-deployment-example.yaml
vi efs-deployment-example.yaml
```

Example of mounting an SFS Turbo volume to a Deployment (PVC-based, shared volume):

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: efs-deployment-example # Workload name
  namespace: default
spec:
  replicas: 1
  selector:
    matchLabels:
      app: efs-deployment-example
template:
  metadata:
    labels:
      app: efs-deployment-example
  spec:
    containers:
      - image: nginx
        name: container-0
        volumeMounts:
          - mountPath: /tmp # Mount path
            name: pvc-efs-example
    restartPolicy: Always
    imagePullSecrets:
      - name: default-secret
    volumes:
      - name: pvc-efs-example
        persistentVolumeClaim:
          claimName: pvc-sfs-auto-example # PVC name
```

**Table 10-26** Key parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Name of the created Deployment.</td>
</tr>
<tr>
<td>app</td>
<td>Name of the application running in the Deployment.</td>
</tr>
<tr>
<td>mountPath</td>
<td>Mount path in the container. In this example, the mount path is <code>/tmp</code>.</td>
</tr>
</tbody>
</table>

**NOTE**

`spec.template.spec.containers.volumeMounts.name` and `spec.template.spec.volumes.name` must be consistent because they have a mapping relationship.
Step 3  Run the following command to create the pod:

    kubectl create -f efs-deployment-example.yaml

After the creation is complete, choose Storage > SFS Turbo on the CCE console and click the PVC name. On the PVC details page, you can view the binding relationship between SFS Turbo and PVC.

----End

10.4.5 (kubectl) Creating a StatefulSet Mounted with an SFS Turbo Volume

Scenario

CCE allows you to use an existing SFS Turbo volume to create a StatefulSet.

Prerequisites

You have created a CCE cluster and installed the FlexVolume plug-in (storage-driver) in the cluster.

Notes and Constraints

The following configuration example applies to clusters of Kubernetes 1.13 or earlier.

Procedure

Step 1  Create an SFS Turbo volume and record the volume name.

Step 2  Use kubectl to connect to the cluster. For details, see Connecting to a Cluster Using kubectl.

Step 3  Create a YAML file for creating the workload. Assume that the file name is efs-statefulset-example.yaml.

    touch efs-statefulset-example.yaml
    vi efs-statefulset-example.yaml

Example YAML:

    apiVersion: apps/v1
    kind: StatefulSet
    metadata:
        name: efs-statefulset-example
        namespace: default
    spec:
        replicas: 1
        selector:
            matchLabels:
                app: efs-statefulset-example
        template:
            metadata:
                annotations:
                    metrics.alpha.kubernetes.io/custom-endpoints: '{"api":"","path":"","port":"","names":""}'
            pod.alpha.kubernetes.io/initialized: "true"
            labels:
app: efs-statefulset-example
spec:
  containers:
  - image: 'nginx:1.0.0'
    name: container-0
    resources:
      requests: {}
      limits: {}
  env:
    - name: PAAS_APP_NAME
      value: efs-statefulset-example
    - name: PAAS_NAMESPACE
      value: default
    - name: PAAS_PROJECT_ID
      value: b1829681cc34f929baa8b9e95abf88b
  volumeMounts:
  - name: efs-statefulset-example
    mountPath: /tmp
    readOnly: false
    subPath: ''
  imagePullSecrets:
  - name: default-secret
  terminationGracePeriodSeconds: 30
  volumes:
  - persistentVolumeClaim:
    claimName: cce-efs-import-jnr481gm-3y5o
  affinity: {}
  tolerations:
  - key: node.kubernetes.io/not-ready
    operator: Exists
    effect: NoExecute
    tolerationSeconds: 300
  - key: node.kubernetes.io/unreachable
    operator: Exists
    effect: NoExecute
    tolerationSeconds: 300
  podManagementPolicy: OrderedReady
  serviceName: test
  updateStrategy:
    type: RollingUpdate

Table 10-27 Key parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>replicas</td>
<td>Number of pods.</td>
</tr>
<tr>
<td>name</td>
<td>Name of the created workload.</td>
</tr>
<tr>
<td>image</td>
<td>Image used by the workload.</td>
</tr>
<tr>
<td>mountPath</td>
<td>Mount path in the container.</td>
</tr>
<tr>
<td>serviceName</td>
<td>Service corresponding to the workload. For details about how to create a Service, see Creating a StatefulSet.</td>
</tr>
<tr>
<td>claimName</td>
<td>Name of an existing PVC.</td>
</tr>
</tbody>
</table>
NOTE

spec.template.spec.containers.volumeMounts.name and
spec.template.spec.volumes.name must be consistent because they have a mapping
relationship.

Step 4 Create the StatefulSet.

kubectl create -f efs-statefulset-example.yaml

----End

10.5 Using OBS Buckets as Storage Volumes

10.5.1 Overview

CCE allows you to mount a volume created from an Object Storage Service (OBS)
bucket to a container to store data persistently. Object storage is commonly used
in cloud workloads, data analysis, content analysis, and hotspot objects.

Figure 10-8 Mounting OBS volumes to CCE

Storage Class

Object storage offers three storage classes, Standard, Infrequent Access, and
Archive, to satisfy different requirements for storage performance and costs.

- The Standard storage class features low access latency and high throughput.
  It is therefore applicable to storing a large number of hot files (frequently
  accessed every month) or small files (less than 1 MB). The application
  scenarios include big data analytics, mobile apps, hot videos, and picture
  processing on social media.
- The Infrequent Access storage class is ideal for storing data that is semi-
  frequently accessed (less than 12 times a year), with requirements for quick
  response. The application scenarios include file synchronization or sharing,
and enterprise-level backup. It provides the same durability, access latency, and throughput as the Standard storage class but at a lower cost. However, the Infrequent Access storage class has lower availability than the Standard storage class.

- The Archive storage class is suitable for archiving data that is rarely-accessed (averagely once a year). The application scenarios include data archiving and long-term data backup. The Archive storage class is secure and durable at an affordable low cost, which can be used to replace tape libraries. However, it may take hours to restore data from the Archive storage class.

**Description**

- **Standard APIs**: With HTTP RESTful APIs, OBS allows you to use client tools or third-party tools to access object storage.
- **Data sharing**: Servers, embedded devices, and IoT devices can use the same path to access shared object data in OBS.
- **Public/Private networks**: OBS allows data to be accessed from public networks to meet Internet application requirements.
- **Capacity and performance**: No capacity limit; high performance (read/write I/O latency within 10 ms).
- **Use cases**: Deployments/StatefulSets in the ReadOnlyMany mode and jobs created for big data analysis, static website hosting, online video on demand (VoD), gene sequencing, intelligent video surveillance, backup and archiving, and enterprise cloud boxes (web disks). You can create object storage by using the OBS console, tools, and SDKs.

**Reference**

CCE clusters can also be mounted with OBS buckets of third-party tenants, including OBS parallel file systems (preferred) and OBS object buckets. For details, see [Mounting an Object Storage Bucket of a Third-Party Tenant](#).

### 10.5.2 Using OBS Volumes

**Prerequisites**

You have created a CCE cluster and installed the FlexVolume plug-in (**storage-driver**) in the cluster.

**Notes and Constraints**

- CCE clusters of v1.7.3-r8 and earlier do not support OBS volumes. You need to upgrade these clusters or create clusters of a later version that supports OBS.
- Kunpeng clusters do not support obsfs. Therefore, parallel file systems cannot be mounted.
- Volumes cannot be created in specified enterprise projects. Only the default enterprise project is supported.
- The following operations apply to clusters of Kubernetes 1.13 or earlier.
Preparations

To mount reliable and stable OBS buckets as volumes, you must create AK/SK before you create OBS buckets.

The procedure for configuring the AK/SK is as follows:

1. Log in to the CCE console. In the navigation pane, choose Resource Management > Storage.

2. On the OBS tab page, click AK/SK in the notice.

Figure 10-9 Configuring the AK/SK

3. Click ..., select a key file, and click Upload to upload the key file.

4. Select the corresponding workload and click Restart.

NOTICE

When creating an OBS volume, you must use the AK/SK. If the key file is not uploaded, the pod will fail to be started or OBS data access will be abnormal due to the volume mounting failure.

- For details about how to obtain access keys, see Obtaining Access Keys (AK and SK).
- For details about how to manage access keys, see Access Keys.

Creating an OBS Volume

Step 1 Log in to the CCE console. In the navigation pane, choose Resource Management > Storage.

Step 2 Click the OBS tab and click Create OBS Bucket.

Step 3 Configure basic information, as shown in Table 10-28.

Table 10-28 Parameters for creating an OBS volume

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>* PVC Name</td>
<td>Name of the new PVC, which is different from the volume name. The actual volume name is automatically generated when the PV is created by the PVC. The name contains 3 to 55 characters (excluding the prefix). It must contain lowercase letters, digits, and hyphens (-), and cannot start or end with a hyphen (-).</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Cluster Name</td>
<td>Cluster to which the OBS volume belongs.</td>
</tr>
<tr>
<td>Namespace</td>
<td>Namespace to which the volume belongs. The default value is default.</td>
</tr>
<tr>
<td>Instance Type</td>
<td>Type of the storage instance created on OBS.</td>
</tr>
<tr>
<td></td>
<td>● Parallel file system: supported when the cluster version is 1.15 or later and the everest add-on version is 1.0.2 or later. For details, see About Parallel File System and Using OBS Parallel File Systems.</td>
</tr>
<tr>
<td></td>
<td>● Object bucket: A bucket is a container for storing objects in OBS. OBS provides flat storage in the form of buckets and objects. Unlike the conventional multi-layer directory structure of file systems, all objects in a bucket are stored at the same logical layer.</td>
</tr>
<tr>
<td>Storage Class</td>
<td>This parameter is displayed when you select Object bucket for Instance Type.</td>
</tr>
<tr>
<td></td>
<td>This parameter indicates the storage classes supported by OBS.</td>
</tr>
<tr>
<td></td>
<td>● Standard: applicable to scenarios where a large number of hotspot files or small-sized files need to be accessed frequently (multiple times per month on average) and require fast access response.</td>
</tr>
<tr>
<td></td>
<td>● Infrequent access: applicable to scenarios where data is not frequently accessed (less than 12 times per year on average) but requires fast access response.</td>
</tr>
<tr>
<td>Storage Policy</td>
<td>Object storage has the following policies:</td>
</tr>
<tr>
<td></td>
<td>Private: Only the bucket owner has full control over the bucket. Unauthorized users do not have permissions to access the bucket.</td>
</tr>
<tr>
<td>Access Mode</td>
<td>Access permissions of user applications on storage resources (PVs).</td>
</tr>
<tr>
<td></td>
<td>● ReadWriteOnce (RWO): The volume is mounted as read-write by a single node.</td>
</tr>
<tr>
<td></td>
<td>● ReadWriteMany (RWX): The volume is mounted as read-write by multiple nodes.</td>
</tr>
</tbody>
</table>

**Step 4** Click Create.

After the OBS volume is successfully created, it is displayed in the OBS volume list. Click the PVC name to view detailed information about the OBS volume.

---End
Adding an OBS Volume

**Step 1** Create a workload or job by referring to Creating a Deployment, Creating a StatefulSet, Creating a DaemonSet, or Creating a Job. After you have added a container, choose **Data Storage > Cloud Volume**, and then click **Add Cloud Volume**.

**Step 2** Set **Type** to **OBS**.

### Table 10-29 OBS volume parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| **Type** | Select **OBS**.  
*OBS*: Standard and Infrequent Access OBS buckets are supported. OBS buckets are commonly used for big data analytics, cloud native applications, static website hosting, and backup/active archiving. |
| **Allocation Mode** |  |
| Manual | **Name**: Select a created OBS volume.  
**Sub-Type**: class of the selected volume. The value can be **Standard** or **Infrequent access**, and you do not need to set this parameter. |
| Automatic | Type of the storage instance created on OBS.  
- **Parallel file system**: supported when the cluster version is 1.15 or later and the everest add-on version is 1.0.2 or later. For details, see About Parallel File System and Using OBS Parallel File Systems.  
- **Object bucket**: A bucket is a container for storing objects in OBS. For details, see Buckets.  
**Sub-Type**: Select **Standard** or **Infrequent access**. |
### Parameter | Description
---|---
Add Container Path | Configure the following parameters:
1. **Container Path**: Enter the mount path in the container, for example, `/tmp`. The mount path must not be a system directory, such as `/` and `/var/run`. Otherwise, an exception occurs. You are advised to mount the volume to an empty directory. If the directory is not empty, ensure that there are no files affecting container startup in the directory. Otherwise, such files will be replaced, resulting in failures to start the container and create the workload.

**NOTICE**
If the volume is mounted to a high-risk directory, you are advised to use an account with minimum permissions to start the container; otherwise, high-risk files on the host machine may be damaged.
2. Set permissions.
   - **Read-only**: You can only read the data in the mounted volumes.
   - **Read/Write**: You can modify the data in the mounted volumes. Newly written data is not migrated if the container is migrated, which causes a data loss.

Click **Add Container Path** to add multiple settings. Then, click **OK**.

---

**Step 3**  
Click **OK**.

---End

### Importing an OBS Volume

CCE allows you to import existing OBS volumes.

**Step 1**  
Log in to the CCE console. In the navigation pane, choose **Resource Management > Storage**. On the **OBS** tab page, click **Import**.

**Step 2**  
Select one or more OBS volumes that you want to import.

**Step 3**  
Select the target cluster and namespace.

**Step 4**  
Click **OK**.

---End

### Unbinding an OBS Volume

When an OBS volume is successfully created, the OBS volume is automatically bound to the current cluster. Other clusters can also use the OBS volume. When the volume is unbound from the cluster, other clusters can still use the volume.

If the volume has been mounted to a workload, the volume cannot be unbound from the cluster.
**Step 1** Log in to the CCE console. In the navigation pane, choose **Resource Management** > **Storage**. In the OBS volume list, click **Unbind** next to the target OBS volume.

**Step 2** In the dialog box displayed, click **Yes**.

---End

**Related Operations**

After an OBS volume is created, you can perform the operation described in **Table 10-30**.

<table>
<thead>
<tr>
<th>Operation</th>
<th>Description</th>
</tr>
</thead>
</table>
| Deleting an OBS volume   | 1. Select the OBS volume to be deleted and click **Delete** in the **Operation** column.  
  2. Follow the prompts to delete the volume. |

### 10.5.3 (kubectl) Automatically Creating an OBS Volume

**Scenario**

During the use of OBS, expected OBS buckets can be automatically created and mounted as volumes. Currently, standard and infrequent access OBS buckets are supported, which correspond to **obs-standard** and **obs-standard-ia**, respectively.

**Prerequisites**

You have created a CCE cluster and installed the FlexVolume plug-in (**storage-driver**) in the cluster.

**Notes and Constraints**

The following configuration example applies to clusters of Kubernetes 1.13 or earlier.

**Procedure**

**Step 1** Use kubectl to connect to the cluster. For details, see **Connecting to a Cluster Using kubectl**.

**Step 2** Run the following commands to configure the **pvc-obs-auto-example.yaml** file, which is used to create a PVC.

- `touch pvc-obs-auto-example.yaml`
- `vi pvc-obs-auto-example.yaml`

**Example YAML:**

```yaml
apiVersion: v1
kind: PersistentVolumeClaim
```
Table 10-31 Key parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>volume.beta.kubernetes.io/storage-class</td>
<td>Bucket type. Currently, <strong>obs-standard</strong> and <strong>obs-standard-ia</strong> are supported.</td>
</tr>
<tr>
<td>name</td>
<td>Name of the PVC to be created.</td>
</tr>
<tr>
<td>accessModes</td>
<td>Only <strong>ReadWriteMany</strong> is supported. <strong>ReadWriteOnly</strong> is not supported.</td>
</tr>
<tr>
<td>storage</td>
<td>Storage capacity in the unit of Gi. For OBS buckets, this field is used only for verification (cannot be empty or 0). Its value is fixed at 1, and any value you set does not take effect for OBS buckets.</td>
</tr>
</tbody>
</table>

**Step 3** Run the following command to create the PVC.

```shell
kubectl create -f pvc-obs-auto-example.yaml
```

After the command is executed, an OBS bucket is created in the VPC to which the cluster belongs. You can click the bucket name in **Storage > OBS** to view the bucket or view it on the OBS console.

---

### 10.5.4 (kubectl) Creating a PV from an Existing OBS Bucket

#### Scenario

CCE allows you to use an existing OBS bucket to create a PersistentVolume (PV). You can create a PersistentVolumeClaim (PVC) and bind it to the PV.

#### Prerequisites

You have created a CCE cluster and installed the FlexVolume plug-in (**storage-driver**) in the cluster.
Notes and Constraints

The following configuration example applies to clusters of Kubernetes 1.13 or earlier.

Procedure

**Step 1** Log in to the OBS console, create an OBS bucket, and record the bucket name and storage class.

**Step 2** Use kubectl to connect to the cluster. For details, see [Connecting to a Cluster Using kubectl](#).

**Step 3** Create two YAML files for creating the PV and PVC. Assume that the file names are `pv-obs-example.yaml` and `pvc-obs-example.yaml`.

```
touch pv-obs-example.yaml pvc-obs-example.yaml
```

<table>
<thead>
<tr>
<th>Kubernetes Version</th>
<th>Description</th>
<th>YAML Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.11 ≤ K8s version ≤ 1.13</td>
<td>Clusters from v1.11 to v1.13</td>
<td>Example YAML</td>
</tr>
<tr>
<td>K8s version = 1.9</td>
<td>Clusters of v1.9</td>
<td>Example YAML</td>
</tr>
</tbody>
</table>

Clusters from v1.11 to v1.13

- **Example YAML file for the PV:**

  ```yaml
  apiVersion: v1
  kind: PersistentVolume
  metadata:
    name: pv-obs-example
  annotations:
    pv.kubernetes.io/provisioned-by: flexvolume-huawei.com/fuxiobs
  spec:
    accessModes:
    - ReadWriteMany
    capacity:
      storage: 1Gi
    claimRef:
      apiVersion: v1
      kind: PersistentVolumeClaim
      name: pvc-obs-example
      namespace: default
    flexVolume:
      driver: huawei.com/fuxiobs
      fsType: obs
    options:
      fsType: obs
      region: ap-southeast-1
      storage_class: STANDARD
      volumeID: test-obs
      persistentVolumeReclaimPolicy: Delete
      storageClassName: obs-standard
  ```
Table 10-32 Key parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>driver</td>
<td>Storage driver used to mount the volume. Set the driver to <strong>huawei.com/fuxiobs</strong> for the OBS volume.</td>
</tr>
<tr>
<td>storage_class</td>
<td>Storage class, including <strong>STANDARD</strong> (standard bucket) and <strong>STANDARD_IA</strong> (infrequent access bucket).</td>
</tr>
<tr>
<td>region</td>
<td>For details about the value of <strong>region</strong>, see <strong>Regions and Endpoints</strong>.</td>
</tr>
<tr>
<td>volumeID</td>
<td>OBS bucket name. To obtain the name, log in to the CCE console, choose <strong>Resource Management &gt; Storage</strong>, click the PVC name in the <strong>OBS</strong> tab page, and copy the PV name on the <strong>PV Details</strong> tab page.</td>
</tr>
<tr>
<td>storage</td>
<td>Storage capacity in the unit of Gi. The value is fixed at 1Gi.</td>
</tr>
<tr>
<td>storageClassName</td>
<td>Storage class supported by OBS, including <strong>obs-standard</strong> (standard bucket) and <strong>obs-standard-ia</strong> (infrequent access bucket).</td>
</tr>
<tr>
<td>spec.claimRef.apiVersion</td>
<td>The value is fixed at <strong>v1</strong>.</td>
</tr>
<tr>
<td>spec.claimRef.kind</td>
<td>The value is fixed at <strong>PersistentVolumeClaim</strong>.</td>
</tr>
<tr>
<td>spec.claimRef.name</td>
<td>The value is the same as the name of the PVC created in the next step.</td>
</tr>
<tr>
<td>spec.claimRef.namespace</td>
<td>The value is the same as the namespace of the PVC created in the next step.</td>
</tr>
</tbody>
</table>

- **Example YAML file for the PVC:**

```yaml
apiVersion: v1
description:
  annotations:
    volume.beta.kubernetes.io/storage-class: obs-standard
    volume.beta.kubernetes.io/storage-provisioner: flexvolume-huawei.com/fuxiobs
  name: pvc-obs-example
  namespace: default
description:
  accessModes:
  - ReadWriteMany
description:
  requests:
    storage: 1Gi
description:
  volumeName: pv-obs-example
```
### Table 10-33 Key parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>volume.beta.kubernetes.io/storage-class</td>
<td>Storage class supported by OBS, including <strong>obs-standard</strong> and <strong>obs-standard-ia</strong>.</td>
</tr>
<tr>
<td>volume.beta.kubernetes.io/storage-provisioner</td>
<td>Must be set to <strong>flexvolume-huawei.com/fuxiobs</strong>.</td>
</tr>
<tr>
<td>volumeName</td>
<td>Name of the PV.</td>
</tr>
<tr>
<td>storage</td>
<td>Storage capacity in the unit of Gi. The value is fixed at <strong>1Gi</strong>.</td>
</tr>
</tbody>
</table>

### Clusters of v1.9

- **Example YAML file for the PV:**

  ```yaml
  apiVersion: v1
  kind: PersistentVolume
  metadata:
    name: pv-obs-example
    namespace: default
  spec:
    accessModes:
      - ReadWriteMany
    capacity:
      storage: 1Gi
    flexVolume:
      driver: huawei.com/fuxiobs
      fsType: obs
      options:
        fsType: obs
        kubernetes.io/namespace: default
        region: ap-southeast-1
        storage_class: STANDARD
        volumeID: test-obs
    persistentVolumeReclaimPolicy: Delete
    storageClassName: obs-standard
  ```

### Table 10-34 Key parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>driver</td>
<td>Storage driver used to mount the volume. Set the driver to <strong>huawei.com/fuxiobs</strong> for the OBS volume.</td>
</tr>
<tr>
<td>storage_class</td>
<td>Storage class, including <strong>STANDARD</strong> (standard bucket) and <strong>STANDARD_IA</strong> (infrequent access bucket).</td>
</tr>
<tr>
<td>region</td>
<td>For details about the value of <strong>region</strong>, see <strong>Regions and Endpoints</strong>.</td>
</tr>
<tr>
<td>volumeID</td>
<td>OBS bucket name. To obtain the name, log in to the CCE console, choose <strong>Resource Management &gt; Storage</strong>, click the PVC name in the <strong>OBS</strong> tab page, and copy the PV name on the <strong>PV Details</strong> tab page.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>storage</td>
<td>Storage capacity in the unit of Gi. The value is fixed at 1Gi.</td>
</tr>
<tr>
<td>storageClassName</td>
<td>Storage class supported by OBS, including obs-standard (standard bucket) and obs-standard-ia (infrequent access bucket).</td>
</tr>
</tbody>
</table>

- **Example YAML file for the PVC:**

```yaml
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  annotations:
  volume.beta.kubernetes.io/storage-class: obs-standard
  volume.beta.kubernetes.io/storage-provisioner: flexvolume-huawei.com/fuxiobs
  name: pvc-obs-example
  namespace: default
spec:
  accessModes:
  - ReadWriteMany
  resources:
    requests:
      storage: 1Gi
  volumeName: pv-obs-example
  volumeNamespace: default
```

**Table 10-35 Key parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>volume.beta.kubernetes.io/storage-class</td>
<td>Storage class supported by OBS, including obs-standard and obs-standard-ia.</td>
</tr>
<tr>
<td>volume.beta.kubernetes.io/storage-provisioner</td>
<td>Must be set to flexvolume-huawei.com/fuxiobs.</td>
</tr>
<tr>
<td>volumeName</td>
<td>Name of the PV.</td>
</tr>
<tr>
<td>storage</td>
<td>Storage capacity in the unit of Gi. The value is fixed at 1Gi.</td>
</tr>
</tbody>
</table>

**Step 4**  Create the PV.

```
kubectl create -f pv-obs-example.yaml
```

**Step 5**  Create the PVC.

```
kubectl create -f pvc-obs-example.yaml
```

---End
10.5.5 (kubectl) Creating a Deployment Mounted with an OBS Volume

Scenario

After an OBS volume is created or imported to CCE, you can mount the volume to a workload.

Prerequisites

You have created a CCE cluster and installed the FlexVolume plug-in (`storage-driver`) in the cluster.

Notes and Constraints

The following configuration example applies to clusters of Kubernetes 1.13 or earlier.

Procedure

**Step 1** Use `kubectl` to connect to the cluster. For details, see Connecting to a Cluster Using `kubectl`.

**Step 2** Run the following commands to configure the `obs-deployment-example.yaml` file, which is used to create a pod.

```bash
touch obs-deployment-example.yaml
vi obs-deployment-example.yaml
```

Example of mounting an OBS volume to a Deployment (PVC-based, shared volume):

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: obs-deployment-example                       # Workload name
  namespace: default
spec:
  replicas: 1
  selector:
    matchLabels:
      app: obs-deployment-example
  template:
    metadata:
      labels:
        app: obs-deployment-example
    spec:
      containers:
        - image: nginx
          name: container-0
          volumeMounts:
            - mountPath: /tmp                       # Mount path
              name: pvc-obs-example
              # Workload name
            - mountPath: /data
              name: pvc-obs-auto-example
              # Mount path
          restartPolicy: Always
          imagePullSecrets:
            - name: default-secret
          persistentVolumeClaim:
            claimName: pvc-obs-auto-example       # PVC name
```

Cloud Container Engine
User Guide
Issue 01 (2021-09-08) Copyright © Huawei Technologies Co., Ltd. 508
Table 10-36 Key parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Name of the pod to be created.</td>
</tr>
<tr>
<td>app</td>
<td>Name of the application running in the pod.</td>
</tr>
<tr>
<td>mountPath</td>
<td>Mount path in the container.</td>
</tr>
</tbody>
</table>

**NOTE**

spec.template.spec.containers.volumeMounts.name and spec.template.spec.volumes.name must be consistent because they have a mapping relationship.

Example of mounting an OBS volume to a StatefulSet (PVC template-based, dedicated volume):

**Example YAML:**

```yaml
apiVersion: apps/v1
group: StatefulSet
metadata:
  name: deploy-obs-standard-in
  namespace: default
  generation: 1
  labels:
    appgroup: ""
spec:
  replicas: 1
  selector:
    matchLabels:
      app: deploy-obs-standard-in
template:
  metadata:
    labels:
      app: deploy-obs-standard-in
  annotations:
    metrics.alpha.kubernetes.io/custom-endpoints: '[[{"api":"","path":"","port":","names":""}]]'
pod.alpha.kubernetes.io/initialized: 'true'
spec:
  containers:
    - name: container-0
      image: 'nginx:1.12-alpine-perl'
      env:
        - name: PAAS_APP_NAME
          value: deploy-obs-standard-in
        - name: PAAS_NAMESPACE
          value: default
        - name: PAAS_PROJECT_ID
          value: a2cd8e998dca42e98a41f596c636dbda
      resources: {}
      volumeMounts:
        - name: obs-bs-standard-mountoptionpvc
          mountPath: /tmp
          terminationMessagePath: /dev/termination-log
          terminationMessagePolicy: File
          imagePullPolicy: IfNotPresent
          restartPolicy: Always
          terminationGracePeriodSeconds: 30
doPolicy: ClusterFirst
      securityContext: {}
      imagePullSecrets: []
      - name: default-secret
```

Cloud Container Engine
User Guide
10 Storage (FlexVolume)

Issue 01 (2021-09-08) Copyright © Huawei Technologies Co., Ltd. 509
Table 10-37 Key parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Name of the created workload.</td>
</tr>
<tr>
<td>image</td>
<td>Image of the workload.</td>
</tr>
<tr>
<td>mountPath</td>
<td>Mount path in the container. In this example, the volume</td>
</tr>
<tr>
<td></td>
<td>is mounted to the /tmp directory.</td>
</tr>
<tr>
<td>serviceName</td>
<td>Service corresponding to the workload. For details about how to create a</td>
</tr>
<tr>
<td></td>
<td>Service, see Creating a StatefulSet.</td>
</tr>
</tbody>
</table>

**NOTE**

spec.template.spec.containers.volumeMounts.name and spec.volumeClaimTemplates.metadata.name must be consistent because they have a mapping relationship.

**Step 3** Run the following command to create the pod:

```
kubectl create -f obs-deployment-example.yaml
```

After the creation is complete, choose Storage > OBS on the CCE console and click the PVC name. On the PVC details page, you can view the binding relationship between the OBS service and the PVC.

----End

10.5.6 (kubectl) Creating a StatefulSet Mounted with an OBS Volume

**Scenario**

CCE allows you to use an existing OBS volume to create a StatefulSet through a PersistentVolumeClaim (PVC).
Prerequisites

You have created a CCE cluster and installed the FlexVolume plug-in (storage-driver) in the cluster.

Notes and Constraints

The following configuration example applies to clusters of Kubernetes 1.13 or earlier.

Procedure

**Step 1** Create an OBS volume by referring to Creating an OBS Volume and obtain the PVC name.

**Step 2** Use kubectl to connect to the cluster. For details, see Connecting to a Cluster Using kubectl.

**Step 3** Create a YAML file for creating the workload. Assume that the file name is obs-statefulset-example.yaml.

```
touch obs-statefulset-example.yaml
vi obs-statefulset-example.yaml
```

**Example YAML:**

```yaml
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: obs-statefulset-example
  namespace: default
spec:
  replicas: 1
  selector:
    matchLabels:
      app: obs-statefulset-example
  serviceName: qwqq
  template:
    metadata:
      annotations:
        metrics.alpha.kubernetes.io/custom-endpoints: '[["api":"","path":"","port":"","names":"]']
        pod.alpha.kubernetes.io/initialized: "true"
    creationTimestamp: null
    labels:
      app: obs-statefulset-example
    spec:
      affinity: {}
      containers:
        - image: nginx:latest
          imagePullPolicy: Always
          name: container-0
          volumeMounts:
            - mountPath: /tmp
              name: pvc-obs-example
          imagePullSecrets:
            - name: default-secret
          volumes:
            - name: pvc-obs-example
              persistentVolumeClaim:
                claimName: cce-obs-demo
```
### Table 10-38 Key parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>replicas</td>
<td>Number of pods.</td>
</tr>
<tr>
<td>name</td>
<td>Name of the created workload.</td>
</tr>
<tr>
<td>image</td>
<td>Image used by the workload.</td>
</tr>
<tr>
<td>mountPath</td>
<td>Mount path in the container.</td>
</tr>
<tr>
<td>serviceName</td>
<td>Service corresponding to the workload. For details about how to create a Service, see <a href="#">Creating a StatefulSet</a>.</td>
</tr>
<tr>
<td>claimName</td>
<td>Name of an existing PVC.</td>
</tr>
</tbody>
</table>

**Step 4** Create the StatefulSet.

```bash
kubectl create -f obs-statefulset-example.yaml
```

---End

## 10.6 Using SFS File Systems as Storage Volumes

### 10.6.1 Overview

CCE allows you to mount a volume created from a Scalable File Service (SFS) file system to a container to store data persistently. SFS volumes are commonly used in ReadWriteMany scenarios, such as media processing, content management, big data analysis, and workload process analysis.

**Figure 10-10** Mounting SFS volumes to CCE

In the same VPC network

Cluster

Node

Node

Cluster

Node

Node

shares network file system

File Storage
Description

- **Standard file protocols**: You can mount file systems as volumes to servers, the same as using local directories.
- **Data sharing**: The same file system can be mounted to multiple servers, so that data can be shared.
- **Private network**: User can access data only in private networks of data centers.
- **Capacity and performance**: The capacity of a single file system is high (PB level) and the performance is excellent (ms-level read/write I/O latency).
- **Use cases**: Deployments/StatefulSets in the ReadWriteMany mode and jobs created for high-performance computing (HPC), media processing, content management, web services, big data analysis, and workload process analysis.

For details, see [SFS Service Overview](#).

### 10.6.2 Using SFS Volumes

**Prerequisites**

You have created a CCE cluster and installed the FlexVolume plug-in ([storage-driver](#)) in the cluster.

**Notes and Constraints**

- Container storage in CCE clusters of Kubernetes 1.13 or later version supports encryption. Currently, E2E encryption is supported only in certain regions.
- Volumes cannot be created in specified enterprise projects. Only the default enterprise project is supported.

**Creating an SFS Volume**

**Step 1** Log in to the CCE console. In the navigation pane, choose Resource Management > Storage.

**Step 2** On the SFS tab, click Create SFS File System.

**Step 3** Configure basic information, as shown in **Table 10-39**.

**Table 10-39 Parameters for Creating a File System Volume**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Parameter Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>* PVC Name</td>
<td>Name of the new PVC, which is different from the volume name. The actual volume name is automatically generated when the PV is created by the PVC.</td>
</tr>
<tr>
<td>Cluster Name</td>
<td>Cluster to which the file system volume belongs.</td>
</tr>
<tr>
<td>Namespace</td>
<td>Namespace with which the snapshot is associated.</td>
</tr>
<tr>
<td>Total Capacity</td>
<td>The total capacity is the capacity of a single volume. Fees are charged by actual usage.</td>
</tr>
</tbody>
</table>
## Adding an SFS Volume

### Step 1
Create a workload or job by referring to Creating a Deployment, Creating a StatefulSet, Creating a DaemonSet, or Creating a Job. During creation, expand Data Storage after adding a container. On the Cloud Volume tab page, click Add Cloud Volume.

### Step 2
Set the storage class to SFS.

### Step 4
Click Create.

The volume is displayed in the list. When PVS Status becomes Bound, the volume is created successfully.

### Step 5
Click the volume name to view detailed information about the volume.
### Table 10-40 Parameters for mounting a file system

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Parameter Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Type</strong></td>
<td><strong>File Storage (NFS):</strong> This type applies to a wide range of scenarios, including media processing, content management, big data, and application analysis.</td>
</tr>
</tbody>
</table>

**Allocation Mode**

| Manual | • **Name:** Select a created file system. You need to create a file system in advance. For details about how to create a file system, see [Creating an SFS Volume](#).  
• **Sub-Type:** subtype of the created file storage.  
• **Storage Capacity:** This field is one of the PVC attributes. If the storage capacity has been expanded on the IaaS side, it is normal that the capacity values are inconsistent. The PVC capacity is the same as the storage entity capacity only after end-to-end container storage capacity expansion is supported for CCE clusters of v1.13.  
| Automatic | An SFS volume is created automatically. You need to enter the storage capacity.  
• **Sub-Type:** Select **NFS**.  
• **Storage Capacity:** Specify the total storage capacity, in GB. Ensure that the storage capacity quota is not exceeded; otherwise, creation will fail.  
• **Storage Format:** The default value is **CSI**. The container storage interface (CSI) is used to establish a set of standard storage management interfaces between Kubernetes and external storage systems to provide storage services for containers.  
• After you select **KMS Encryption**, Key Management Service (KMS), an easy-to-use and highly secure service, will be enabled for file systems. This function is supported only for clusters of v1.13.10 and later in certain regions. If no agency has been created, click [Create Agency](#) and set the following parameters:  
  – **Agency Name:** Agencies can be used to assign permissions to trusted accounts or cloud services for a specific period of time. If no agency is created, click [Create Agency](#). The agency name **SFSAccessKMS** indicates that SFS is granted the permission to access KMS. After SFS is authorized successfully, it can obtain KMS keys to encrypt and decrypt file systems.  
  – **Key Name:** After a key is created, it can be loaded and used in containerized applications. For details on how to create a key, see [Creating a CMK](#).  
  – **Key ID:** generated by default. |
Configure the following parameters:

1. **subPath**: Enter the subpath of the file storage, for example, `/tmp`. If this parameter is not specified, the root path of the data volume is used by default. Currently, only file storage is supported. The value must be a relative path and cannot start with a slash (/) or `../`.

2. **Container Path**: Enter the path of the container, for example, `/tmp`. The container path must not be a system directory, such as `/` and `/var/run`. Otherwise, an exception occurs. You are advised to mount the volume to an empty directory. If the directory is not empty, ensure that there are no files affecting container startup in the directory. Otherwise, such files will be replaced, resulting in failures to start the container and create the workload.

   **NOTICE**
   If the volume is mounted to a high-risk directory, you are advised to use an account with minimum permissions to start the container; otherwise, high-risk files on the host machine may be damaged.

3. **Set permissions**.
   - **Read-only**: You can only read the data volumes mounted to the path.
   - **Read/Write**: You can modify the data volumes mounted to the path. Newly written data is not migrated if the container is migrated, which may cause a data loss.

Click **Add Container Path** to add multiple settings. Then, click **OK**.

--- End

### Importing an SFS Volume

CCE allows you to import existing SFS volumes.

**Step 1** Log in to the CCE console. In the navigation pane, choose **Resource Management > Storage**. On the **SFS** tab page, click **Import**.

**Step 2** Select one or more SFS volumes that you want to attach.

**Step 3** Select the target cluster and namespace. Then, click **OK**.

--- End

### Unbinding an SFS Volume

When an SFS volume is successfully created or imported, the volume is automatically bound to the current cluster. Other clusters can also use the volume.
When the SFS volume is unbound from the cluster, other clusters can still import and use the volume.

If the SFS volume has been attached to a workload, the volume cannot be unbound from the cluster.

**Step 1** Log in to the CCE console. In the navigation pane, choose Resource Management > Storage. In the SFS volume list, click Unbind next to the target volume.

**Step 2** Confirm the unbinding, and click OK.

----End

**Related Operations**

After an SFS volume is created, you can perform the operation described in Table 10-41.

**Table 10-41 Other operations**

<table>
<thead>
<tr>
<th>Operation</th>
<th>Description</th>
</tr>
</thead>
</table>
| Deleting an SFS volume     | 1. Select the SFS volume to be deleted and click **Delete** in the **Operation** column.  
                              | 2. Follow the prompts to delete the EVS disk.                                                                                                 |
| Importing an SFS volume    | CCE allows you to import existing SFS volumes.  
                              | 1. On the **SFS** tab page, click **Import**.  
                              | 2. Select one or more SFS volumes that you want to attach.  
                              | 3. Select the target cluster and namespace.  
                              | 4. Click **Yes**.                                                                                                                               |

**10.6.3 (kubectl) Automatically Creating an SFS Volume**

**Scenario**

CCE supports creating SFS volumes through PersistentVolumeClaims (PVCs).

**Prerequisites**

You have created a CCE cluster and installed the FlexVolume plug-in (storage-driver) in the cluster.

**Notes and Constraints**

The following configuration example applies to clusters of Kubernetes 1.13 or earlier.
Procedure

Step 1  Use kubectl to connect to the cluster. For details, see Connecting to a Cluster Using kubectl.

Step 2  Run the following commands to configure the `pvc-sfs-auto-example.yaml` file, which is used to create a PVC.

```bash
  touch pvc-sfs-auto-example.yaml
  vi pvc-sfs-auto-example.yaml
```

**Example YAML file:**
```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  annotations:
    volume.beta.kubernetes.io/storage-class: nfs-rw
  name: pvc-sfs-auto-example
  namespace: default
spec:
  accessModes:
  - ReadWriteMany
  resources:
    requests:
      storage: 10Gi
```

**Table 10-42 Key parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>volume.beta.kubernetes.io/storage-class</code></td>
<td>File storage class. Currently, the standard file protocol type (nfs-rw) is supported.</td>
</tr>
<tr>
<td><code>name</code></td>
<td>Name of the PVC to be created.</td>
</tr>
<tr>
<td><code>accessModes</code></td>
<td>Only <strong>ReadWriteMany</strong> is supported. <strong>ReadWriteOnly</strong> is not supported.</td>
</tr>
<tr>
<td><code>storage</code></td>
<td>Storage capacity in the unit of Gi.</td>
</tr>
</tbody>
</table>

Step 3  Run the following command to create the PVC.

```bash
  kubectl create -f pvc-sfs-auto-example.yaml
```

After the command is executed, a file system is created in the VPC to which the cluster belongs. Choose **Storage > SFS** on the CCE console or log in to the SFS console to view the file system.

----End

## 10.6.4 (kubectl) Creating a PV from an Existing SFS File System

**Scenario**

CCE allows you to use an existing file system to create a PersistentVolume (PV). After the creation is successful, create the corresponding PersistentVolumeClaim (PVC) and bind it to the PV.
Prerequisites

You have created a CCE cluster and installed the FlexVolume plug-in (storage-driver) in the cluster.

Notes and Constraints

The following configuration example applies to clusters of Kubernetes 1.13 or earlier.

Procedure

Step 1  Log in to the SFS console, create a file system, and record the file system ID, shared path, and capacity.

Step 2  Use kubectl to connect to the cluster. For details, see Connecting to a Cluster Using kubectl.

Step 3  Create two YAML files for creating the PV and PVC. Assume that the file names are pv-sfs-example.yaml and pvc-sfs-example.yaml.

```
touch pv-sfs-example.yaml  pvc-sfs-example.yaml
```

<table>
<thead>
<tr>
<th>Kubernetes Version</th>
<th>Description</th>
<th>YAML Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.11 ≤ K8s version ≤ 1.13</td>
<td>Clusters from v1.11 to v1.13</td>
<td>Example YAML</td>
</tr>
<tr>
<td>K8s version = 1.9</td>
<td>Clusters of v1.9</td>
<td>Example YAML</td>
</tr>
</tbody>
</table>

Clusters from v1.11 to v1.13

- **Example YAML file for the PV:**

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: pv-sfs-example
  annotations:
    pv.kubernetes.io/provisioned-by: flexvolume-huawei.com/fuxinfs
spec:
  accessModes:
    - ReadWriteMany
  capacity:
    storage: 10Gi
  claimRef:
    apiVersion: v1
    kind: PersistentVolumeClaim
    name: pvc-sfs-example
    namespace: default
  flexVolume:
    driver: huawei.com/fuxinfs
    fsType: nfs
    options:
      deviceMountPath: sfs-nas1.southchina.huaweicloud.com:/share-73bdfafd
      fsType: nfs
      volumeID: f6976f9e-2493-419b-97ca-d7816008d91c
    persistentVolumeReclaimPolicy: Delete
    storageClassName: nfs-rw
```
Table 10-43 Key parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>driver</td>
<td>Storage driver used to mount the volume. Set the driver to <code>huawei.com/fuxinfs</code> for the file system.</td>
</tr>
<tr>
<td>deviceMountPath</td>
<td>Shared path of the file system. On the management console, choose <strong>Service List &gt; Storage &gt; Scalable File Service</strong>. You can obtain the shared path of the file system from the <strong>Mount Address</strong> column, as shown in <strong>Figure 10-11</strong>.</td>
</tr>
<tr>
<td>volumeID</td>
<td>File system ID. To obtain the ID, log in to the CCE console, choose <strong>Resource Management &gt; Storage</strong>, click the PVC name in the SFS tab page, and copy the PVC ID on the PVC details page.</td>
</tr>
<tr>
<td>storage</td>
<td>File system size.</td>
</tr>
<tr>
<td>storageClassName</td>
<td>Read/write mode supported by the file system. Currently, <code>nfs-rw</code> and <code>nfs-ro</code> are supported.</td>
</tr>
<tr>
<td>spec.claimRef.apiVersion</td>
<td>The value is fixed at <code>v1</code>.</td>
</tr>
<tr>
<td>spec.claimRef.kind</td>
<td>The value is fixed at <code>PersistentVolumeClaim</code>.</td>
</tr>
<tr>
<td>spec.claimRef.name</td>
<td>The value is the same as the name of the PVC created in the next step.</td>
</tr>
<tr>
<td>spec.claimRef.namespace</td>
<td>Namespace of the PVC. The value is the same as the namespace of the PVC created in the next step.</td>
</tr>
</tbody>
</table>

**Example YAML file for the PVC:**
```yaml
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  annotations:
    volume.beta.kubernetes.io/storage-class: nfs-rw
    volume.beta.kubernetes.io/storage-provisioner: flexvolume-huawei.com/fuxinfs
name: pvc-sfs-example
namespace: default
spec:
  accessModes:
  - ReadWriteMany
  resources:
    requests:
      storage: 10Gi
volumeName: pv-sfs-example
```

Cloud Container Engine
User Guide
10 Storage (FlexVolume)

Issue 01 (2021-09-08) Copyright © Huawei Technologies Co., Ltd. 520
Table 10-44 Key parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>volume.beta.kubernetes.io/storage-class</td>
<td>Read/write mode supported by the file system. <strong>nfs-rw</strong> and <strong>nfs-ro</strong> are supported. The value must be the same as that of the existing PV.</td>
</tr>
<tr>
<td>volume.beta.kubernetes.io/storage-provisioner</td>
<td>Must be set to <strong>flexvolume-huawei.com/fuxins</strong>.</td>
</tr>
<tr>
<td>storage</td>
<td>Storage capacity, in the unit of Gi. The value must be the same as the storage size of the existing PV.</td>
</tr>
<tr>
<td>volumeName</td>
<td>Name of the PV.</td>
</tr>
</tbody>
</table>

Clusters of v1.9

- **Example YAML file for the PV:**

```yaml
apiVersion: v1
kind: PersistentVolume
metadata:
  name: pv-sfs-example
  namespace: default
spec:
  accessModes:
    - ReadWriteMany
  capacity:
    storage: 10Gi
  flexVolume:
    driver: huawei.com/fuxins
    fsType: nfs
    options:
      deviceMountPath: sfs-nas1.southchina.huaweicloud.com:/share-73bdfaf
      fsType: nfs
      kubernetes.io/namespace: default
      volumeID: f6976f9e-2493-419b-97ca-d7816008d91c
    persistentVolumeReclaimPolicy: Delete
    storageClassName: nfs-rw
```

Table 10-45 Key parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>driver</td>
<td>Storage driver used to mount the volume. Set the driver to <strong>huawei.com/fuxins</strong> for the file system.</td>
</tr>
<tr>
<td>deviceMountPath</td>
<td>Shared path of the file system. On the management console, choose <strong>Service List &gt; Storage &gt; Scalable File Service</strong>. You can obtain the shared path of the file system from the <strong>Mount Address</strong> column, as shown in <strong>Figure 10-11</strong>.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>volumeID</td>
<td>File system ID. To obtain the ID, log in to the CCE console, choose Resource Management &gt; Storage, click the PVC name in the SFS tab page, and copy the PVC ID on the PVC details page.</td>
</tr>
<tr>
<td>storage</td>
<td>File system size.</td>
</tr>
<tr>
<td>storageClassName</td>
<td>Read/write mode supported by the file system. Currently, nfs-rw and nfs-ro are supported.</td>
</tr>
</tbody>
</table>

**Example YAML file for the PVC:**

```yaml
apiVersion: v1
category: PersistentVolumeClaim
metadata:
  annotations:
    - name: volume.beta.kubernetes.io/storage-class
      value: nfs-rw
    - name: volume.beta.kubernetes.io/storage-provisioner
      value: flexvolume-huawei.com/fuxins
  name: pvc-sfs-example
  namespace: default
spec:
  accessModes:
    - ReadWriteMany
  resources:
    requests:
      - resource: storage
        quantity: 10Gi
  volumeName: pv-sfs-example
  volumeNamespace: default
```

**Table 10-46 Key parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>volume.beta.kubernetes.io/storage-class</td>
<td>Read/write mode supported by the file system. nfs-rw and nfs-ro are supported. The value must be the same as that of the existing PV.</td>
</tr>
<tr>
<td>volume.beta.kubernetes.io/storage-provisioner</td>
<td>The field must be set to flexvolume-huawei.com/fuxins.</td>
</tr>
<tr>
<td>storage</td>
<td>Storage capacity, in the unit of Gi. The value must be the same as the storage size of the existing PV.</td>
</tr>
<tr>
<td>volumeName</td>
<td>Name of the PV.</td>
</tr>
</tbody>
</table>
Figure 10-11 SFS - file system mount address

NOTE

The VPC to which the file system belongs must be the same as the VPC of the ECS VM to which the workload is planned.

Step 4 Create the PV.

```
kubectl create -f pv-sfs-example.yaml
```

Step 5 Create the PVC.

```
kubectl create -f pvc-sfs-example.yaml
```

---End

10.6.5 (kubectl) Creating a Deployment Mounted with an SFS Volume

Scenario

After an SFS volume is created or imported to CCE, you can mount the volume to a workload.

Prerequisites

You have created a CCE cluster and installed the FlexVolume plug-in (`storage-driver`) in the cluster.

Notes and Constraints

The following configuration example applies to clusters of Kubernetes 1.13 or earlier.

Procedure

Step 1 Use kubectl to connect to the cluster. For details, see Connecting to a Cluster Using kubectl.
Step 2 Run the following commands to configure the `sfs-deployment-example.yaml` file, which is used to create a pod.

```
touch sfs-deployment-example.yaml
vi sfs-deployment-example.yaml
```

Example of mounting an SFS volume to a Deployment (PVC-based, shared volume):

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: sfs-deployment-example  # Workload name
  namespace: default
spec:
  replicas: 1
  selector:
    matchLabels:
      app: sfs-deployment-example
template:
  metadata:
    labels:
      app: sfs-deployment-example
  spec:
    containers:
      - image: nginx
        name: container-0
        volumeMounts:
          - mountPath: /tmp  # Mount path
            name: pvc-sfs-example  # PVC name
        imagePullSecrets:
          - name: default-secret
        restartPolicy: Always
        volumes:
          - name: pvc-sfs-example
            persistentVolumeClaim:
              claimName: pvc-sfs-auto-example  # PVC name
```

<table>
<thead>
<tr>
<th>Parent Parameter</th>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>metadata</td>
<td>name</td>
<td>Name of the pod to be created.</td>
</tr>
<tr>
<td>spec.template.spec.containers.volumeMounts</td>
<td>mountPath</td>
<td>Mount path in the container. In this example, the mount path is <code>/tmp</code>.</td>
</tr>
<tr>
<td>spec.template.spec.volumes.persistentVolumeClaim</td>
<td>claimName</td>
<td>Name of an existing PVC.</td>
</tr>
</tbody>
</table>

**NOTE**

```
spec.template.spec.containers.volumeMounts.name
spec.template.spec.volumes.name
```

must be consistent because they have a mapping relationship.

Example of mounting an SFS volume to a StatefulSet (PVC template-based, dedicated volume):

Example YAML:
```yaml
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: deploy-sfs-nfs-rw-in
  namespace: default
labels:
  appgroup: 
spec:
  replicas: 2
selector:
  matchLabels:
    app: deploy-sfs-nfs-rw-in
template:
  metadata:
    labels:
      app: deploy-sfs-nfs-rw-in
  spec:
    containers:
    - name: container-0
      image: 'nginx:1.12-alpine-perl'
      volumeMounts:
      - name: bs-nfs-rw-mountoptionpvc
        mountPath: /aaa
      imagePullSecrets:
      - name: default-secret
    volumeClaimTemplates:
    - metadata:
      annotations:
        volume.beta.kubernetes.io/storage-class: nfs-rw
        volume.beta.kubernetes.io/storage-provisioner: flexvolume-huawei.com/fuxinfs
      spec:
        accessModes:
        - ReadWriteMany
        resources:
          requests:
            storage: 1Gi
        serviceName: wwww
```

### Table 10-48 Key parameters

<table>
<thead>
<tr>
<th>Parent Parameter</th>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>metadata</td>
<td>name</td>
<td>Name of the created workload.</td>
</tr>
<tr>
<td>spec.template.spec.containers</td>
<td>image</td>
<td>Image of the workload.</td>
</tr>
<tr>
<td>spec.template.spec.containers.volumeMount</td>
<td>mountPath</td>
<td>Mount path in the container. In this example, the mount path is <code>/tmp</code>.</td>
</tr>
<tr>
<td>spec</td>
<td>serviceName</td>
<td>Service corresponding to the workload. For details about how to create a Service, see Creating a StatefulSet.</td>
</tr>
</tbody>
</table>

**NOTE**

`spec.template.spec.containers.volumeMounts.name` and `spec.volumeClaimTemplates.metadata.name` must be consistent because they have a mapping relationship.

**Step 3** Run the following command to create the pod:
kubectl create -f sfs-deployment-example.yaml

After the creation is complete, log in to the CCE console. In the navigation pane, choose Resource Management > Storage > SFS. Click the PVC name. On the PVC details page, you can view the binding relationship between SFS and PVC.

---End

10.6.6 (kubectl) Creating a StatefulSet Mounted with an SFS Volume

Scenario

CCE allows you to use an existing SFS volume to create a StatefulSet through a PersistentVolumeClaim (PVC).

Prerequisites

You have created a CCE cluster and installed the FlexVolume plug-in (storage-driver) in the cluster.

Notes and Constraints

The following configuration example applies to clusters of Kubernetes 1.13 or earlier.

Procedure

Step 1 Create an SFS volume by referring to Creating an SFS Volume and record the volume name.

Step 2 Use kubectl to connect to the cluster. For details, see Connecting to a Cluster Using kubectl.

Step 3 Create a YAML file for creating the workload. Assume that the file name is sfs-statefulset-example.yaml.

touch sfs-statefulset-example.yaml

vi sfs-statefulset-example.yaml

Example YAML:

```yaml
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: sfs-statefulset-example
  namespace: default
spec:
  replicas: 2
  selector:
    matchLabels:
      app: sfs-statefulset-example
  serviceName: qwqq
  template:
    metadata:
      annotations:
        metrics.alpha.kubernetes.io/custom-endpoints: '[["api":"","path":"","port":"","names":""}}]
        pod.alpha.kubernetes.io/initialized: "true"
```
labels:
  - app: sfs-statefulset-example
spec:
  affinity: {}
containers:
  - image: nginx:latest
    name: container-0
    volumeMounts:
    - mountPath: /tmp
      name: pvc-sfs-example
imagePullSecrets:
  - name: default-secret
volumes:
  - name: pvc-sfs-example
    persistentVolumeClaim:
      claimName: cce-sfs-demo

<table>
<thead>
<tr>
<th>Parent Parameter</th>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>spec</td>
<td>replicas</td>
<td>Number of pods.</td>
</tr>
<tr>
<td>metadata</td>
<td>name</td>
<td>Name of the created workload.</td>
</tr>
<tr>
<td>spec.template.spec.containers</td>
<td>image</td>
<td>Image used by the workload.</td>
</tr>
<tr>
<td>spec.template.spec.containers.volumeMounts</td>
<td>mountPath</td>
<td>Mount path in the container.</td>
</tr>
<tr>
<td>spec</td>
<td>serviceName</td>
<td>Service corresponding to the workload. For details about how to create a Service, see Creating a StatefulSet.</td>
</tr>
<tr>
<td>spec.template.spec.persistentVolumeClaim</td>
<td>claimName</td>
<td>Name of an existing PVC.</td>
</tr>
</tbody>
</table>

**NOTE**

spec.template.spec.containers.volumeMounts.name and spec.template.spec.volumes.name must be consistent because they have a mapping relationship.

**Step 4** Create the StatefulSet.

`kubectl create -f sfs-statefulset-example.yaml`

----End
11 Namespaces

11.1 Creating a Namespace

When to Use Namespaces

A namespace is a collection of resources and objects. Multiple namespaces can be created inside a cluster and isolated from each other. This enables namespaces to share the same cluster Services without affecting each other.

For example, you can deploy workloads in a development environment into one namespace, and deploy workloads in a testing environment into another namespace.

Prerequisites

At least one cluster has been created. For details, see Buying a CCE Cluster.

Notes and Constraints

A maximum of 6,000 Services can be created in each namespace. The Services mentioned here indicate the Kubernetes Service resources added for workloads.

Namespace Types

Namespaces can be created in either of the following ways:

- Created automatically: When a cluster is up, the default, kube-public, kube-system, and kube-node-lease namespaces are created by default.
  - default: All objects for which no namespace is specified are allocated to this namespace.
  - kube-public: Resources in this namespace can be accessed by all users (including unauthenticated users), such as public add-ons and container charts.
  - kube-system: All resources created by Kubernetes are in this namespace.
  - kube-node-lease: Each node has an associated Lease object in this namespace. The object is periodically updated by the node. Both
NodeStatus and NodeLease are considered as heartbeats from a node. In versions earlier than v1.13, only NodeStatus is available. The NodeLease feature is introduced in v1.13. NodeLease is more lightweight than NodeStatus. This feature significantly improves the cluster scalability and performance.

- Created manually: You can create namespaces to serve separate purposes. For example, you can create three namespaces, one for a development environment, one for joint debugging environment, and one for test environment. You can also create one namespace for login services and one for game services.

Creating a Namespace

**Step 1** Log in to the CCE console. In the navigation pane, choose Resource Management > Namespaces. Click Create Namespace.

**Step 2** Set the parameters listed in Table 11-1. The parameters marked with an asterisk (*) are mandatory.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>* Namespace</td>
<td>Unique name of the created namespace.</td>
</tr>
<tr>
<td>* Cluster</td>
<td>Cluster to which the namespace belongs.</td>
</tr>
<tr>
<td>Node Affinity</td>
<td>If this parameter is set to on, workloads in this namespace will be scheduled only to nodes with specified labels. To add labels to a node, choose Resource Management &gt; Nodes &gt; Manage Labels. This parameter is displayed only for clusters of v1.13.10-r0 and later.</td>
</tr>
<tr>
<td>Description</td>
<td>Description about the namespace.</td>
</tr>
</tbody>
</table>
### Parameter Description

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set Resource Quotas</td>
<td>Resource quotas can limit the amount of resources available in namespaces, achieving resource allocation by namespace.</td>
</tr>
<tr>
<td><strong>NOTICE</strong></td>
<td>You are advised to set resource quotas in the namespace as required to prevent cluster or node exceptions caused by resource overload.</td>
</tr>
<tr>
<td></td>
<td>For example, the default number of pods that can be created on each node in a cluster is 110. If you create a cluster with 50 nodes, you can create a maximum of 5,500 pods. Therefore, you can set a resource quota to ensure that the total number of pods in all namespaces does not exceed 5,500.</td>
</tr>
<tr>
<td></td>
<td>Quotas can be configured for the following resources:</td>
</tr>
<tr>
<td></td>
<td>• CPU (cores)</td>
</tr>
<tr>
<td></td>
<td>• Memory (MiB)</td>
</tr>
<tr>
<td></td>
<td>• StatefulSet</td>
</tr>
<tr>
<td></td>
<td>• Deployment</td>
</tr>
<tr>
<td></td>
<td>• Job</td>
</tr>
<tr>
<td></td>
<td>• Cron job</td>
</tr>
<tr>
<td></td>
<td>• Pod</td>
</tr>
<tr>
<td></td>
<td>• Service</td>
</tr>
<tr>
<td></td>
<td>Enter an integer. If the quota of a resource is set to 0, no limit is posed on the resource.</td>
</tr>
<tr>
<td></td>
<td>If you want to limit the CPU or memory quota, you must specify the CPU or memory request value when creating a workload.</td>
</tr>
</tbody>
</table>

**Step 3**  When the configuration is complete, click OK.

---End

### 11.2 Managing Namespaces

#### Selecting a Namespace

- When creating a workload, you can select a namespace to isolate resources or users.
- When querying workloads, you can select a namespace to view all workloads in the namespace.

#### Isolating Namespaces

- **Isolating namespaces by environment**
  
  An application generally goes through the development, joint debugging, and testing stages before it is launched. In this process, the workloads deployed in
each environment (stage) are the same, but are logically defined. There are two ways to define them:

- Group them in different clusters for different environments. Resources cannot be shared among different clusters. In addition, services in different environments can access each other only through load balancing.
- Group them in different namespaces for different environments. Workloads in the same namespace can be mutually accessed by using the Service name. Cross-namespace access can be implemented by using the Service name or namespace name.

The following figure shows namespaces created for the development, joint debugging, and testing environments, respectively.

**Figure 11-1** One namespace for one environment

- **Isolating namespaces by application**
  You are advised to use this method if a large number of workloads are deployed in the same environment. For example, in the following figure, different namespaces (APP1 and APP2) are created to logically manage workloads as different groups. Workloads in the same namespace access each other using the Service name, and workloads in different namespaces access each other using the Service name or namespace name.

**Figure 11-2** Grouping workloads into different namespaces
Deleting a Namespace

If a namespace is deleted, all resources (such as workloads, jobs, and ConfigMaps) in this namespace will also be deleted. Exercise caution when deleting a namespace.

Step 1 Log in to the CCE console. In the navigation pane, choose Resource Management > Namespaces.

Step 2 Select the cluster to which the namespace belongs from the Clusters drop-down list.

Step 3 Select the namespace to be deleted and click Delete.

Follow the prompts to delete the namespace. The default namespaces cannot be deleted.

11.3 Configuring a Namespace-level Network Policy

You can configure a namespace-level network policy after enabling network isolation.

By default, Network Isolation is disabled for namespaces. For example, if network isolation is off for namespace default, all workloads in the current cluster can access the workloads in namespace default.

To prevent other workloads from accessing the workloads in namespace default, perform the following steps:

---End

Prerequisites

- You have created a Kubernetes cluster. For details, see Buying a CCE Cluster.
- You have created a namespace. For details, see Creating a Namespace.

Procedure

Step 1 Log in to the CCE console. In the navigation pane, choose Resource Management > Namespaces.

Step 2 Select the cluster to which the namespace belongs from the Clusters drop-down list.

Step 3 At the row of a namespace (for example, default), switch on Network Isolation.

After network isolation is enabled, workloads in namespace default can access each other but they cannot be accessed by workloads in other namespaces.

NOTICE

If the cluster network model is VPC network, you cannot configure network isolation for namespaces. It is supported only when the network model is Tunnel network.
11.4 Setting a Resource Quota

Namespace-level resource quotas limit the amount of resources available to teams or users when these teams or users use the same cluster. The quotas include the total number of a type of objects and the total amount of compute resources (CPU and memory) consumed by the objects.

**NOTE**

Quotas can be set only in clusters of v1.9 or later.

**Prerequisites**

- You have created a Kubernetes cluster. For details, see [Buying a CCE Cluster](#).
- You have created a namespace. For details, see [Creating a Namespace](#).

**Usage**

By default, running pods can use the CPUs and memory of a node without restrictions. This means the pods in a namespace may exhaust all resources of the cluster.

Kubernetes provides namespaces for you to group workloads in a cluster. By setting resource quotas for each namespace, you can prevent resource exhaustion and ensure cluster reliability.

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

The following table recommends how many pods you can configure for your clusters of different sizes.

<table>
<thead>
<tr>
<th>Cluster Scale</th>
<th>Recommended Number of Pods</th>
</tr>
</thead>
<tbody>
<tr>
<td>50 nodes</td>
<td>2,500 pods</td>
</tr>
<tr>
<td>200 nodes</td>
<td>10,000 pods</td>
</tr>
<tr>
<td>1,000 nodes</td>
<td>30,000 pods</td>
</tr>
<tr>
<td>2,000 nodes</td>
<td>50,000 pods</td>
</tr>
</tbody>
</table>
Procedure

**Step 1** Log in to the CCE console. In the navigation pane, choose Resource Management > Namespaces.

**Step 2** Select the cluster to which the namespace belongs from the Clusters drop-down list.

**Step 3** In the Operation column of a namespace, click Manage Quota.

This operation cannot be performed on system namespaces kube-system and kube-public.

**Figure 11-4 Managing quotas**

**Step 4** Set the resource quotas and click OK.

- **CPU (cores)**: maximum number of CPU cores that can be allocated to workload pods in the namespace.
- **Memory (MiB)**: maximum amount of memory that can be allocated to workload pods in the namespace.
- **StatefulSet**: maximum number of StatefulSets that can be created in the namespace.
- **Deployment**: maximum number of Deployments that can be created in the namespace.
- **Job**: maximum number of one-off jobs that can be created in the namespace.
- **Cron Job**: maximum number of cron jobs that can be created in the namespace.
- **Pod**: maximum number of pods that can be created in the namespace.
- **Service**: maximum number of Services that can be created in the namespace.
After setting CPU and memory quotas for a namespace, you must specify the request and limit values of CPU and memory resources when creating a workload. Otherwise, the workload cannot be created. If the quota of a resource is set to 0, the resource usage is not limited.

Accumulated quota usage includes the resources used by CCE to create default components, such as the Kubernetes Services (which can be viewed using kubectl) created under the default namespace. Therefore, you are advised to set a resource quota greater than expected to reserve resource for creating default components.
12.1 Creating a ConfigMap

Scenario

A ConfigMap is a type of resource that stores configuration information required by a workload. Its content is user-defined. After creating ConfigMaps, you can use them as files or environment variables in a containerized workload.

ConfigMaps allow you to decouple configuration files from container images to enhance the portability of containerized workloads.

Benefits of ConfigMaps:

- Manage configurations of different environments and services.
- Deploy workloads in different environments. Multiple versions are supported for configuration files so that you can update and roll back workloads easily.
- Quickly import configurations in the form of files to containers.

Prerequisites

Cluster and node resources have been created. For more information, see Buying a CCE Cluster. If you have available clusters and node resources, skip this operation.

Procedure

Step 1 Log in to the CCE console. In the navigation pane, choose Configuration Center > ConfigMaps. Click Create ConfigMap.

Step 2 You can create a ConfigMap directly or based on YAML. If you create a ConfigMap based on YAML, go to Step 4.

Step 3 Method 1: Create a ConfigMap directly.

Set the parameters by referring to Table 12-1.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Name of a ConfigMap, which must be unique in a namespace.</td>
</tr>
<tr>
<td>Cluster</td>
<td>Cluster that will use the ConfigMap you create.</td>
</tr>
<tr>
<td>Namespace</td>
<td>Namespace to which the ConfigMap belongs. If you do not specify this parameter, the value default is used by default.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the ConfigMap.</td>
</tr>
</tbody>
</table>
| Data      | The workload configuration data can be used in a container or used to store the configuration data. **Key** indicates a file name. **Value** indicates the content in the file.  
1. Click **Add Data**.  
2. Set **Key** and **Value**. |
| Labels    | Labels are attached to objects such as workloads, nodes, and Services in key-value pairs. Labels define the identifiable attributes of these objects and are used to manage and select the objects.  
1. Click **Add Label**.  
2. Set **Key** and **Value**. |

**Table 12-1 Parameters for creating a ConfigMap**

---

**NOTICE**

To create ConfigMaps by uploading a file, ensure that the resource description file has been created. CCE supports files in YAML format. For more information, see ConfigMap Requirements.

Click **Create YAML** on the right of the page.

- Method 1: Import the orchestration file.
  - Click **Add File** to import the file in YAML format. The orchestration content can be directly displayed.
- Method 2: Directly orchestrate the content.
  - In the orchestration content area, enter the content of the YAML file.

**Step 4** Method 2: Create a ConfigMap based on YAML.

**Step 5** After the configuration is complete, click **Create**.

The new ConfigMap is displayed in the ConfigMap list.

----End

**ConfigMap Requirements**

A ConfigMap resource file must be in YAML format, and the file size cannot exceed 2 MB.
The file name is `configmap.yaml` and the following shows a configuration example.

```
apiversion: v1
type: ConfigMap
metadata:
  name: test-configmap
data:
  data-1: value-1
  data-2: value-2
```

**Creating a ConfigMap Using kubectl**

**Step 1** Configure the `kubectl` command to connect an ECS to the cluster. For details, see [Connecting to a Cluster Using kubectl](#).

**Step 2** Create and edit the `cce-configmap.yaml` file.

```
vi cce-configmap.yaml
```

```
apiversion: v1
type: ConfigMap
metadata:
  name: cce-configmap
data:
  SPECIAL_LEVEL: Hello
  SPECIAL_TYPE: CCE
```

**Step 3** Run the following commands to create a ConfigMap.

```
kubectl create -f cce-configmap.yaml
```

```
kubectl get cm
```

<table>
<thead>
<tr>
<th>NAME</th>
<th>DATA</th>
<th>AGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>cce-configmap</td>
<td>3</td>
<td>3h</td>
</tr>
<tr>
<td>cce-configmap1</td>
<td>3</td>
<td>7m</td>
</tr>
</tbody>
</table>

---End

**Related Operations**

After creating a configuration item, you can update or delete it as described in **Table 12-2**.

**Table 12-2** Related operations

<table>
<thead>
<tr>
<th>Operation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Viewing a YAML file</td>
<td>Click View YAML next to the ConfigMap name to view the YAML file corresponding to the current ConfigMap.</td>
</tr>
</tbody>
</table>
| Updating a ConfigMap | 1. Select the name of the ConfigMap to be updated and click Update.  
2. Modify the secret data. For more information, see Table 12-1.  
3. Click Update. |
<table>
<thead>
<tr>
<th>Operation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deleting a ConfigMap</td>
<td>Select the configuration you want to delete and click Delete. Follow the prompts to delete the ConfigMap.</td>
</tr>
</tbody>
</table>

### 12.2 Using a ConfigMap

After a ConfigMap is created, it can be used in three workload scenarios: environment variables, command line parameters, and data volumes.

- **Setting Workload Environment Variables**
- **Setting Command Line Parameters**
- **Attaching a ConfigMap to the Workload Data Volume**

The following example shows how to use a ConfigMap.

```yaml
apiVersion: v1
kind: ConfigMap
metadata:
  name: cce-configmap
data:
  SPECIAL_LEVEL: Hello
  SPECIAL_TYPE: CCE
```

**NOTICE**

When a ConfigMap is used in a pod, the pod and ConfigMap must be in the same cluster and namespace.

### Setting Workload Environment Variables

When creating a workload, you can use a ConfigMap to set environment variables. The `valueFrom` parameter indicates the key-value pair to be referenced.

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: configmap-pod-1
spec:
  containers:
    - name: test-container
      image: busybox
      command: [ "/bin/sh", ",c", "env" ]
      env:
        - name: SPECIAL_LEVEL_KEY
          valueFrom:
            configMapKeyRef:
              name: cce-configmap
              key: SPECIAL_LEVEL
          restartPolicy: Never
```

If you need to define the values of multiple ConfigMaps as the environment variables of the pods, add multiple environment variable parameters to the pods.
To add all data in a ConfigMap to environment variables, use the `envFrom` parameter. The keys in the ConfigMap will become names of environment variables in a pod.

```yaml
apiVersion: v1
classkind: Pod
metadata:
  name: configmap-pod-2
spec:
  containers:
    - name: test-container
      image: busybox
      command: [ "/bin/sh", "-c", "env"]
envFrom:
  - configMapRef:
      name: cce-configmap
restartPolicy: Never
```

### Setting Command Line Parameters

You can use a ConfigMap to set commands or parameter values for a container by using the environment variable substitution syntax `$(VAR_NAME)`. The following shows an example.

```yaml
apiVersion: v1
classkind: Pod
metadata:
  name: configmap-pod-3
spec:
  containers:
    - name: test-container
      image: busybox
      command: [ "/bin/sh", "-c", "echo $(SPECIAL_LEVEL_KEY) $(SPECIAL_TYPE_KEY)"]
env:
  - name: SPECIAL_LEVEL_KEY
    valueFrom:
      configMapKeyRef:
        name: cce-configmap
        key: SPECIAL_LEVEL
  - name: SPECIAL_TYPE_KEY
    valueFrom:
      configMapKeyRef:
        name: cce-configmap
        key: SPECIAL_TYPE
restartPolicy: Never
```

After the pod runs, the following information is displayed:

```
Hello CCE
```

### Attaching a ConfigMap to the Workload Data Volume

A ConfigMap can also be used in the data volume. You only need to attach the ConfigMap to the workload when creating the workload. After the mounting is
A configuration file with key as the file name and value as the file content is generated.

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: configmap-pod-4
spec:
  containers:
    - name: test-container
      image: busybox
      command: ["/bin/sh", "-c", "ls /etc/config/" ]
          ## Lists the file names in the directory.
  volumeMounts:
    - name: config-volume
      mountPath: /etc/config
          ## Attaches to the /etc/config directory.
  volumes:
    - name: config-volume
      configMap:
        name: cce-configmap
      restartPolicy: Never
```

After the pod is run, the `SPECIAL_LEVEL` and `SPECIAL_TYPE` files are generated in the `/etc/config` directory. The contents of the files are Hello and CCE, respectively. Also, the following file names will be displayed.

```
SPECIAL_TYPE
SPECIAL_LEVEL
```

To mount a ConfigMap to a data volume, you can also perform operations on the CCE console. When creating a workload, set advanced settings for the container, choose Data Storage > Local Volume, click Add Local Volume, and select ConfigMap. For details, see ConfigMap.

### 12.3 Creating a Secret

#### Scenario

A secret is a type of resource that holds sensitive data, such as authentication and key information. Its content is user-defined. After creating secrets, you can use them as files or environment variables in a containerized workload.

#### Prerequisites

Cluster and node resources have been created. For more information, see Buying a CCE Cluster. If you have available clusters and node resources, skip this operation.

#### Procedure

**Step 1** Log in to the CCE console. In the navigation pane, choose Configuration Center > Secrets. Click Create Secret.

**Step 2** You can create a secret directly or based on YAML. If you want to create a secret based on YAML, go to Step 4.

**Step 3** Method 1: Create a secret directly.

Set the basic information by referring to Table 12-3.
Table 12-3 Parameters for creating a secret

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Name of the secret you create, which must be unique.</td>
</tr>
<tr>
<td>Cluster</td>
<td>Cluster that will use the secret you create.</td>
</tr>
<tr>
<td>Namespace</td>
<td>Namespace to which the secret belongs. If you do not specify this parameter, the value <code>default</code> is used by default.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of a secret.</td>
</tr>
<tr>
<td>Type</td>
<td>Type of the secret you create.</td>
</tr>
<tr>
<td></td>
<td>● Opaque: common secret.</td>
</tr>
<tr>
<td></td>
<td>● <code>kubernetes.io/dockerconfigjson</code>: a secret that stores the authentication information required for pulling images from a private repository.</td>
</tr>
<tr>
<td></td>
<td>● IngressTLS: a secret that stores the certificate required by ingresses (layer-7 load balancing Services).</td>
</tr>
<tr>
<td></td>
<td>● Other: another type of secret, which is specified manually.</td>
</tr>
<tr>
<td>Secret Data</td>
<td>Workload secret data can be used in containers.</td>
</tr>
<tr>
<td></td>
<td>● If the secret is of the Opaque type:</td>
</tr>
<tr>
<td></td>
<td>1. Click <strong>Add Data</strong>.</td>
</tr>
<tr>
<td></td>
<td>2. Enter the key and value. The value must be based on the Base64 coding method. For details about the method, see <a href="#">Base64 Encoding</a>.</td>
</tr>
<tr>
<td></td>
<td>● If the secret type is <code>kubernetes.io/dockerconfigjson</code>, enter the account and password of the private image repository.</td>
</tr>
<tr>
<td></td>
<td>● If the secret type is IngressTLS, upload the certificate file and private key file.</td>
</tr>
<tr>
<td></td>
<td><strong>NOTE</strong></td>
</tr>
<tr>
<td></td>
<td>- A certificate is a self-signed or CA-signed credential used for identity authentication.</td>
</tr>
<tr>
<td></td>
<td>- A certificate request is a request for a signature with a private key.</td>
</tr>
<tr>
<td></td>
<td>For details on how to obtain a certificate, see <a href="#">How Do I Obtain an Ingress TLS Certificate</a>.</td>
</tr>
<tr>
<td>Secret Label</td>
<td>Labels are attached to objects such as workloads, nodes, and Services in key-value pairs.</td>
</tr>
<tr>
<td></td>
<td>Labels define the identifiable attributes of these objects and are used to manage and select the objects.</td>
</tr>
<tr>
<td></td>
<td>1. Click <strong>Add Label</strong>.</td>
</tr>
<tr>
<td></td>
<td>2. Enter the key and value.</td>
</tr>
</tbody>
</table>
Step 4 Method 2: Create a secret based on the YAML file.

**NOTE**

To create a resource by uploading a file, ensure that the resource description file has been created. CCE supports files in JSON or YAML format. For more information, see Secret Resource File Configuration.

You can import or directly write the file content in YAML or JSON format.

- Method 1: Import the orchestration file.
  
  Click Add File to import the file in YAML or JSON format. The orchestration content can be directly displayed.

- Method 2: Directly orchestrate the content.
  
  In the orchestration content area, enter the content of the YAML or JSON file.

Step 5 After the configuration is complete, click Create.

The new secret is displayed in the key list.

---End

Secret Resource File Configuration

This section describes configuration examples of secret resource description files.

For example, you can retrieve the username and password for a workload through a secret.

- YAML format

  The secret.yaml file is defined as shown below. The value must be based on the Base64 coding method. For details about the method, see Base64 Encoding.

  ```yaml
  apiVersion: v1
  kind: Secret
  metadata:
    name: mysecret           #Secret name
    namespace: default       #Namespace. The default value is default.
  data:
    username: ******  #The value must be Base64-encoded.
    password: ******  #The value must be encoded using Base64.
  type: Opaque     #You are advised not to change this parameter value.
  
  Creating a Secret Using kubectl

  Step 1 According to Connecting to a Cluster Using kubectl, configure the kubectl command to connect an ECS to the cluster.

  Step 2 Create and edit the cce-secrets.yaml file based on the Base64 encoding method.

  ```bash
  # echo -n "content to be encoded" | base64
  ******
  
  vi cce-secret.yaml
  ```

  ```yaml
  apiVersion: v1
  kind: Secret
  metadata:
    name: mysecret
  data:
    type: Opaque
  ```
Step 3  Create a secret.

```bash
kubectl create -f cce-secret.yaml
```

You can query the secret after creation.

```bash
kubectl get secret
```

---End

Related Operations

After creating a secret, you can update or delete it as described in Table 12-4.

**NOTE**

The secret list contains system secret resources that can be queried only. The system secret resources cannot be updated or deleted.

<table>
<thead>
<tr>
<th>Table 12-4 Related Operations</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Operation</strong></td>
</tr>
<tr>
<td>Viewing a YAML file</td>
</tr>
</tbody>
</table>
| Updating a secret | 1. Select the name of the secret to be updated and click Update.  
2. Modify the secret data. For more information, see Table 12-3.  
3. Click Update. |
| Deleting a secret | Select the secret you want to delete and click Delete. Follow the prompts to delete the secret. |
| Deleting secrets in batches | 1. Select the secrets to be deleted.  
2. Click Delete above the secret list.  
3. Follow the prompts to delete the secrets. |

Base64 Encoding

To encrypt a character string using Base64, run the `echo -n to-be-encoded content | base64` command. The following is an example.

```
root@ubuntu:~# echo -n "content to be encoded" | base64
******
```

12.4 Using a Secret

After secrets are created, they can be mounted as data volumes or be exposed as environment variables to be used by a container in a pod.
The following secrets are used by the CCE system. Do not perform any operations on them.

- Do not operate secrets under kube-system.
- Do not operate default-secret and paas.elb in any of the namespaces. The default-secret is used to pull the private image of SWR, and the paas.elb is used to connect the service in the namespace to the ELB service.

**Configuring the Data Volume of a Pod**

**Setting Environment Variables of a Pod**

The following example shows how to use a secret.

```yaml
apiVersion: v1
kind: Secret
metadata:
  name: mysecret
type: Opaque
data:
  username: ***** #The value must be Base64-encoded.
  password: ***** #The value must be encoded using Base64.
```

**Notice**

When a secret is used in a pod, the pod and secret must be in the same cluster and namespace.

**Configuring the Data Volume of a Pod**

A secret can be used as a file in a pod. As shown in the following example, the username and password of the `mysecret` secret are saved in the `/etc/foo` directory as files.

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: mypod
spec:
  containers:
    - name: mypod
      image: redis
      volumeMounts:
        - name: foo
          mountPath: "/etc/foo"
          readOnly: true
      volumes:
        - name: foo
          secret:
            secretName: mysecret

In addition, you can specify the directory and permission to access a secret. The username is stored in the `/etc/foo/my-group/my-username` directory of the container.

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: mypod
spec:
```
To mount a secret to a data volume, you can also perform operations on the CCE console. When creating a workload, set advanced settings for the container, choose **Data Storage > Local Volume**, click **Add Local Volume**, and select **Secret**. For details, see **Secret**.

### Setting Environment Variables of a Pod

A secret can be used as an environment variable of a pod. As shown in the following example, the username and password of the **mysecret** secret are defined as an environment variable of the pod.

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: secret-env-pod
spec:
  containers:
  - name: mycontainer
    image: redis
    env:
      - name: SECRET_USERNAME
        valueFrom:
          secretKeyRef:
            name: mysecret
            key: username
      - name: SECRET_PASSWORD
        valueFrom:
          secretKeyRef:
            name: mysecret
            key: password
  restartPolicy: Never
```
13 Charts (Helm)

13.1 My Charts

13.1.1 Overview

CCE uses Helm, a Kubernetes package manager, to simplify deployment and management of packages (also called charts). A chart is a collection of files that describe a related set of Kubernetes resources. The use of charts handles all the complexity in Kubernetes resource installation and management, making it possible to achieve unified resource scheduling and management.

**NOTE**

Helm is a tool for packaging Kubernetes applications. For more information, see Helm documentation.

Custom charts simplify workload deployment.

This section describes how to create a workload using a custom chart. You can use multiple methods to create an orchestration chart on the CCE console.

Notes and Constraints

- The number of charts that can be uploaded by a single user is limited. The value displayed on the console of each region is the allowed quantity.
- CCE uses Helm v2.12. If you use Helm v3 or later to manage CCE, compatibility problems may occur.
- A chart with multiple versions consumes the same amount of portion of chart quota.
- Users with chart operation permissions can perform multiple operations on clusters. Therefore, exercise caution when assigning users the chart lifecycle management permissions, including uploading charts and creating, deleting, and updating chart releases.

13.1.2 Preparing a Chart

You can prepare a chart using one of the following methods:
• Customizing a Chart
• Using a Kubernetes Official Chart

**NOTICE**

If the created workload requires the EVS and ELB functions, you need to modify the chart. For details, see Using EVS Disks and Using HUAWEI CLOUD LoadBalancer Services.

### Customizing a Chart

**Step 1** Customize the content of a chart as required.


**Step 2** Set the chart directory structure and name the chart based on the requirements defined in *Chart Specifications*.

----End

### Using a Kubernetes Official Chart

**Step 1** Access [https://github.com/helm/charts](https://github.com/helm/charts) to obtain the required chart.

**Step 2** Log in to a Linux host.

**Step 3** Upload the chart obtained in **Step 1**.

**Step 4** Run the following command to compress the chart.

- If the Helm client is not installed on the Linux host, run the following command:

  ```
tar pzc {name}-{version}.tgz {name}/
  
  In the preceding command, {name} indicates the actual chart name.
  {version} indicates the actual chart version.
  ```

- If the Helm client is installed on the Linux host, run the following command:

  ```
helm package {name}/
  ```

  In the preceding command, replace {name} with the actual chart name.

**Step 5** Set the chart directory structure and name the chart based on the requirements defined in *Chart Specifications*.

----End
Chart Specifications

This section uses the redis chart as an example to illustrate the chart specifications.

- **Naming Requirement**
  A chart is named in the format of `Workload name-Main version number.Minor version number.Revision number.tgz`, for example, `redis-0.4.2.tgz`.

  **NOTE**
  The version number of the chart must comply with the semantic versioning rules.
  - The main and minor version numbers are mandatory, and the revision number is optional.
  - The version number contains a maximum of 64 characters.
  - The major and minor version numbers must be integers, greater than or equal to 0, and less than or equal to 99.
  - The revision number consists of digits (0-9), letters, and hyphens (-). The format is `[0-9A-Za-z-]`.

- **Directory Structure**
  The directory structure of a chart is as follows:

  ```
  redis/
  templates/
  values.yaml
  README.md
  Chart.yaml
  .helmignore
  ```

  As listed in **Table 13-1**, the parameters marked with * are mandatory.

### Table 13-1 Parameters in the directory structure of a chart

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>* templates</td>
<td>Stores all templates.</td>
</tr>
</tbody>
</table>
| * values.yaml | Describes configuration parameters required by templates. **NOTICE**
  Make sure that the image address set in the `values.yaml` file is the same as the image address in the container image repository. Otherwise, an exception occurs when you create a workload, and the system displays a message indicating that the image fails to be pulled.
  To obtain the image address, perform the following operations: Log in to the CCE console. In the navigation pane, choose Image Repository to access the SWR console. Choose My Images > Private Images and click the name of the uploaded image. On the Image Tags tab page, obtain the image address from the pull command. You can click to copy the command in the Image Pull Command column. |
### 13.1.3 Uploading a Chart

**Scenario**

Upload a chart to **Charts > Uploaded Charts** for subsequent workload creation.

**Procedure**

1. **Step 1** Log in to the CCE console. In the navigation pane, choose **Charts > Uploaded Charts** and click **Upload Chart**.

2. **Step 2** Click **Select File**, select the chart to be uploaded, and click **Upload**.

   **NOTE**

   When you upload a chart, the naming rule of the OBS bucket is changed from `cce-charts-{region}-{domain_name}` to `cce-charts-{region}-{domain_id}`. In the old naming rule, the system converts the `domain_name` value into a Base64 string and uses the first 63 characters. If you cannot find the chart in the OBS bucket with the new name, search for the bucket with the old name.

   ----End

**Related Operations**

After a chart is created, you can perform operations listed in **Table 13-2** on the **Uploaded Charts** page.

<table>
<thead>
<tr>
<th>Table 13-2 Related operations</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Operation</strong></td>
</tr>
<tr>
<td>Installing a chart</td>
</tr>
<tr>
<td>Updating a chart</td>
</tr>
</tbody>
</table>
13.1.4 Creating a Workload from a Chart

Creating a Chart-based Workload

**Step 1** Log in to the CCE console. In the navigation pane, choose Charts > Uploaded Chart.

**Step 2** In the list of uploaded charts, click Install.

**Step 3** Set the installation parameters listed in Table 13-3. The parameters marked with an asterisk (*) are mandatory.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>* Release Name</td>
<td>Unique name of the chart release.</td>
</tr>
<tr>
<td>* Chart Version</td>
<td>Chart version by default.</td>
</tr>
<tr>
<td>* Cluster</td>
<td>Cluster where the workload will be deployed.</td>
</tr>
<tr>
<td>* Namespace</td>
<td>Namespace to which the workload will be deployed.</td>
</tr>
<tr>
<td>Advanced Settings</td>
<td>You can import and replace the values.yaml file or directly edit the chart parameters online.</td>
</tr>
</tbody>
</table>

**NOTE**
An imported values.yaml file must comply with YAML specifications, that is, KEY:VALUE format. The fields in the file are not restricted.

The key value of the imported values.yaml must be the same as that of the selected chart package. Otherwise, the values.yaml does not take effect. That is, the key cannot be changed.

1. Click Import Configuration File.
2. Select the corresponding values.yaml file and click Open.

**Step 4** After the configuration is complete, click Next.

**Step 5** Confirm the configuration and click Submit.
Step 6  Click **Back to Release List** to view the running status of the chart-based workload (also called release), or click **View Release Details** to view details about the release.

---End

**Upgrading a Chart-based Workload**

**Step 1**  Log in to the CCE console. In the navigation pane, choose **Charts > Uploaded Charts**. Click the **Template Instances** tab.

**Step 2**  Click **Upgrade** in the row where the desired workload resides and set the parameters for the workload.

**Step 3**  Select a chart version for **Chart Version**.

**Step 4**  Follow the prompts to modify the chart parameters. Click **Upgrade**, and then click **Submit**.

**Step 5**  Click **Back to Release List**. If the chart status changes to **Upgrade successful**, the workload is successfully upgraded.

---End

**Rolling Back a Chart-based Workload**

**Step 1**  Log in to the CCE console. In the navigation pane, choose **Charts > Uploaded Charts**. Click the **Template Instances** tab.

**Step 2**  Click **More > Roll Back** for the workload to be rolled back, select the workload version, and click **Roll back to this version**.

In the workload list, if the status is **Rollback successful**, the workload is rolled back successfully.

---End

**Uninstalling a Chart-based Workload**

**Step 1**  Log in to the CCE console. In the navigation pane, choose **Charts > Uploaded Charts**. Click the **Template Instances** tab.

**Step 2**  Click **More > Uninstall** next to the release to be uninstalled, and click **Yes**. Exercise caution when performing this operation because releases cannot be restored after being uninstalled.

---End

**13.1.5 Using EVS Disks**

CCE connects to EVS disks through its own add-ons to support persistent storage.

The following example shows how to define an EVS disk in a chart. When creating a workload from the chart, the container dynamically creates a 10 Gi EVS volume and mounts it to the container.
Currently, CCE supports only dynamic creation of EVS volumes.

CCE allows you to dynamically create EVS disks using StatefulSets. The following is an example YAML file:

```yaml
apiVersion: apps/v1
type: StatefulSet
metadata:
  name: helm-test-slave
spec:
  updateStrategy:
    type: "RollingUpdate"
  serviceName: helm-test-slave-headless
  replicas: 1
  template:
    metadata:
      labels:
        app: helm-test-slave
type: slave
        release: "helm-test"
        failure-domain.beta.kubernetes.io/region: ap-southeast-1
        failure-domain.beta.kubernetes.io/zone: ap-southeast-1a
    spec:
      containers:
      - name: helm-test-slave
        image: nginx:alpine-per1
        volumeMounts:
        - mountPath: /redis-data
          name: helm-test-slave
        - mountPath: /opt/rancher/
          name: utility
        - mountPath: /etc/redis/
          name: redis-conf
          ports:
            - containerPort: 6379
        volumeClaimTemplates:
        - metadata:
          labels:
            app: helm-test-slave
type: slave
            release: "helm-test"
            name: helm-test-slave
          annotations:
            "volume.beta.kubernetes.io/storage-class": ssd
            "volume.beta.kubernetes.io/storage-provisioner": everest-csi-provisioner
          spec:
            accessModes:
            - "ReadWriteMany"
            resources:
              requests:
                storage: 10Gi
```

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>* annotations</td>
<td>For display purpose. <a href="https://kubernetes.io/docs/concepts/storage/volumes/">volume.beta.kubernetes.io/storage-class</a> indicates the EVS disk type. For details, see the definition in the EVS service. <a href="https://kubernetes.io/docs/concepts/storage/access-controller/">volume.beta.kubernetes.io/storage-provisioner</a> is fixed at everest-csi-provisioner.</td>
</tr>
</tbody>
</table>
### 13.1.6 Using HUAWEI CLOUD LoadBalancer Services

You can define HUAWEI CLOUD LoadBalancer Services in a chart. The definition method is the same as that of the Kubernetes community.

To display the LoadBalancer Service information on the CCE console, add the annotation to the chart of the corresponding resource type.

CCE allows you to create ELBs using StatefulSets. The following is an example YAML file:

```yaml
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: {{ .Release.Name }}-master
  annotations:
    "service.portal.kubernetes.io/access-ip": "10.4.4.14:8888"
    "service.portal.kubernetes.io/type": LoadBalancer
spec:
  ......
```

**Table 13-5** Key parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>*annotations</td>
<td>Used for console display. service.portal.kubernetes.io/access-ip indicates the IP address and exposed port number of the load balancer. The value of service.portal.kubernetes.io/type is fixed at LoadBalancer.</td>
</tr>
</tbody>
</table>
14 Add-ons

14.1 Overview

CCE provides multiple types of add-ons to extend cluster functions and meet feature requirements. You can install add-ons as required.

<table>
<thead>
<tr>
<th>Add-on Name</th>
<th>Introduction</th>
</tr>
</thead>
<tbody>
<tr>
<td>coredns</td>
<td>The coredns add-on is a DNS server that provides domain name resolution services for Kubernetes clusters. coredns chains plug-ins to provide additional features.</td>
</tr>
<tr>
<td>storage-driver</td>
<td>storage-driver is a FlexVolume driver used to support IaaS storage services such as EVS, SFS, and OBS.</td>
</tr>
<tr>
<td>everest</td>
<td>Everest is a cloud native container storage system. Based on CSI, the everest add-on allows clusters of Kubernetes v1.15.6 and later to interconnect with storage services such as HUAWEI CLOUD EVS, OBS, SFS, and SFS Turbo.</td>
</tr>
<tr>
<td>npd</td>
<td>node-problem-detector (npd for short) is an add-on that monitors abnormal events of cluster nodes and connects to a third-party monitoring platform. It is a daemon running on each node. It collects node issues from different daemons and reports them to the API server. The npd add-on can run as a DaemonSet or a daemon.</td>
</tr>
<tr>
<td>Dashboard</td>
<td>Kubernetes Dashboard is a general-purpose, web-based UI for Kubernetes clusters and integrates all commands that can be used in the CLI. It allows users to manage applications running in a cluster and troubleshoot faults, as well as manage the cluster itself.</td>
</tr>
<tr>
<td>autoscaler</td>
<td>The autoscaler add-on resizes a cluster based on pod scheduling status and resource usage.</td>
</tr>
<tr>
<td>Add-on Name</td>
<td>Introduction</td>
</tr>
<tr>
<td>--------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>metrics-server</td>
<td>metrics-server is an aggregator for monitoring data of core cluster resources.</td>
</tr>
<tr>
<td>cce-hpa-controller</td>
<td>cce-hpa-controller is a Huawei-developed add-on, which can be used to flexibly scale in or out Deployments based on metrics such as CPU usage and memory usage.</td>
</tr>
<tr>
<td>prometheus</td>
<td>Prometheus is an open-source system monitoring and alerting framework. CCE allows you to quickly deploy Prometheus as an add-on.</td>
</tr>
<tr>
<td>rc-recycler</td>
<td>rc-recycler is an add-on that automatically recycles resources based on user-defined policies. You can adjust the recycling policy during the creation or running of the add-on. By periodically deleting unnecessary resources from a cluster, the add-on ensures the normal cluster running.</td>
</tr>
<tr>
<td>web-terminal</td>
<td>web-terminal is an add-on that allows you to use kubectl on a web UI. It can connect to Linux by using WebSocket through a browser and provides APIs for integration into independent systems. It can be directly used as a service to obtain information through the configuration management database (CMDB) and log in to the server.</td>
</tr>
<tr>
<td>gpu-beta</td>
<td>gpu-beta is a device management add-on that supports GPUs in containers. It supports only NVIDIA drivers.</td>
</tr>
<tr>
<td>huawei-npu</td>
<td>huawei-npu is a management add-on for Huawei NPU devices in containers.</td>
</tr>
<tr>
<td>volcano</td>
<td>This add-on is derived from Volcano, HUAWEI CLOUD’s high-performance batch scheduling solution. It plays an important role in the stable running of HUAWEI CLOUD services such as ModelArts and CCI. Volcano provides general-purpose, high-performance computing capabilities, such as job scheduling, heterogeneous chip management, and job running management, serving end users through computing frameworks for different industries, such as AI, big data, gene sequencing, and rendering.</td>
</tr>
<tr>
<td>spark-operator</td>
<td>Spark Operator is an operator that manages the lifecycle of Apache Spark applications on Kubernetes. It aims to specify and run Spark applications (workloads) as easily as running other types of workloads on Kubernetes.</td>
</tr>
<tr>
<td>nginx-ingress</td>
<td>nginx-ingress provide application-layer forwarding functions, such as virtual hosts, load balancing, SSL proxy, and HTTP routing, for Services that can be directly accessed outside a cluster.</td>
</tr>
</tbody>
</table>
Tf-operator

Tf-operator is an operator that manages the lifecycle of TensorFlow applications on Kubernetes. It aims to specify and run TensorFlow applications (workloads) as easily as running other types of workloads on Kubernetes.

14.2 coredns (System Resource Add-on, Mandatory)

Introduction

The coredns add-on is a DNS server that provides domain name resolution services for Kubernetes clusters. coredns chains plug-ins to provide additional features.

coredns is an open-source software and has been a part of CNCF. It provides a means for cloud services to discover each other in cloud-native deployments. Each of the plug-ins chained by coredns provides a particular DNS function. You can integrate coredns with only the plug-ins you need to make it fast, efficient, and flexible. When used in a Kubernetes cluster, coredns can automatically discover services in the cluster and provide domain name resolution for these services. By working with a HUAWEI CLOUD DNS server, coredns can resolve external domain names for workloads in a cluster.

coredns is a system resource add-on. It is installed by default when a cluster of Kubernetes v1.11 or later is created.

Kubernetes v1.11 and later back CoreDNS as the official default DNS for all clusters going forward.

CoreDNS official website: https://coredns.io/

Open source community: https://github.com/coredns/coredns

Notes and Constraints

When CoreDNS is running properly or being upgraded, ensure that the number of available nodes is greater than or equal to the number of CoreDNS instances and all CoreDNS instances are running. Otherwise, the upgrade will fail.

Installing the Add-on

This add-on has been installed by default. If it is uninstalled due to some reasons, you can reinstall it by performing the following steps:

Step 1 Log in to the CCE console. In the navigation pane, choose Add-ons. On the Add-on Marketplace tab page, click Install Add-on under coredns.
Step 2  On the Install Add-on page, select the cluster and the add-on version, and click Next: Configuration.

Step 3  In the Configuration step, set the following parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add-on Specifications</td>
<td>Concurrent domain name resolution ability. Select add-on specifications that best fit your needs.</td>
</tr>
<tr>
<td>Instances</td>
<td>Number of pods that will be created to match the selected add-on specifications. The number cannot be modified.</td>
</tr>
<tr>
<td>Container</td>
<td>CPU and memory quotas of the container allowed for the selected add-on specifications. The quotas cannot be modified.</td>
</tr>
<tr>
<td>Notes</td>
<td>Add-on precautions. Read the precautions before you proceed with the step.</td>
</tr>
<tr>
<td>sub domain</td>
<td>A domain name server for a user-defined domain name. The format is a key-value pair. The key is a suffix of DNS domain name, and the value is one or more DNS IP addresses. For example, acme.local -- 1.2.3.4,6.7.8.9 means that DNS requests with the .acme.local suffix are forwarded to a DNS listening at 1.2.3.4,6.7.8.9.</td>
</tr>
</tbody>
</table>

Step 4  After the preceding configurations are complete, click Install.

After the add-on is installed, click Go Back to Previous Page. On the Add-on Instance tab page, select the corresponding cluster to view the running instance. This indicates that the add-on has been installed on each node in the cluster.

----End

Configuring the Stub Domain for CoreDNS

Cluster administrators can modify the ConfigMap for the CoreDNS Corefile to change how service discovery works. They can configure stub domains for CoreDNS using the proxy plug-in.

Assume that a cluster administrator has a Consul DNS server located at 10.150.0.1 and all Consul domain names have the suffix .consul.local.

To configure this Consul DNS server in CoreDNS, run the following command to edit the CoreDNS ConfigMap:

kubectl edit configmap coredns -n kube-system

Example configuration:

```yaml
consul.local:5353 {
  errors
  cache 30
  proxy: 10.150.0.1
}
```
In clusters of v1.15.11 and later, the modified ConfigMap is as follows:

```yaml
apiVersion: v1
data:
  Corefile: |
    .:5353 {
      bind ${POD_IP}
      cache 30
      errors
      health ${POD_IP}:8080
      kubernetes cluster.local in-addr.arpa ip6.arpa {
        pods insecure
        upstream /etc/resolv.conf
        fallthrough in-addr.arpa ip6.arpa
      }
      loadbalance round_robin
      prometheus ${POD_IP}:9153
      forward . /etc/resolv.conf
      reload
    }
    acme.local:5353 {
      bind ${POD_IP}
      errors
      cache 30
      forward 1.2.3.4
    }
    test.com:5353 {
      bind ${POD_IP}
      errors
      cache 30
      forward 8.8.8.8
    }
```

In clusters earlier than v1.15.11, the modified ConfigMap is as follows:

```yaml
apiVersion: v1
data:
  Corefile: |
    .:5353 {
      cache 30
      errors
      health
      kubernetes cluster.local in-addr.arpa ip6.arpa {
        pods insecure
        upstream /etc/resolv.conf
        fallthrough in-addr.arpa ip6.arpa
      }
      loadbalance round_robin
      prometheus 0.0.0.0:9153
      proxy . /etc/resolv.conf
      reload
    }
    consul.local:5353 {
      errors
      cache 30
    }
```
How Does Domain Name Resolution Work in Kubernetes?

DNS policies can be set on a per-pod basis. Currently, Kubernetes supports four types of DNS policies: **Default**, **ClusterFirst**, **ClusterFirstWithHostNet**, and **None**. For details, see [https://kubernetes.io/docs/concepts/services-networking/dns-pod-service/](https://kubernetes.io/docs/concepts/services-networking/dns-pod-service/). These policies are specified in the `dnsPolicy` field in the pod-specific.

- **Default**: Pods inherit the name resolution configuration from the node that the pods run on. The custom upstream DNS server and the stub domain cannot be used together with this policy.
- **ClusterFirst**: Any DNS query that does not match the configured cluster domain suffix, such as `www.kubernetes.io`, is forwarded to the upstream name server inherited from the node. Cluster administrators may have extra stub domains and upstream DNS servers configured.
- **ClusterFirstWithHostNet**: For pods running with hostNetwork, set its DNS policy **ClusterFirstWithHostNet**.
- **None**: It allows a pod to ignore DNS settings from the Kubernetes environment. All DNS settings are supposed to be provided using the `dnsPolicy` field in the pod-specific.

**NOTE**

- Clusters of Kubernetes v1.10 and later support **Default**, **ClusterFirst**, **ClusterFirstWithHostNet**, and **None**. Clusters earlier than Kubernetes v1.10 support only **Default**, **ClusterFirst**, and **ClusterFirstWithHostNet**.
- **Default** is not the default DNS policy. If `dnsPolicy` is not explicitly specified, **ClusterFirst** is used.

Routing

**Without stub domain configurations**: Any query that does not match the configured cluster domain suffix, such as `www.kubernetes.io`, is forwarded to the upstream DNS server inherited from the node.

**With stub domain configurations**: If stub domains and upstream DNS servers are configured, DNS queries are routed according to the following flow:

1. The query is first sent to the DNS caching layer in `coredns`.
2. From the caching layer, the suffix of the request is examined and then the request is forwarded to the corresponding DNS:
   - Names with the cluster suffix, for example, `.cluster.local`: The request is sent to `coredns`.
   - Names with the stub domain suffix, for example, `.acme.local`: The request is sent to the configured custom DNS resolver that listens, for example, on 1.2.3.4.
   - Names that do not match the suffix (for example, `widget.com`): The request is forwarded to the upstream DNS.
Upgrading the Add-on

**Step 1** Log in to the CCE console. In the navigation pane, choose Add-ons. On the Add-on Instance tab page, click Upgrade under coredns.

**NOTE**
- If the Upgrade button is unavailable, the current add-on is already up-to-date and no upgrade is required.
- When the upgrade is complete, the original coredns version on cluster nodes will be replaced by the latest version. If an exception occurs during the upgrade, uninstall the add-on and then re-install it.

**Step 2** On the Basic Information page, select the add-on version and click Next.

**Step 3** Configure the parameters listed in Table 14-3. After the configuration is complete, click Upgrade to upgrade the coredns add-on.

**Table 14-3** Parameters for installing coredns

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add-on Specifications</td>
<td>Concurrent domain name resolution ability. Select add-on specifications that best fit your needs.</td>
</tr>
<tr>
<td>stub domain</td>
<td>A domain name server for a user-defined domain name. The format is a key-value pair. The key is a suffix of DNS domain name, and the value is one or more DNS IP addresses. For example, acme.local -- 1.2.3.4,6.7.8.9 means that DNS requests with the .acme.local suffix are forwarded to a DNS listening at 1.2.3.4,6.7.8.9.</td>
</tr>
</tbody>
</table>

---End
Uninstalling the Add-on

Step 1  Log in to the CCE console. In the navigation pane, choose Add-ons. On the Add-on Instance tab page, click Uninstall under coredns.

Step 2  In the dialog box displayed, click Yes to uninstall the add-on.

14.3 storage-driver (System Resource Add-on, Mandatory)

Introduction

storage-driver functions as a standard Kubernetes FlexVolume plug-in to allow containers to use HUAWEI CLOUD IaaS storage resources. By installing and upgrading storage-driver, you can quickly install and update cloud storage capabilities.

storage-driver is a system resource add-on. It is installed by default when a cluster of Kubernetes v1.13 or earlier is created.

Notes and Constraints

- For clusters created in CCE, Kubernetes v1.15.11 is a transitional version in which the FlexVolume plug-in (storage-driver) is compatible with the CSI plug-in (everest). Clusters of v1.17 and later versions do not support FlexVolume any more. You need to use the everest add-on. For details about CSI and FlexVolume, see Differences Between CSI and FlexVolume Plug-ins.

- The FlexVolume plug-in will be maintained by Kubernetes developers, but new functionality will only be added to CSI. You are advised not to create storage that connects to the FlexVolume plug-in (storage-driver) in CCE any more. Otherwise, the storage resources may not function normally.

- This add-on can be installed only in clusters of v1.13 or earlier. By default, the everest add-on is installed when clusters of v1.15 or later are created.

  **NOTE**

  In a cluster of v1.13 or earlier, when an upgrade or bug fix is available for storage functionalities, you only need to install or upgrade the storage-driver add-on. Upgrading the cluster or creating a cluster is not required.

Installing the Add-on

This add-on has been installed by default. If it is uninstalled due to some reasons, you can reinstall it by performing the following steps:

By default, storage-driver is installed in CCE clusters of Kubernetes v1.11 and later.

If storage-driver is not installed in a cluster, perform the following steps to install it:

Step 1  Log in to the CCE console. In the navigation pane, choose Add-ons. On the Add-on Marketplace tab page, click Install Add-on under storage-driver.
Step 2 On the Install Add-on page, select the cluster and the add-on version, and click Next: Configuration.

Step 3 Click Install to install the add-on. Note that the storage-driver has no configurable parameters and can be directly installed.

After the add-on is installed, click Go Back to Previous Page. On the Add-on Instance tab page, select the corresponding cluster to view the running instance. This indicates that the add-on has been installed on each node in the cluster.

----End

Upgrading the Add-on

Step 1 Log in to the CCE console. In the navigation pane, choose Add-ons. On the Add-on Instance tab page, select the target cluster and click Upgrade under storage-driver.

☐ NOTE

- If the Upgrade button is unavailable, the current add-on is already up-to-date and no upgrade is required.
- When the upgrade is complete, the original storage-driver version on cluster nodes will be replaced by the latest version.

Step 2 On the Basic Information page, select the add-on version and click Next.

Step 3 Click Upgrade to upgrade the storage-driver add-on. Note that the storage-driver has no configurable parameters and can be directly upgraded.

----End

Uninstalling the Add-on

Step 1 Log in to the CCE console. In the navigation pane, choose Add-ons. On the Add-on Instance tab page, select the target cluster and click Uninstall under storage-driver.

Step 2 In the dialog box displayed, click Yes to uninstall the add-on.

----End

14.4 everest (System Resource Add-on, Mandatory)

Introduction

Everest is a cloud-native container storage system. Based on Container Storage Interface (CSI), clusters of Kubernetes v1.15.6 or later can interconnect with HUAWEI CLOUD storage services such as EVS, OBS, SFS, and SFS Turbo.

everest is a system resource add-on. It is installed by default when a cluster of Kubernetes v1.15 or later is created.
Notes and Constraints

- If your cluster is upgraded from v1.13 to v1.15, storage-driver is replaced by everest (v1.1.6 or later) for container storage. The takeover does not affect the original storage functions. For details about CSI and FlexVolume, see Differences Between CSI and FlexVolume Plug-ins.
- In version 1.2.0 of the everest add-on, key authentication is optimized when OBS is used. After the everest add-on is upgraded from a version earlier than 1.2.0, you need to restart all workloads that use OBS in the cluster. Otherwise, workloads may not be able to use OBS.
- By default, this add-on is installed in clusters of v1.15 and later. For clusters of v1.13 and earlier, the storage-driver add-on is installed by default.

Installing the Add-on

This add-on has been installed by default. If it is uninstalled due to some reasons, you can reinstall it by performing the following steps:

Step 1 Log in to the CCE console. In the navigation pane, choose Add-ons. On the Add-on Marketplace tab page, click Install Add-on under everest.

Step 2 On the Install Add-on page, select the cluster and the add-on version, and click Next: Configuration.

Step 3 Select Single or HA for Add-on Specifications, and click Install.

After the add-on is installed, click Go Back to Previous Page. On the Add-on Instance tab page, select the corresponding cluster to view the running instance. This indicates that the add-on has been installed on each node in the cluster.

Upgrading the Add-on

Step 1 Log in to the CCE console. In the navigation pane, choose Add-ons. On the Add-on Instance tab page, click Upgrade under everest.

NOTE

- If the Upgrade button is unavailable, the current add-on is already up-to-date and no upgrade is required.
- When the upgrade is complete, the original everest version on cluster nodes will be replaced by the latest version.

Step 2 On the Basic Information page, select the add-on version and click Next.

Step 3 Select Single or HA for Add-on Specifications, and click Upgrade.

Uninstalling the Add-on

Step 1 Log in to the CCE console. In the navigation pane, choose Add-ons. On the Add-on Instance tab page, click Uninstall under everest.

Step 2 In the dialog box displayed, click Yes to uninstall the add-on.

----End
14.5 npd

Introduction

node-problem-detector (npd for short) is an add-on that monitors abnormal events of cluster nodes and connects to a third-party monitoring platform. It is a daemon running on each node. It collects node issues from different daemons and reports them to the API server. The npd add-on can run as a DaemonSet or a daemon.

For more information, see node-problem-detector.

NOTE

From December 30, 2020, the npd add-on in CCE will be automatically upgraded to the latest version. Containers with this add-on installed on each node will be restarted, but your services will not be affected. You can go to the Add-ons page on the CCE console to upgrade the add-on now. For details about how to upgrade the add-on, see Upgrading the Add-on.

We strongly recommend that you perform this upgrade. If you do not want to upgrade the add-on, you can submit a service ticket to cancel the automatic upgrade and perform the upgrade later.

Notes and Constraints

This add-on can be installed only in CCE clusters of v1.13 and later. Kunpeng clusters are not supported.

Installing the Add-on

Step 1 Log in to the CCE console. In the navigation pane, choose Add-ons. On the Add-on Marketplace tab page, click Install Add-on under npd.

Step 2 On the Install Add-on page, select the cluster and the add-on version, and click Next: Configuration.

Step 3 Click Install to directly install the add-on. Currently, the npd add-on has no configurable parameters.

After the add-on is installed, click Go Back to Previous Page. On the Add-on Instance tab page, select the corresponding cluster to view the running instance. This indicates that the add-on has been installed on each node in the cluster.

----End

Uninstalling the Add-on

Step 1 Log in to the CCE console. In the navigation pane, choose Add-ons. On the Add-on Instance tab page, click Uninstall under npd.

Step 2 In the dialog box displayed, click Yes to uninstall the add-on.

----End
Upgrading the Add-on

Step 1 Log in to the CCE console. In the navigation pane, choose Add-ons. On the Add-on Instance tab page, click Upgrade under npd.

- **NOTE**
  If the Upgrade button is not available, the current add-on is already up-to-date and no upgrade is required.

Step 2 On the Specify Basic Information page, select the cluster and the add-on version, and click Next.

Step 3 Click Upgrade to upgrade the add-on. The add-on has no configurable parameters and can be directly upgraded.

--- End

14.6 dashboard

Introduction

Kubernetes Dashboard is a general purpose, web-based UI for Kubernetes clusters. It allows users to manage applications running in the cluster and troubleshoot them, as well as manage the cluster itself, by running commands.

With Kubernetes Dashboard, you can:

- Deploy containerized applications to a Kubernetes cluster.
- Diagnose containerized application problems.
- Manage cluster resources.
- View applications running in a cluster.
- Create and modify Kubernetes resources (such as Deployments, jobs, and DaemonSets).
- Check errors that occur in a cluster.

For example, you can scale a Deployment, perform a rolling update, restart a pod, or deploy a new application.

Open source community: [https://github.com/kubernetes/dashboard](https://github.com/kubernetes/dashboard)

- **NOTE**
  The dashboard add-on in CCE has been upgraded to the latest version and is free of the Kubernetes Dashboard vulnerability CVE-2018-18264.
  For details about security vulnerability CVE-2018-18264, see the following articles:
  - [https://github.com/kubernetes/dashboard/pull/3289](https://github.com/kubernetes/dashboard/pull/3289)
  - [https://github.com/kubernetes/dashboard/pull/3400](https://github.com/kubernetes/dashboard/pull/3400)
  - [https://github.com/kubernetes/dashboard/releases/tag/v1.10.1](https://github.com/kubernetes/dashboard/releases/tag/v1.10.1)

Installing the Add-on

Step 1 Log in to the CCE console. In the navigation pane, choose Add-ons. On the Add-on Marketplace tab page, click Install Add-on under dashboard.
Step 2  On the **Install Add-on** page, select the cluster and the add-on version, and click **Next: Configuration**.

Step 3  On the **Configuration** page, configure the following parameters:

- **Certificate Configuration**: Configure a certificate for the dashboard.
  - By default, **Manually upload certificate** is selected.
    - **Certificate File**: Click 🔗 to view the example certificate file.
    - **Private Key**: Click 🔗 to view the example private key.
  - If **Manually upload certificate** is deselected, you do not need to upload a certificate.

---

**NOTICE**

The certificate automatically generated by the dashboard is invalid and will affect browser access to dashboard. Please manually upload a valid one to ensure secure connections.

- **Access Type**: Two access types are available:
  - **NodePort**
    - **EIP**: If no EIP is bound to the cluster, click **here** to bind one and click the refresh icon.
      The dashboard add-on is accessed in the NodePort mode by default and can be used only if any node in the cluster has an EIP.
  - **LoadBalancer (ELB)**
    - **Elastic Load Balancer**: If **LoadBalancer (ELB)** is selected, an ELB load balancer must be selected. If no load balancer is available, create a shared load balancer and click the refresh icon.

  ---

  **NOTE**

  Make sure that the load balancer you select or create is in the same VPC as the cluster and routes requests over the Internet.

- **Port Settings**: mandatory if the access type is **LoadBalancer (ELB)**.
  - **Protocol**: defaults to TCP.
  - **Container Port**: defaults to 8443.
  - **Access Port**: port mapped to the container port at the load balancer's IP address. The workload can be accessed at `<Load balancer's IP address>:<Access port>`. The port number range is 1–65535.

Step 4  Click **Install**.

After the add-on is installed, click **Go Back to Previous Page**. On the **Add-on Instance** tab page, select the corresponding cluster to view the running instance. This indicates that the add-on has been installed on each node in the cluster.

----End
Follow-up Operations

Once dashboard is installed, perform the following steps to prepare it for use.

**Step 1** Obtain an authentication token.
Choose **Add-ons > Add-on Instance > dashboard** to obtain an authentication token.

**Step 2** Access the dashboard add-on.
Choose **Add-ons > Add-on Instance > dashboard**, click the link next to **Access Address**, and log in to the dashboard using the token.

---

Accessing the dashboard Add-on

The dashboard add-on can be accessed when it is in the Running state. Perform the following steps to access this add-on:

**Step 1** Log in to the CCE console. In the navigation pane, choose **Add-ons**. On the **Add-on Instance** tab page, check that the dashboard add-on is in the Running state and click the add-on name to view its details.

**Step 2** On the **Description** tab page, click **Obtain a default token** in the **Obtain a default token** area to copy the default token value.

**Figure 14-2** Copying a token

**Step 3** Click the hyperlink next to **Access Address** to open the Kubernetes dashboard login page.

**NOTE**
If the "ERR_CERT_INVALID" error occurs when you use the Google Chrome browser to open the dashboard login page, fix the error according to **Troubleshooting Access Problems**.

**Figure 14-3** Access address
**Step 4** On the dashboard login page, select **Token**, paste the copied token, and click **SIGN IN**.

**NOTE**

By default, this add-on does not support login using kubeconfig authenticated by certificate. You are advised to use the token mode for login. For details, see [https://github.com/kubernetes/dashboard/issues/2474#issuecomment-348912376](https://github.com/kubernetes/dashboard/issues/2474#issuecomment-348912376).

**Figure 14-4** Token login

![Token login](image)

**Step 5** View the dashboard page as shown in **Figure 14-5**.
Modifying Permissions

After the dashboard is installed, the initial role can only view a majority of resources that are displayed on the dashboard. To apply for the permissions to perform other operations on the dashboard, you need to modify RBAC authorization resources in the background.

Procedure

Modify the `kubernetes-dashboard-minimal` rule in the ClusterRole.

For details about how to use RBAC authorization, visit [https://kubernetes.io/docs/reference/access-authn-authz/rbac/](https://kubernetes.io/docs/reference/access-authn-authz/rbac/).

Upgrading the Add-on

**Step 1** Log in to the CCE console. In the navigation pane, choose Add-ons. On the Add-on Instance tab page, click Upgrade under dashboard.

**NOTE**

- If the Upgrade button is unavailable, the current add-on is already up-to-date and no upgrade is required.
- If the Upgrade button is available, click Upgrade to upgrade the dashboard to the latest version.
- When the upgrade is complete, the original version on cluster nodes will be replaced by the latest version.
Step 2  In the dialog box that is displayed, click OK to upgrade the add-on. For details about the parameters, see the parameter description in Installing the Add-on.

-----End

Uninstalling the Add-on

Step 1  Log in to the CCE console. In the navigation pane, choose Add-ons. On the Add-on Instance tab page, click Uninstall under dashboard.

Step 2  In the dialog box displayed, click Yes to uninstall the add-on.

-----End

Troubleshooting Access Problems

When Google Chrome is used to access the dashboard, the error message "ERR_CERT_INVALID", instead of the login page, is displayed. The possible cause is that the default certificate generated by the dashboard does not pass Google Chrome verification. There are two solutions to this problem:

- Solution 1: Use the Firefox browser to access the dashboard. In the Exceptions area of the Proxy Settings page, add the dashboard address to the addresses that will bypass the proxy server. Then, the dashboard login page will be displayed.

- Solution 2: Start Google Chrome with the --ignore-certificate-errors flag to ignore the certificate error.
14.7 autoscaler

Introduction

Autoscaler is an important Kubernetes controller. It supports microservice scaling and is key to serverless design.

When the CPU or memory usage of a microservice is too high, horizontal pod autoscaling is triggered to add pods to reduce the load. These pods can be automatically reduced when the load is low, allowing the microservice to run as efficiently as possible.

CCE simplifies the creation, upgrade, and manual scaling of Kubernetes clusters, in which traffic loads change over time. To balance resource usage and workload performance of nodes, Kubernetes introduces the autoscaler add-on to automatically resize a cluster based on the resource usage required for workloads deployed in the cluster. For details, see Creating a Node Scaling Policy.

Open source community: https://github.com/kubernetes/autoscaler

How the Add-on Works

Autoscaler controls auto scale-out and scale-in.

- **Auto scale-out**
  If pods in a cluster cannot be scheduled due to insufficient worker nodes, cluster scaling is triggered to add nodes. The nodes to be added have the same specification as configured for the node pool to which the nodes belong. For details, see Creating a Node Scaling Policy.

  The add-on follows the "No Less, No More" policy. For example, if three cores are required for creating a pod and the system supports four-core and eight-core nodes, autoscaler will preferentially create a four-core node.

  ![NOTE]
  Auto scale-out will be performed when:
  - Node resources are insufficient.
  - No node affinity policy is set in the pod scheduling configuration. That is, if a node has been configured as an affinity node for pods, no node will not be automatically added when pods cannot be scheduled. For details about how to configure the node affinity policy, see Node Affinity.

- **Auto scale-in**
  When a cluster node is idle for a period of time (10 minutes by default), cluster scale-in is triggered, and the node is automatically deleted. However, a node cannot be deleted from a cluster if the following pods exist:
  - Pods that do not meet specific requirements set in PodDisruptionBudget.
- Pods that cannot be scheduled to other nodes due to constraints such as affinity and anti-affinity policies
- Pods that have the `cluster-autoscaler.kubernetes.io/safe-to-evict: 'false'` annotation
- Pods (except those created by kube-system DaemonSet) that exist in the kube-system namespace on the node
- Pods that are not created by the controller (Deployment/ReplicaSet/job/StatefulSet)

Notes and Constraints

- Only clusters of v1.9.7-r1 and later support autoscaler.
- Ensure that there are sufficient resources for installing the add-on.
- Only pay-per-use VM nodes can be added or removed by autoscaler.

Installing the Add-on

**Step 1** Log in to the CCE console. In the navigation pane, choose Add-ons. On the Add-on Marketplace tab page, click Install Add-on under autoscaler.

**Step 2** On the Install Add-on page, select the cluster and the add-on version, and click Next: Configuration.

**Step 3** Configure add-on installation parameters listed in Table 14-4.

### Table 14-4 Basic settings

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Add-on Version</th>
<th>Description</th>
</tr>
</thead>
</table>
| Add-on Specification(s)    | Available in all versions | The add-on can be deployed in the following specifications:
|                            |                 | - **Single**: The add-on is deployed with only one pod.
|                            |                 | - **HA50**: The add-on is deployed with two pods, serving a cluster with 50 nodes and ensuring high availability.
|                            |                 | - **HA200**: The add-on is deployed with two pods, serving a cluster with 50 nodes and ensuring high availability. Each pod uses more resources than those of the **HA50** specification.
<p>|                            |                 | - <strong>Custom</strong>: You can customize the number of pods and specifications as required. |
| Instances                  | Available in all versions | Number of pods that will be created to match the selected add-on specifications. The number cannot be modified. |</p>
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Add-on Version</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Container</td>
<td>Available in all versions</td>
<td>CPU and memory quotas of the container allowed for the selected add-on specifications. The quotas cannot be modified.</td>
</tr>
</tbody>
</table>
| Login Mode | Available only in certain versions | Select a login mode for the worker nodes to be added during auto scale-up. Passwords and key pairs are supported for login. If you select **Password**:  
  - **Password**: Set a password for logging in to the added worker nodes as user **root**  
  - **Confirm Password**: Enter the password again.  
  If you select **Key pair**:  
  **Key pair**: Select an existing key pair or create a new one for identity authentication during remote login to the added nodes. |
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Add-on Version</th>
<th>Description</th>
</tr>
</thead>
</table>
| Auto Scale-In                     | Available in all versions | **Off**: Auto scale-down is not allowed. Only auto scale-up is allowed. **On**: Auto scale-down is allowed for both existing and added nodes.  
- **Idle Time (min)**: Time for which a node should be unneeded before it is eligible for scale-down. Default value: 10 minutes.  
- **Resource Usage**: If the percentage of both CPU and memory usage on a node is below this threshold, auto scale-down will be triggered to delete the node from the cluster. The default value is 0.5, which means 50%.  
- **Scale-in Cooldown After Scale-out**: The time after scale-up that the scale-down evaluation will resume. Default value: 10 minutes.  
- **Scale-in Cooldown After Node Deletion**: The time after node deletion that the scale-down evaluation will resume. Default value: 10 minutes.  
- **Scale-in Cooldown After Failure**: The time after a scale-down failure that the scale-down evaluation will resume. Default value: 3 minutes. For details about the impact and relationship between the scale-in cooling intervals configured in the node pool and autoscaler, see **Scale-in Cooling Interval**.  
- **Max empty bulk delete**: The maximum number of empty nodes that can be deleted at the same time. Default value: 10.  
- **Node Recheck Timeout**: The timeout before autoscaler checks again the node that could not be previously removed. Default value: 5 minutes. |
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Add-on Version</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td><strong>NOTE</strong> If a node is consistently unneeded for a significant amount of time (default: 10 min), it will be considered for removal. However, the following pods cannot be removed from a cluster:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>● Pods that do not meet specific requirements set in <strong>PodDisruptionBudget</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>● Pods with local storage</td>
</tr>
<tr>
<td></td>
<td></td>
<td>● Pods that cannot be scheduled to other nodes due to constraints such as affinity and anti-affinity policies</td>
</tr>
<tr>
<td></td>
<td></td>
<td>● Pods that have the <strong>cluster-autoscaler.kubernetes.io/safe-to-evict: 'false'</strong> annotation</td>
</tr>
<tr>
<td></td>
<td></td>
<td>● Pods (except those created by kube-system DaemonSet) that exist in the kube-system namespace on the node</td>
</tr>
<tr>
<td></td>
<td></td>
<td>● Pods that are not created by the controller (Deployment/ReplicaSet/job/StatefulSet)</td>
</tr>
<tr>
<td>Parameter</td>
<td>Add-on Version</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------</td>
<td>----------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| Node Pool Configuration    | Available only in certain versions | Configuration of the default node pool. A node pool is a group of compute nodes with the same node type (VM or BMS), specifications, and labels. When a cluster needs to be scaled up, autoscaler will automatically add nodes from node pools to the cluster. If no custom node pool is available, autoscaler will use the default node pool. Click **Add Node Pool Configuration** and set the following parameters:  
  - **AZ**: A physical region where resources use independent power supplies and networks. AZs are physically isolated but interconnected through the internal network.  
  - **OS**: OS of the nodes to be created.  
  - **Taints**: No taints are added by default. Taints allow nodes to repel a set of pods. You can add a maximum of 10 taints for each node pool. Each taint contains the following parameters:  
    - **Key**: A key must contain 1 to 63 characters starting with a letter or digit. Only letters, digits, hyphens (-), underscores (_), and periods (.) are allowed. A DNS subdomain name can be used as the prefix of a key.  
    - **Value**: A value must start with a letter or digit and can contain a maximum of 63 characters, including letters, digits, hyphens (-), underscores (_), and periods (.).  
    - **Effect**: Available options are **NoSchedule**, **PreferNoSchedule**, and **NoExecute**.  
  NOTICE  
  - If taints are used, you must configure tolerations in the YAML files of pods. Otherwise, scale-up may fail or pods cannot be scheduled onto the added nodes.  
  - Taints cannot be modified after configuration. Incorrect taints may cause a scale-up failure or prevent pods from being scheduled onto the added nodes.  
  - **Resource Tags**: Resource tags can be added to classify resources. |
NOTE
You can create predefined tags in Tag Management Service (TMS). Predefined tags are visible to all service resources that support the tagging function. You can use these tags to improve tagging and resource migration efficiency.

- **Specifications**: CPU and memory of the added nodes.

To configure more add-on parameters, click **Advanced Settings** at the bottom of this page.

**Table 14-5 Advanced settings**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Add-on Version</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total Nodes</td>
<td>Available in all versions</td>
<td>The total number of nodes up to which the cluster can be scaled.</td>
</tr>
<tr>
<td>Total Cores</td>
<td>Available in all versions</td>
<td>The total number of cores up to which the cluster can be scaled.</td>
</tr>
<tr>
<td>Total Memory (GB)</td>
<td>Available in all versions</td>
<td>The total memory up to which the cluster can be scaled.</td>
</tr>
<tr>
<td>Auto Scale-Out</td>
<td>Available only in certain versions</td>
<td><strong>Triggered when there are pods unscheduled</strong>: Selected by default.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Utilization Scale-up</strong>: Select this option to allow auto scale-up to be triggered when cluster resource usage reaches the threshold.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <strong>Cpu Utilization</strong>: The autoscaler adds nodes if the CPU usage of the node pool reaches the specified limit.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <strong>Memory Utilization</strong>: The autoscaler adds nodes if the memory usage of the node pool reaches the specified limit.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Add-on Version</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------</td>
<td>-----------------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| Disk            | Available only in certain versions                  | **System Disk** and **Data Disk**: Set the disk space of the nodes.  
  - The system disk capacity ranges from 40 to 1,024 GB. The default value is 40 GB.  
  - The data disk capacity ranges from 100 to 32,768 GB. The default value is 100 GB.  
  If you select **Disk Space Allocation**, you can allocate data disk space for storing Docker and Kubelet resources.  
  **NOTICE**  
  - By default, disks run in the direct-lvm mode. If data disks are removed, the loop-lvm mode will be used and this will impair system stability.  
  - Docker resources include Docker images and image metadata. kubelet resources include pod configuration files, secrets, and emptyDirs.  
  - Click **Add Data Disk** to add a data disk.  
  System disks and data disks deliver three levels of I/O performance:  
  - **High I/O**: uses SAS drives to store data. EVS disks of this level provide a maximum IOPS of 3,000 and a minimum read/write latency of 1 ms. They are suitable for RDS, NoSQL, data warehouse, and file system applications.  
  - **Ultra-high I/O**: uses SSD drives to store data. EVS disks of this level provide a maximum IOPS of 20,000 and a minimum read/write latency of 1 ms. They are suitable for RDS, NoSQL, and data warehouse applications. |
| Command Injection| Available only in certain versions                  | Enter a script command.  
  **NOTE**  
  - Enter a maximum of 1,000 characters.  
  - For details about the command line injection example and method, see *Injecting User Data into ECS*. |

**Step 4**  
When the configuration is complete, click **Install**.  
After the add-on is installed, click **Go Back to Previous Page**. On the **Add-on Instance** tab page, select the corresponding cluster to view the running instance. This indicates that the add-on has been installed on each node in the cluster.  

----End
Upgrading the Add-on

**Step 1** Log in to the CCE console. In the navigation pane, choose Add-ons. On the Add-on Instance tab page, click Upgrade under autoscaler.

- **NOTE**
  - If the Upgrade button is unavailable, the current add-on is already up-to-date and no upgrade is required.
  - If the Upgrade button is available, click Upgrade to upgrade the add-on.
  - During the upgrade, the coredns add-on of the original version on cluster nodes will be discarded, and the add-on of the target version will be installed.

**Step 2** In the dialog box displayed, set parameters and upgrade the add-on. For details about the parameters, see the parameter description in Installing the Add-on.

----End

Uninstalling the Add-on

**Step 1** Log in to the CCE console. In the navigation pane, choose Add-ons. On the Add-on Instance tab page, select the target cluster and click Uninstall under autoscaler.

**Step 2** In the dialog box displayed, click Yes to uninstall the add-on.

----End

14.8 nginx-ingress

Introduction

Kubernetes uses kube-proxy to expose Services and provide load balancing. The implementation is at the transport layer. When it comes to Internet applications, where a bucket-load of information is generated, forwarding needs to be more fine-grained, precisely and flexibly controlled by policies and load balancers to deliver higher performance.

This is where ingresses enter. Ingresses provide application-layer forwarding functions, such as virtual hosts, load balancing, SSL proxy, and HTTP routing, for Services that can be directly accessed outside a cluster.

Kubernetes has officially released the Nginx-based ingress controller. nginx-ingress is an add-on that uses ConfigMaps to store Nginx configurations. The Nginx ingress controller generates Nginx configurations for an ingress and writes the configurations to the pod of Nginx through Kubernetes API. These configurations can be modified and updated by reloading.

The nginx-ingress add-on in CCE is implemented using the open-source community chart and image. CCE does not maintain the add-on. Therefore, it is not recommended that the nginx-ingress add-on be used commercially.

You can visit the open source community for more information.
### How nginx-ingress Works

nginx-ingress consists of the ingress object, ingress controller, and Nginx. The ingress controller assembles ingresses into the Nginx configuration file (nginx.conf) and reloads Nginx to make the changed configurations take effect. When it detects that the pod in a Service changes, it dynamically changes the upstream server group configuration of Nginx. In this case, the Nginx process does not need to be reloaded. Figure 14-7 shows how nginx-ingress works.

- An ingress is a group of access rules that forward requests to specified Services based on domain names or URLs. Ingresses are stored in the object storage service etcd and are added, deleted, modified, and queried through APIs.
- The ingress controller monitors the changes of resource objects such as ingresses, Services, endpoints, secrets (mainly TLS certificates and keys), nodes, and ConfigMaps in real time and automatically performs operations on Nginx.
- Nginx implements load balancing and access control at the application layer.
Constraints

- This add-on can be installed only in CCE clusters of v1.15 or later. Kunpeng clusters are not supported.
- `kubernetes.io/ingress.class: "nginx"` must be added to `annotations` of the ingress created by calling the API. If an existing ingress is connected, add `kubernetes.io/ingress.class: "cce"` to `annotations`.

Prerequisites

Before creating a workload, you must have an available cluster. If no cluster is available, create one according to Buying a CCE Cluster.

Installing the Add-on

**Step 1** Log in to the CCE console. In the navigation pane, choose Add-ons. On the Add-on Marketplace tab page, click Install Add-on under nginx-ingress.

**Step 2** On the Install Add-on page, select the cluster and the add-on version, and click Next: Configuration.

**Step 3** In the Configuration step, set the parameters listed in Table 14-6. Parameters marked with an asterisk (*) are mandatory.

### Table 14-6 nginx-ingress add-on parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add-on Specifications</td>
<td>Select add-on specifications based on service requirements. You can customize resource specifications.</td>
</tr>
<tr>
<td>Instances</td>
<td>Number of pods that will be created to match the selected add-on specifications.</td>
</tr>
<tr>
<td>Container</td>
<td>CPU and memory quotas of the container allowed for the selected add-on specifications.</td>
</tr>
</tbody>
</table>

**NOTE**

- Ensure that there are sufficient nodes in the cluster. If not, the add-on instance cannot be scheduled. In this case, you need to reinstall the add-on.
- The request value must be less than or equal to the limit value. Otherwise, the creation fails.
- You are advised to set the request value to be the same as the limit value. If nodes are insufficient, containers whose request value is less than the limit value are preferentially cleared.
- For details about the performance results of different configurations, see the Nginx performance test report.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ConfigMap</td>
<td><strong>Config</strong> The configuration takes effect globally. This parameter is generated by configuring the <code>nginx.conf</code> file and affects all managed ingresses. You can search for related parameters in the <strong>ConfigMap</strong>. If your configuration is not included in the options listed in the ConfigMap, your configuration will not take effect. <strong>Example:</strong></td>
</tr>
</tbody>
</table>
|                      | - **worker-processes**: number of worker processes when Nginx provides web services for external systems. The default value is **auto**.  
- **max-worker-connections**: maximum number of concurrent connections of a single worker process. The default value is **16384**.  
- **keep-alive**: timeout interval for a keep-alive connection, in unit of seconds. The default value is **75**. |
| Custom Header        | By default, Nginx filters out custom headers. This parameter allows you to redefine or add request headers to be sent to backend servers.                                                                     |
| Default backend      | enabled The nginx-ingress add-on provides the 404 backend service by default. If you need to customize the 404 backend service, enter a value in format of `<namespace/serviceName>`.                               |
| Elastic Load         | Balancer You can select an existing public or private network load balancer. This function enables the traffic from a public or private network to be forwarded to the Service backing the add-on.  
**Once this parameter is set, do not modify the configuration on the ELB console. Otherwise, the Service will be abnormal.** If you have modified the configuration, uninstall the add-on and reinstall it.  
**NOTE**  
- Ensure that the load balancer you select or create is in the same VPC as the cluster and routes requests over the Internet.  
- The load balancer has at least two listeners, and ports 80 and 443 are not occupied by listeners. |

**Step 4** After the configuration is complete, click **Install**. After the add-on is installed, click **Back to Add-on List**.
Step 5  On the Add-on Instance tab page, select the corresponding cluster. If the installed add-on is displayed and in running state, it has been successfully installed in the current cluster.

---End

Upgrading the Add-on

Step 1  Log in to the CCE console. In the navigation pane, choose Add-ons. On the Add-on Instance tab page, click Upgrade under nginx-ingress.

NOTE

- If the Upgrade button is not available, the current add-on is already up-to-date and no upgrade is required.
- During the upgrade, the nginx-ingress add-on of the original version on cluster nodes will be discarded, and the add-on of the target version will be installed.
- After the upgrade, nginx-ingress will be restarted. Related services may be interrupted. Therefore, you are advised to upgrade the add-on during off-peak hours.

Step 2  On the Basic Information page, select the add-on version and click Next.

Step 3  In the Configuration step, set the parameters listed in Table 14-6. Parameters marked with an asterisk (*) are mandatory.

Step 4  After setting the parameters, click Upgrade to upgrade the add-on.

----End
Uninstalling the Add-on

Step 1 Log in to the CCE console. In the navigation pane, choose Add-ons. On the Add-on Instance tab page, click Uninstall under nginx-ingress.

Step 2 In the dialog box displayed, click Yes to uninstall the add-on.

---End

14.9 metrics-server

From version 1.8 onwards, Kubernetes provides resource usage metrics, such as the container CPU and memory usage, through the Metrics API. These metrics can be directly accessed by users (for example, by using the kubectl top command) or used by controllers (for example, Horizontal Pod Autoscaler) in a cluster for decision-making. The specific component is metrics-server, which is used to substitute for heapster for providing the similar functions. heapster has been gradually abandoned since v1.11.

metrics-server is an aggregator for monitoring data of core cluster resources. You can quickly install this add-on on the CCE console.

After metrics-server is installed, you can create an HPA policy on the Workload Scaling tab page of the Auto Scaling page. For details, see Creating an HPA Policy for Workload Auto Scaling.

The official community project and documentation are available at https://github.com/kubernetes-sigs/metrics-server.

Notes and Constraints

This add-on can be installed only in CCE clusters of v1.13 or later. Kunpeng clusters are not supported.

Installing the Add-on

Step 1 Log in to the CCE console. In the navigation pane, choose Add-ons. On the Add-on Marketplace tab page, click Install Add-on under metrics-server.

Step 2 On the Install Add-on page, select the cluster and the add-on version, and click Next: Configuration.

Step 3 Select Single or HA for Add-on Specifications, and click Install.

After the add-on is installed, click Go Back to Previous Page. On the Add-on Instance tab page, select the corresponding cluster to view the running instance. This indicates that the add-on has been installed on each node in the cluster.

---End

Upgrading the Add-on

Step 1 Log in to the CCE console. In the navigation pane, choose Add-ons. On the Add-on Instance tab page, click Upgrade under metrics-server.
If the Upgrade button is not available, the current add-on is already up-to-date and no upgrade is required. During the upgrade, the metrics-server add-on of the original version on cluster nodes will be discarded, and the add-on of the target version will be installed.

Step 2 On the Basic Information page, select the add-on version and click Next.

Step 3 Set the parameters by referring to the parameter description in Installing the Add-on and click Upgrade.

Uninstalling the Add-on

Step 1 Log in to the CCE console. In the navigation pane, choose Add-ons. On the Add-on Instance tab page, click Uninstall under metrics-server.

Step 2 In the dialog box displayed, click Yes to uninstall the add-on.

14.10 cce-hpa-controller

cci-hpa-controller is a Huawei-developed add-on, which can be used to flexibly scale in or out Deployments based on metrics such as CPU usage and memory usage.

After installing this add-on, you can create a CustomedHPA policy on the Workload Scaling tab page of the Auto Scaling page. For details, see Creating a CustomedHPA Policy for Workload Auto Scaling.

Main Functions

- Scaling can be performed based on the percentage of the current number of pods.
- The minimum scaling step can be set.
- Different scaling operations can be performed based on the actual metric values.

Notes and Constraints

- This add-on can be installed only in CCE clusters of v1.15 or later. Kunpeng clusters are not supported.
- After installing cce-hpa-controller, you must also install the prometheus add-on (1.1.0 or later) for metric collection.

Installing the Add-on

Step 1 Log in to the CCE console. In the navigation pane, choose Add-ons. On the Add-on Marketplace tab page, click Install Add-on under cce-hpa-controller.

Step 2 On the Install Add-on page, select the cluster and the add-on version, and click Next: Configuration.
Step 3  Select **Single** or **Custom** for **Add-on Specifications**, and click **Install**. 

After the add-on is installed, click **Go Back to Previous Page**. On the **Add-on Instance** tab page, select the corresponding cluster to view the running instance. This indicates that the add-on has been installed on each node in the cluster.

#### End

**Upgrading the Add-on**

**Step 1**  Log in to the CCE console. In the navigation pane, choose **Add-ons**. On the **Add-on Instance** tab page, click **Upgrade** under **cce-hpa-controller**.

- **NOTE**
  - If the **Upgrade** button is not available, the current add-on is already up-to-date and no upgrade is required.
  - During the upgrade, the cce-hpa-controller add-on of the original version on cluster nodes will be discarded, and the add-on of the target version will be installed.

**Step 2**  Set the parameters by referring to the parameter description in **Installing the Add-on** and click **Upgrade**.

#### End

**Uninstalling the Add-on**

**Step 1**  Log in to the CCE console. In the navigation pane, choose **Add-ons**. On the **Add-on Instance** tab page, click **Uninstall** under **cce-hpa-controller**.

**Step 2**  In the dialog box displayed, click **Yes** to uninstall the add-on.

#### End

**14.11 prometheus**

**Introduction**

Prometheus is an open-source system monitoring and alerting framework. It is derived from Google's borgmon monitoring system, which was created by former Google employees working at SoundCloud in 2012. Prometheus was developed as an open-source community project and officially released in 2015. In 2016, Prometheus officially joined the Cloud Native Computing Foundation, after Kubernetes.

CCE allows you to quickly install Prometheus as an add-on.

Official website of Prometheus: [https://prometheus.io/](https://prometheus.io/)

Open source community: [https://github.com/prometheus/prometheus/prometheus](https://github.com/prometheus/prometheus/prometheus)

**Features**

As a next-generation monitoring framework, Prometheus has the following features:
- Powerful multi-dimensional data model
  - Time series data is identified by metric name and key-value pair.
  - Multi-dimensional labels can be set for all metrics.
  - Data models do not require dot-separated character strings.
  - Data models can be aggregated, cut, and sliced.
  - The double floating-point format is supported. Labels can all be set to unicode.
- Flexible and powerful query statement (PromQL): One query statement supports addition, multiplication, and connection for multiple metrics.
- Easy to manage: The Prometheus server is a separate binary file that can work locally. It does not depend on distributed storage.
- Efficient: Each sampling point occupies only 3.5 bytes, and one Prometheus server can process millions of metrics.
- The pull mode is used to collect time series data, which facilitates local tests and prevents faulty servers from pushing bad metrics.
- Time series data can be pushed to the Prometheus server in push gateway mode.
- Users can obtain the monitored targets through service discovery or static configuration.
- Multiple visual GUIs are available.
- Easy to scale

As collected data may be lost, Prometheus is not applicable if there is a high requirement on accuracy of the collected data. However, Prometheus has great query advantages if it is used to record time series data. In addition, Prometheus is applicable to the microservice architecture.

Notes and Constraints

This add-on can be installed only in CCE clusters of v1.11 or later. Kunpeng clusters are not supported.

Installing the Add-on

**Step 1** Log in to the CCE console. In the navigation pane, choose Add-ons. On the Add-on Marketplace tab page, click Install Add-on under prometheus.

**Step 2** On the Install Add-on page, select the cluster and the add-on version, and click Next: Configuration.

**Step 3** In the Configuration step, set the following parameters:
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Add-on Specifications</strong></td>
<td>Select an add-on specification based on service requirements. The options are as follows:</td>
</tr>
<tr>
<td></td>
<td>● <strong>Demo</strong>(&lt;= 100 containers): The specification type is applicable to the experience and function demonstration environment. In this specification, Prometheus occupies few resources but has limited processing capabilities. You are advised to use this specification when the number of containers in the cluster does not exceed 100.</td>
</tr>
<tr>
<td></td>
<td>● <strong>Small</strong>(&lt;= 2000 containers): You are advised to use this specification when the number of containers in the cluster does not exceed 2,000.</td>
</tr>
<tr>
<td></td>
<td>● <strong>Medium</strong>(&lt;= 5000 containers): You are advised to use this specification when the number of containers in the cluster does not exceed 5,000.</td>
</tr>
<tr>
<td></td>
<td>● <strong>Large</strong>(&gt; 5000 containers): You are advised to use this specification when the number of containers in the cluster exceeds 5,000.</td>
</tr>
<tr>
<td><strong>Instances</strong></td>
<td>Number of pods that will be created to match the selected add-on specifications. The number cannot be modified.</td>
</tr>
<tr>
<td><strong>Container</strong></td>
<td>CPU and memory quotas of the container allowed for the selected add-on specifications. The quotas cannot be modified.</td>
</tr>
<tr>
<td><strong>Remote Write</strong></td>
<td>Select a value.</td>
</tr>
<tr>
<td></td>
<td>● <strong>Local</strong>: Data collected by the prometheus add-on is stored only in local data disks.</td>
</tr>
<tr>
<td></td>
<td>● <strong>CIE</strong>: Data collected by the prometheus add-on is stored in both local data disks and CIE.</td>
</tr>
<tr>
<td></td>
<td>● <strong>Custom</strong>: Data collected by the prometheus add-on is stored in both local data disks and a custom remote end. The remote end address and HTTPS authentication information need to be obtained from third-party services.</td>
</tr>
<tr>
<td><strong>Monitoring Data Retention Period</strong></td>
<td>Number of days for storing customized monitoring data. The default value is 15 days.</td>
</tr>
</tbody>
</table>
### Parameter Description

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| Storage   | Set the following parameters as prompted:  
  - **Type**: EVS is supported.  
  - **AZ**: Set this parameter based on the site requirements. An AZ is a physical region where resources use independent power supply and networks. AZs are physically isolated but interconnected through an internal network.  
  - **Disk Type**: Common I/O, high I/O, and ultra-high I/O are supported. For details about the comparison among these disk types, see [System Disks and Data Disks](#).  
  - **Capacity**: Enter the storage capacity based on service requirements. The default value is 10 GB.  

**NOTE**  
If a PVC already exists in the namespace monitoring, the configured storage will be used as the storage source.

---

#### Step 4  
Click **Install**.

After the add-on is installed, click **Go Back to Previous Page**. On the **Add-on Instance** tab page, select the corresponding cluster to view the running instance. This indicates that the add-on has been installed on each node in the cluster.

#### Step 5  
In the navigation pane on the left, choose **Add-ons**. On the **Add-on Instance** tab page, click **prometheus** to view details about the add-on instance.

----End

### Upgrading the Add-on

#### Step 1  
Log in to the CCE console. In the navigation pane, choose **Add-ons**. On the **Add-on Instance** tab page, click **Upgrade** under **prometheus**.

**NOTE**
- If the **Upgrade** button is not available, the current add-on is already up-to-date and no upgrade is required.
- During the upgrade, the prometheus add-on of the original version on cluster nodes will be discarded, and the add-on of the target version will be installed.

#### Step 2  
On the **Basic Information** page, select the add-on version and click **Next**.

#### Step 3  
Set the parameters by referring to the parameter description in [Installing the Add-on](#) and click **Upgrade**.

----End

### Uninstalling the Add-on

#### Step 1  
Log in to the CCE console. In the navigation pane, choose **Add-ons**. On the **Add-on Instance** tab page, click **Uninstall** under **prometheus**.
Step 2  In the dialog box displayed, click Yes to uninstall the add-on.

----End

Reference

- For details about the Prometheus concepts and configurations, see the Prometheus Official Documentation.
- For details about how to install Node Exporter, see the node_exporter GitHub.
- For details about how to send Slack messages, see Incoming Webhooks.

14.12 web-terminal

The web-terminal add-on is a lightweight terminal server that allows you to use kubectl on the web UI. It provides a remote command-line interface (CLI) via web browser and HTTP, and can be easily integrated into an independent system. You can directly access the add-on as a service to obtain information and log in to a server through cmdb.

web-terminal can run on all operating systems supported by Node.js and does not depend on local modules. It is fast and easy to install and supports multiple sessions.

Open source community: https://github.com/rabchev/web-terminal

Notes and Constraints

- This add-on can be installed only in clusters of v1.9 or later. Kunpeng clusters are not supported.
- This add-on is currently in the beta phase. Cloudshell will replace this add-on in the future.
- When installing web-terminal to use kubectl, you must log in using your cloud account or as an IAM user with the CCE Administrator permission. For details about how to control the kubectl permission, see Controlling web-terminal Permissions.
- This add-on can be deployed only on x86 nodes. On Kunpeng nodes, you can directly connect to your cluster through kubectl.

Precautions

The web-terminal add-on can be used to manage CCE clusters. Keep the login password secure to prevent unexpected operation.

Installing the Add-on

Step 1  Log in to the CCE console. In the navigation pane, choose Add-ons. On the Add-on Marketplace tab page, click Install Add-on under web-terminal.

Step 2  On the Install Add-on page, select the cluster and the add-on version, and click Next: Configuration.
Step 3  On the Configuration page, set the following parameters:

- **Username**: The default value is root and cannot be changed.
- **Password**: password for logging in to web-terminal. Keep secure the password. The web-terminal add-on can be used to manage CCE clusters. Keep the login password secure to prevent unexpected operation.
- **Confirm Password**: Enter the password again.
- **Access Type**
  - **NodePort**: The web-terminal add-on is accessed in the NodePort mode by default and can be used only if any node in the cluster has an EIP. If this access type is selected, an EIP must be bound to the cluster where web-terminal will be installed.
  - **LoadBalancer (ELB)**: If this access type is selected, an ELB must be selected. If no load balancer is available, create a shared load balancer and click the refresh icon. Make sure that the load balancer you select or create is in the same VPC as the cluster and routes requests over the Internet.
- **Port Settings**: mandatory if the access type is LoadBalancer (ELB).
  - **Protocol**: The value is fixed to TCP.
  - **Container Port**: port on which the workload in the container image listens. The default value is used and cannot be replaced by another value.
  - **Access Port**: port mapped to the container port at the load balancer's IP address. The workload can be accessed at <load balancer's IP address>:<access port>. The port number range is 1–65535.

Step 4  Click Install.

After the add-on is installed, click Go Back to Previous Page. On the Add-on Instance tab page, select the corresponding cluster to view the running instance. This indicates that the add-on has been installed on each node in the cluster.

---End

Connecting to a Cluster Using the web-terminal Add-on

**Method 1**

**Step 1**  Log in to the CCE console. In the navigation pane, choose Resource Management > Clusters.

**Step 2**  Click Command Line Tool > web-terminal to go to the add-on instance details page.
Step 3  Click the hyperlink in the Access Address field to access the add-on.

Controlling web-terminal Permissions

After web-terminal is installed, kubectl uses the ClusterRole cluster-admin by default and can operate Kubernetes resources in the cluster. If you need to manually change to another ClusterRole, you can run kubectl edit clusterrolebinding web-terminal to modify the web-terminal ServiceAccount.

For details about ClusterRole and ClusterRoleBinding, see Namespace Permissions (Kubernetes RBAC-based).
NOTICE

- Manually configured web-terminal permissions could be reset after the web-terminal add-on is upgraded. You are advised to back up the configurations before the upgrade.
- Before using kubectl to modify ClusterRoleBindings, ensure that kubectl has been configured with the required permissions.

Upgrading the Add-on

Step 1 Log in to the CCE console. In the navigation pane, choose Add-ons. On the Add-on Instance tab page, click Upgrade under web-terminal.

NOTE

- If the Upgrade button is not available, the current add-on is already up-to-date and no upgrade is required.
- During the upgrade, the web-terminal add-on of the original version on cluster nodes will be discarded, and the add-on of the target version will be installed.

Step 2 On the Basic Information page, select the add-on version and click Next.

Step 3 Set the parameters by referring to the parameter description in Installing the Add-on and click Upgrade.

----End

Uninstalling the Add-on

Step 1 Log in to the CCE console. In the navigation pane, choose Add-ons. On the Add-on Instance tab page, click Uninstall under web-terminal.

Step 2 In the dialog box displayed, click Yes to uninstall the add-on.

----End

14.13 gpu-beta

Introduction

gpu-beta is a device management add-on that supports GPUs in containers. It supports only NVIDIA drivers.

Notes and Constraints

- This add-on can be installed only in CCE clusters of v1.11 or later. Kunpeng clusters are not supported.
- If GPU nodes are used in the cluster, the gpu-beta add-on must be installed.
- The driver to be downloaded must be a .run file.
• If the download link is a public network address, for example, https://us.download.nvidia.com/tesla/396.37/NVIDIA-Linux-x86_64-396.37.run, bind an EIP to each GPU node. For details about how to obtain the driver link, see Obtaining the Driver Link from Public Network.

• If the download link is an OBS URL, you do not need to bind an EIP to GPU nodes. For details about how to obtain the driver link, see Obtaining the Driver Link from OBS.

• Ensure that the NVIDIA driver version matches the GPU node.

• After the driver version is changed, restart the node for the change to take effect.

---End

Obtaining the Driver Link from Public Network


Step 2 Select the driver information on the NVIDIA Driver Downloads page, as shown in Figure 14-12. Operating System must be Linux 64-bit.
Step 3 After confirming the driver information, click SEARCH. A page is displayed, showing the driver information, as shown in Figure 14-13. Click DOWNLOAD.

Step 4 Obtain the driver link in either of the following ways:

- Method 1: As shown in Figure 14-14, find url=/tesla/396.37/NVIDIA-Linux-x86_64-396.37.run in the browser address box. Then, supplement it to obtain the driver link https://us.download.nvidia.com/tesla/396.37/NVIDIA-Linux-x86_64-396.37.run. By using this method, you must bind an EIP to each GPU node.

- Method 2: As shown in Figure 14-14, click AGREE & DOWNLOAD to download the driver. Then, upload the driver to OBS and record the OBS URL. By using this method, you do not need to bind an EIP to GPU nodes.
Obtaining the Driver Link from OBS

**Step 1** Upload the driver to OBS and set the driver file to public read. For details, see [Uploading a File](#).

**Step 2** In the navigation pane on the OBS console, select **Object Storage**.

**Step 3** In the bucket list, click a bucket name, and then the **Overview** page of the bucket is displayed.

**Step 4** In the navigation pane, choose **Objects**.

**Step 5** Select the target object and copy the driver link on the object details page.

---End

Upgrading the Add-on

**Step 1** Log in to the CCE console. In the navigation pane, choose **Add-ons**. On the **Add-on Instance** tab page, click **Upgrade** under **gpu-beta**.

⚠️ **NOTE**

- If the **Upgrade** button is not available, the current add-on is already up-to-date and no upgrade is required.
- During the upgrade, the gpu-beta add-on of the original version on cluster nodes will be discarded, and the add-on of the target version will be installed.

**Step 2** On the **Basic Information** page, select the add-on version and click **Next**.

**Step 3** Set the parameters by referring to the parameter description in [Installing the Add-on](#) and click **Upgrade**.
Step 4  (Mandatory) Restart the node.

Restart the node on the ECS console. Log in to the management console, select the region where the ECS is located, and choose Service List > Computing > Elastic Cloud Server. In the ECS list, locate the target node, and click More > Restart in the Operation column.

---End

Uninstalling the Add-on

Step 1  Log in to the CCE console. In the navigation pane, choose Add-ons. On the Add-on Instance tab page, select the cluster and click Uninstall under gpu-beta.

Step 2  In the dialog box displayed, click Yes to uninstall the add-on.

NOTE

The driver will not be uninstalled during gpu-beta add-on uninstall. If the driver is reinstalled, you must restart all GPU nodes.

---End

Helpful Links

- How Do I Troubleshoot gpu-beta and GPU Driver Problems
- What Should I Do If GPU Node Exceptions Occur?

14.14 huawei-npu

Introduction

huawei-npu is a management add-on for Huawei NPU devices in containers.

Notes and Constraints

- This add-on can be installed in CCE clusters and Kunpeng clusters of v1.13 or later.
- If Ascend-accelerated nodes are used in the cluster, the huawei-npu add-on must be installed.

Installing the Add-on

Step 1  Log in to the CCE console. In the navigation pane, choose Add-ons. On the Add-on Marketplace tab page, click Install Add-on under huawei-npu.

Step 2  On the Install Add-on page, select the cluster and the add-on version, and click Next: Configuration.

Step 3  Click Install to directly install the add-on. Currently, the huawei-npu add-on has no configurable parameters.
After the add-on is installed, click **Go Back to Previous Page**. On the **Add-on Instance** tab page, select the corresponding cluster to view the running instance. This indicates that the add-on has been installed on each node in the cluster.

### Upgrading the Add-on

**Step 1**  
Log in to the CCE console. In the navigation pane, choose **Add-ons**. On the **Add-on Instance** tab page, click **Upgrade** under **huawei-npu**.

**NOTE**  
- If the **Upgrade** button is not available, the current add-on is already up-to-date and no upgrade is required.
- During the upgrade, the huawei-npu add-on of the original version on cluster nodes will be discarded, and the add-on of the target version will be installed.

**Step 2**  
On the **Basic Information** page, select the add-on version and click **Next**.

**Step 3**  
Click **Upgrade**.

### Uninstalling the Add-on

**Step 1**  
Log in to the CCE console. In the navigation pane, choose **Add-ons**. On the **Add-on Instance** tab page, click **Uninstall** under **huawei-npu**.

**Step 2**  
In the dialog box displayed, click **Yes** to uninstall the add-on.

### 14.15 volcano

#### Introduction

Volcano is a batch processing platform based on Kubernetes. It provides a series of features required by machine learning, deep learning, bioinformatics, genomics, and other big data applications, as a powerful supplement to Kubernetes capabilities.

Volcano is derived from the HUAWEI CLOUD high-performance batch computing solution. It plays an important role in the stable running of HUAWEI CLOUD services such as ModelArts and CCI. Volcano provides general-purpose, high-performance computing capabilities, such as job scheduling engine, heterogeneous chip management, and job running management, serving end users through computing frameworks for different industries, such as AI, big data, gene sequencing, and rendering. (Volcano has been open-sourced in GitHub.)

Volcano provides job scheduling, job management, and queue management for computing applications. Its main features are as follows:

- Diverse computing frameworks, such as TensorFlow, MPI, and Spark, can run on Kubernetes in containers. Common APIs for batch computing jobs through CRD, various plug-ins, and advanced job lifecycle management are provided.
- Advanced scheduling capabilities are provided for batch computing and high-performance computing scenarios, including group scheduling, preemptive priority scheduling, packing, resource reservation, and task topology.
- Queues can be effectively managed for scheduling jobs. Complex job scheduling capabilities such as queue priority and multi-level queues are supported.

Open source community: [https://github.com/volcano-sh/volcano](https://github.com/volcano-sh/volcano)

**Notes and Constraints**

This add-on can be installed in CCE clusters and Kunpeng clusters of v1.13 or later.

**Installing the Add-on**

**Step 1** Log in to the CCE console. In the navigation pane, choose Add-ons. On the Add-on Marketplace tab page, click Install Add-on under volcano.

**Step 2** On the Install Add-on page, select the cluster and the add-on version, and click Next: Configuration.

**Step 3** Click Install to directly install the add-on. Currently, the volcano add-on has no configurable parameters. After the add-on is installed, click Go Back to Previous Page. On the Add-on Instance tab page, select the corresponding cluster to view the running instance. This indicates that the add-on has been installed on each node in the cluster.

----End

**Upgrading the Add-on**

**Step 1** Log in to the CCE console. In the navigation pane, choose Add-ons. On the Add-on Instance tab page, click Upgrade under volcano.

**NOTE**
- If the Upgrade button is not available, the current add-on is already up-to-date and no upgrade is required.
- During the upgrade, the volcano add-on of the original version on cluster nodes will be discarded, and the add-on of the target version will be installed.

**Step 2** On the Basic Information page, select the add-on version and click Next.

**Step 3** Click Upgrade.

----End

**Uninstalling the Add-on**

**Step 1** Log in to the CCE console. In the navigation pane, choose Add-ons. On the Add-on Instance tab page, click Uninstall under volcano.

**Step 2** In the dialog box displayed, click Yes to uninstall the add-on.

----End
14.16 spark-operator

Introduction

Spark Operator is an operator who manages the lifecycle of Apache Spark applications on Kubernetes. It aims to specify and run Spark applications (workloads) as easily as running other types of workloads on Kubernetes.

Paired with Spark Operator, Kubernetes can better control and manage the lifecycle of Spark applications, including application status monitoring, log obtaining, and application running control. This allows the Spark on Kubernetes solution to be fully integrated with Kubernetes.

Spark Operator consists of the following components:

1. SparkApplication controller: This controller is used to create, update, and delete SparkApplication objects. It also monitors events and performs actions accordingly.
2. Submission Runner: This component invokes spark-submit to submit Spark jobs. The job submission process is the same as the Spark on Kubernetes solution.
3. Spark Pod Monitor: This component monitors the status of pods related to Spark jobs and synchronizes the status to the controller.
4. Muting Admission Webhook: (Optional) This component implements some customized requirements of the driver and executor pods based on annotations.
5. SparkCtl: command line tool for interacting with the Spark Operator.

In addition to basic job submission, Spark Operator also supports the following features:

- Declarative job management;
- Automatic job re-submission after a SparkApplication object is updated;
- Configurable restart policies;
- Retry for a failed task submission;
- Integration with Prometheus to collect and forward Spark application-level metrics and driver/executor pod metrics to Prometheus.


Notes and Constraints

This add-on can be installed only in CCE clusters of v1.13.

Installing the Add-on

**Step 1** Log in to the CCE console. In the navigation pane, choose Add-ons. On the Add-on Marketplace tab page, click Install Add-on under spark-operator.
Step 2  On the Install Add-on page, select the cluster and the add-on version, and click Next: Configuration.

Step 3  Click Install to directly install the add-on. Currently, the spark-operator add-on has no configurable parameters.

After the add-on is installed, click Go Back to Previous Page. On the Add-on Instance tab page, select the corresponding cluster to view the running instance. This indicates that the add-on has been installed on each node in the cluster.

----End

Upgrading the Add-on

Step 1  Log in to the CCE console. In the navigation pane, choose Add-ons. On the Add-on Instance tab page, click Upgrade under spark-operator.

☐ NOTE

- If the Upgrade button is not available, the current add-on is already up-to-date and no upgrade is required.
- During the upgrade, the spark-operator add-on of the original version on cluster nodes will be discarded, and the add-on of the target version will be installed.

Step 2  On the Basic Information page, select the add-on version and click Next.

Step 3  Click Upgrade.

----End

Uninstalling the Add-on

Step 1  Log in to the CCE console. In the navigation pane, choose Add-ons. On the Add-on Instance tab page, click Uninstall under spark-operator.

Step 2  In the dialog box displayed, click Yes to uninstall the add-on.

----End

14.17 tf-operator

Introduction

tf-operator is an operator who manages the lifecycle of TensorFlow applications on Kubernetes. It aims to specify and run TensorFlow applications (workloads) as easily as running other types of workloads on Kubernetes.

Official project introduction and documentation: https://github.com/kubeflow/tf-operator

Notes and Constraints

- This add-on can be installed only in CCE clusters of v1.13 and v1.15.
- Some regions do not support this add-on. For details, refer to the console.
Installing the Add-on

Step 1 Log in to the CCE console. In the navigation pane, choose Add-ons. On the Add-on Marketplace tab page, click Install Add-on under tf-operator.

Step 2 On the Install Add-on page, select the cluster and the add-on version, and click Next: Configuration.

Step 3 Click Install to directly install the add-on. Currently, the tf-operator add-on has no configurable parameters.

After the add-on is installed, click Go Back to Previous Page. On the Add-on Instance tab page, select the corresponding cluster to view the running instance. This indicates that the add-on has been installed on each node in the cluster.

Upgrading the Add-on

Step 1 Log in to the CCE console. In the navigation pane, choose Add-ons. On the Add-on Instance tab page, click Upgrade under tf-operator.

**NOTE**
- If the Upgrade button is not available, the current add-on is already up-to-date and no upgrade is required.
- During the upgrade, the tf-operator add-on of the original version on cluster nodes will be discarded, and the add-on of the target version will be installed.

Step 2 On the Basic Information page, select the add-on version and click Next.

Step 3 Click Upgrade.

--- End

Uninstalling the Add-on

Step 1 Log in to the CCE console. In the navigation pane, choose Add-ons. On the Add-on Instance tab page, click Uninstall under tf-operator.

Step 2 In the dialog box displayed, click Yes to uninstall the add-on.

--- End

14.18 rc-recycler

Introduction
rc-recycler periodically deletes unnecessary resources from clusters to ensure optimal cluster performance. You can adjust the recycling policy when creating or running rc-recycler.

Notes and Constraints
- This add-on can be installed only in CCE clusters of v1.9 and v1.11.
Currently, the following features are available: parallel recycling of jobs, log dumping, statistical analysis of recycling, and incremental list caching. Only jobs in the default namespace can be cleared.

Installing the Add-on

**Step 1** Log in to the CCE console. In the navigation pane, choose Add-ons. On the Add-on Marketplace tab page, click Install Add-on under rc-recycler.

**Step 2** On the Install Add-on page, select the cluster and the add-on version, and click Next: Configuration.

**Step 3** Set add-on installation parameters listed in Table 14-8.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scanning Interval (s)</td>
<td>Interval for scanning resources. Unit: second.</td>
</tr>
<tr>
<td>Retention Period upon Completion (min)</td>
<td>Period for retaining job resources in the cluster after a job is completed.</td>
</tr>
<tr>
<td></td>
<td>● -1: Permanently retain resources of a completed job.</td>
</tr>
<tr>
<td></td>
<td>● 0: Recycle resources immediately after a job is completed.</td>
</tr>
<tr>
<td>Retention Period upon Failure (min)</td>
<td>Period for retaining job resources in the cluster after a job fails.</td>
</tr>
<tr>
<td></td>
<td>● -1: Permanently retain resources of a failed job.</td>
</tr>
<tr>
<td></td>
<td>● 0: Recycle resources immediately after a job fails.</td>
</tr>
</tbody>
</table>

**Step 4** When the configuration is complete, click Install.

After the add-on is installed, click Go Back to Previous Page. On the Add-on Instance tab page, select the corresponding cluster to view the running instance. This indicates that the add-on has been installed on each node in the cluster.

---End

Upgrading the Add-on

**Step 1** Log in to the CCE console. In the navigation pane, choose Add-ons. On the Add-on Instance tab page, click Upgrade under rc-recycler.

**NOTE**

- If the Upgrade button is not available, the current add-on is already up-to-date and no upgrade is required.
- During the upgrade, the rc-recycler add-on of the original version on cluster nodes will be discarded, and the add-on of the target version will be installed.

**Step 2** On the Basic Information page, select the add-on version and click Next.
Step 3  Set the parameters by referring to the parameter description in Installing the Add-on and click Upgrade.

----End

Uninstalling the Add-on

Step 1  Log in to the CCE console. In the navigation pane, choose Add-ons. On the Add-on Instance tab page, click Uninstall under rc-recycler.

Step 2  In the dialog box displayed, click Yes to uninstall the add-on.

----End
15.1 Overview

Auto scaling is a service that automatically and economically adjusts service resources based on your service requirements and configured policies.

Context

More and more applications are developed based on Kubernetes. It becomes increasingly important to quickly scale out applications on Kubernetes to cope with service peaks and to scale in applications during off-peak hours to save resources and reduce costs.

In a Kubernetes cluster, auto scaling involves pods and nodes. A pod is an application instance. Each pod contains one or more containers and runs on a node (VM or bare-metal server). If a cluster does not have sufficient nodes to run new pods, you need to add nodes to the cluster to ensure service running.

In CCE, auto scaling is used for online services, large-scale computing and training, deep learning GPU or shared GPU training and inference, periodic load changes, and many other scenarios.

Auto Scaling in CCE

CCE supports auto scaling for workloads and nodes.

- **Workload scaling**: Auto scaling at the scheduling layer to change the scheduling capacity of workloads. For example, you can use the HPA, a scaling component at the scheduling layer, to adjust the number of replicas of an application. Adjusting the number of replicas changes the scheduling capacity occupied by the current workload, thereby enabling scaling at the scheduling layer.

- **Node scaling**: Auto scaling at the resource layer. When the planned cluster nodes cannot allow workload scheduling, ECS resources are provided to support scheduling.

Workload scaling and node scaling can work separately or together. For details, see Using HPA and CA for Auto Scaling of Workloads and Nodes.
**Components**

Workload scaling components are described as follows:

**Table 15-1 Workload scaling components**

<table>
<thead>
<tr>
<th>Type</th>
<th>Component Name</th>
<th>Component Description</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>HPA</td>
<td><strong>metrics-server</strong></td>
<td>A built-in component of Kubernetes, which enables horizontal scaling of pods. It adds the application-level cooldown time window and scaling threshold functions based on the HPA.</td>
<td>Creating an HPA Policy for Workload Auto Scaling</td>
</tr>
<tr>
<td>Customed HPA</td>
<td><strong>cce-hpa-controller</strong></td>
<td>An enhanced auto scaling feature developed by HUAWEI CLOUD, used for auto scaling of Deployments based on metrics (CPU usage and memory usage) or at a periodic interval (a specific time point every day, every week, every month, or every year).</td>
<td>Creating a Customed HPA Policy for Workload Auto Scaling</td>
</tr>
</tbody>
</table>
Node scaling components are described as follows:

<table>
<thead>
<tr>
<th>Component Name</th>
<th>Component Description</th>
<th>Application Scenario</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>prometheus</td>
<td>An open-source system monitoring and alarm framework, which collects public metrics (CPU usage and memory usage) of kubelet in the Kubernetes cluster.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Table 15-2** Node scaling components

15.2 Scaling a Workload

15.2.1 Workload Scaling Mechanisms

CCE supports HPA and CustomedHPA policies for workload scaling. The following table describes the differences between these two types of policies.

**Table 15-3** Comparison between HPA and CustomedHPA policies

<table>
<thead>
<tr>
<th>Item</th>
<th>HPA Policy</th>
<th>CustomedHPA Policy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Implemention</td>
<td>Kubernetes Horizontal Pod Autoscaling</td>
<td>Enhanced auto scaling capabilities</td>
</tr>
<tr>
<td>Rules</td>
<td>Scales Deployments based on metrics (CPU usage and memory usage).</td>
<td>Scales Deployments based on metrics (CPU usage and memory usage) or at a periodic interval (a specific time point every day, every week, every month, or every year).</td>
</tr>
</tbody>
</table>
### How HPA Works

HPA is a controller that controls horizontal pod scaling. HPA periodically checks the pod metrics, calculates the number of replicas required to meet the target values configured for HPA resources, and then adjusts the value of the `replicas` field in the target resource object (such as a Deployment).

A prerequisite for auto scaling is that your container running data can be collected, such as number of cluster nodes/pods, and CPU and memory usage of containers. Kubernetes does not provide such monitoring capabilities itself. You can use extensions to monitor and collect your data. CCE integrates Prometheus and Metrics Server to realize such capabilities:

- **Prometheus** is an open-source monitoring and alarming framework that can collect multiple types of metrics. Prometheus has been a standard monitoring solution of Kubernetes.
- **Metrics Server** is a cluster-wide aggregator of resource utilization data. Metrics Server collects metrics from the Summary API exposed by kubelet. These metrics are set for core Kubernetes resources, such as pods, nodes, containers, and Services. Metrics Server provides a set of standard APIs for external systems to collect these metrics.

HPA can work with Metrics Server to implement auto scaling based on the CPU and memory usage. It can also work with Prometheus to implement auto scaling based on custom monitoring metrics.

*Figure 15-1* shows how HPA works.
Two core modules of HPA:

- **Data Source Monitoring**
  The community provided only CPU- and memory-based HPA at the early stage. With the population of Kubernetes and Prometheus, developers need more custom metrics or monitoring information at the access layer for their own applications, for example, the QPS of the load balancer and the number of online users of the website. In response, the community defines a set of standard metric APIs to provide services externally through these aggregated APIs.
  - **metrics.k8s.io** provides monitoring metrics related to the CPU and memory of pods and nodes.
  - **custom.metrics.k8s.io** provides custom monitoring metrics related to Kubernetes objects.
  - **external.metrics.k8s.io** provides metrics that come from external systems and are irrelevant to any Kubernetes resource metrics.

- **Scaling Decision-Making Algorithms**
  The HPA controller calculates the scaling ratio based on the current metric values and desired metric values using the following formula:
  \[
  \text{desiredReplicas} = \text{ceil}\left[\frac{\text{currentReplicas} \times \text{currentMetricValue}}{\text{desiredMetricValue}}\right]
  \]
  For example, if the current metric value is 200m and the target value is 100m, the desired number of pods will be doubled according to the formula. In practice, pods may be constantly added or reduced. To ensure stability, the HPA controller is optimized from the following aspects:
  - **Cooldown interval:** In v1.11 and earlier versions, Kubernetes introduced the startup parameters `horizontal-pod-autoscaler-downscale-stabilization-window` and `horizontal-pod-autoScaler-upscale-stabilization-window` to indicate the cooldown intervals after a scale-in and scale-out, respectively, in which no scaling operation will not be performed. In versions later than v1.14, the scheduling queue is introduced to store all decision-making suggestions detected within a period of time. Then, the system makes decisions based on all valid decision-making suggestions to minimize changes of the desired number of replicas to ensure stability.
- Tolerance: It can be considered as a buffer zone. If the pod number changes can be tolerated, the number of pods remains unchanged.
  Use the formula: \( \text{ratio} = \frac{\text{currentMetricValue}}{\text{desiredMetricValue}} \)
  When \(|\text{ratio} - 1.0| \leq \text{tolerance}\), scaling will not be performed.
  When \(|\text{ratio} - 1.0| > \text{tolerance}\), the desired value is calculated using the formula mentioned above.
  The default value is 0.1 in the current community version.

The community settings of the cooldown time window and tolerance are globally applied to all applications, but you may want different settings for different applications, and CCE allows you to do so easily on its console.

The HPA performs scaling based on metric thresholds. Common metrics include the CPU and memory usage. You can also set custom metrics, such as the QPS and number of connections, to trigger scaling. However, metric-based scaling brings in latency of minutes generated during data collection, determination, and scaling phases. Such latency may cause high CPU usage and slow response. To solve this problem, CCE allows you to configure scheduled policies to scale resources regularly for applications with periodic changes.

15.2.2 Creating an HPA Policy for Workload Auto Scaling

Horizontal Pod Autoscaling (HPA) in Kubernetes implements horizontal scaling of pods. In a CCE HPA policy, you can configure different cooldown time windows and scaling thresholds for different applications based on the Kubernetes HPA.

Prerequisites

The metrics-server add-on has been installed. This add-on collects public metrics of kubelet in Kubernetes clusters, including the CPU usage and memory usage. If an HPA scaling policy is created based on the CPU or memory usage, this add-on must be installed.

Notes and Constraints

- HPA policies can be created only for clusters of v1.13 or later.
- The metrics-server and prometheus add-ons on which HPA depends cannot be installed in Kunpeng clusters.
- Only one policy can be created for each workload. That is, if you have created an HPA policy, you cannot create CustomedHPA policies or other HPA policies for the workload. You can delete the created HPA policy and create a new one.

Procedure

Step 1 Log in to the CCE console. In the navigation pane, choose Auto Scaling. On the Workload Scaling tab page, click Create HPA Policy.

Step 2 In the Check Add-ons step:

- If Not Installed is displayed next to the add-on name, click Install, set add-on parameters as required, and click Install to install the add-on.
If is displayed next to the add-on name, the add-on has been installed.

**Step 3** After the required add-ons have been installed, click **Next: Policy configuration**.

**NOTE**

If the add-ons have been installed, after you click **Create HPA Policy**, you will directly land on the second step to configure the policy. The first step (checking the add-ons) has been completed almost instantly.

**Step 4** Set policy parameters by referring to **Table 15-4**.

### Table 15-4 HPA policy parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Policy Name</td>
<td>Name of the policy to be created. Set this parameter as required.</td>
</tr>
<tr>
<td>Cluster Name</td>
<td>Cluster to which the workload belongs.</td>
</tr>
<tr>
<td>Namespace</td>
<td>Namespace to which the workload belongs.</td>
</tr>
<tr>
<td>Associated Workload</td>
<td>Workload with which the HPA policy is associated.</td>
</tr>
<tr>
<td>Pod Range</td>
<td>Minimum and maximum numbers of pods. When a policy is triggered, the workload pods are scaled within this range.</td>
</tr>
</tbody>
</table>
| Cooldown Period         | Interval between a scale-in and a scale-out. The unit is minute. **The interval cannot be shorter than 1 minute.**  
                          | This parameter is available only for clusters of v1.15 and later. **It is not supported in clusters of v1.13 or earlier.**  
<pre><code>                      | This parameter indicates the interval between consecutive scaling operations. The cooldown period ensures that a scaling operation is initiated only when the previous one is completed and the system is running stably. |
</code></pre>
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rules</td>
<td>Policy rules can be based on system metrics.</td>
</tr>
<tr>
<td>System metrics</td>
<td></td>
</tr>
<tr>
<td>Metric</td>
<td>You can select <strong>CPU usage</strong> or <strong>Memory usage</strong>.</td>
</tr>
<tr>
<td></td>
<td><strong>NOTE</strong> Usage = CPUs or memory used by pods/Requested CPUs or memory.</td>
</tr>
<tr>
<td>Expected Value</td>
<td>Enter the expected average resource usage. This parameter indicates the</td>
</tr>
<tr>
<td></td>
<td>expected value of the selected metric.</td>
</tr>
<tr>
<td></td>
<td>The number of new pods required (rounded up) = Current metric value/Expected</td>
</tr>
<tr>
<td></td>
<td>value x Number of current pods.</td>
</tr>
<tr>
<td>Threshold</td>
<td>Enter the scaling thresholds.</td>
</tr>
<tr>
<td></td>
<td>If the metric value is greater than the scale-in threshold and less than</td>
</tr>
<tr>
<td></td>
<td>the scale-out threshold, no scaling is triggered. This parameter is</td>
</tr>
<tr>
<td></td>
<td>supported only in clusters of v1.15 or later.</td>
</tr>
</tbody>
</table>

You can click **Add Rule** again to add more scaling policies.

**NOTE** When calculating the number of pods to be added or reduced, the HPA policy uses the maximum metrics values in the last 5 minutes.

---

**Step 5** After the configuration is complete, click **Create**. If the system displays a message indicating that the request to create workload policy *** is successfully submitted, click **Back to Workload Scaling**.

**Step 6** On the **Workload Scaling** tab page, you can view the newly created HPA policy.

**Figure 15-2 Creating an HPA policy**

<table>
<thead>
<tr>
<th>Workload Scaling</th>
<th>Node Scaling</th>
</tr>
</thead>
<tbody>
<tr>
<td>Related add-on</td>
<td>metrics-view R</td>
</tr>
<tr>
<td>Create HPA Policy</td>
<td>Create CustomedHPA Policy</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Policy Name</th>
<th>Type</th>
<th>Latest Status</th>
<th>Pod Range</th>
<th>Cooldown Period</th>
<th>Rules</th>
<th>Associated Workload</th>
<th>Namespace</th>
</tr>
</thead>
<tbody>
<tr>
<td>HPA Policy</td>
<td></td>
<td></td>
<td>2 - 4</td>
<td>Per scale-down...</td>
<td>1</td>
<td></td>
<td>default</td>
</tr>
</tbody>
</table>

---End

### 15.2.3 Creating a CustomedHPA Policy for Workload Auto Scaling

A CustomedHPA policy scales Deployments based on metrics (such as CPU usage and memory usage) or at a periodic interval (a specific time point every day, every week, every month, or every year). This type of policy is a CCE-enhanced auto scaling capability.

**Supported functions:**

- Scaling can be performed based on the percentage of the current number of pods.
- The minimum scaling step can be set.
- Different scaling operations can be performed based on the actual metric values.

**Prerequisites**

Before configuring a CustomedHPA policy, you must install the `cce-hpa-controller` and `prometheus` add-ons of v1.1.0 or later.
- `cce-hpa-controller` performs scaling based on the percentage of the current number of pods. The minimum step for a scaling operation can be set. Different scaling actions can be performed based on the actual metric values. This add-on must be installed if you want to create CustomedHPA policies.
- `prometheus` collects public metrics (CPU usage and memory usage) of kubelet in a Kubernetes cluster. This add-on must be installed if you want to create CustomedHPA policies.

**Notes and Constraints**

- CustomedHPA policies can be created only for clusters of v1.15 or later.
- CustomedHPA policies cannot be created for Kunpeng clusters.
- Only one policy can be created for each workload. That is, if you have created an HPA policy, you cannot create a CustomedHPA policy or other HPA policies for the workload. You can delete the created HPA policy and create a new one.

**Procedure**

**Step 1** Log in to the CCE console. In the navigation pane, choose Auto Scaling. On the Workload Scaling tab page, click Create CustomedHPA Policy.

**Step 2** In the Check Add-ons step:
- If ✗ Not Installed is displayed next to the add-on name, click Install, set add-on parameters as required, and click Install to install the add-on.
- If ✗ Already installed is displayed next to the add-on name, the add-on has been installed.

**Step 3** After the required add-ons have been installed, click Next: Policy configuration.

**NOTE**

If the add-ons have been installed, after you click Create CustomedHPA Policy, you will directly land on the second step to configure the policy. The first step (checking the add-ons) has been completed almost instantly.

**Step 4** Set policy parameters by referring to Table 15-5.

**Table 15-5** CustomedHPA policy parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Policy Name</td>
<td>Name of the policy to be created. Set this parameter as required.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Cluster Name</td>
<td>Cluster to which the workload belongs.</td>
</tr>
<tr>
<td>Namespace</td>
<td>Namespace to which the workload belongs.</td>
</tr>
<tr>
<td>Associated Workload</td>
<td>Workload with which the CustomedHPA policy is associated.</td>
</tr>
<tr>
<td>Pod Range</td>
<td>Minimum and maximum numbers of pods. When a policy is triggered, the workload pods are scaled within this range.</td>
</tr>
<tr>
<td>Cooldown Period</td>
<td>Enter an interval, in minutes. This parameter indicates the interval between consecutive scaling operations. The cooldown period ensures that a scaling operation is initiated only when the previous one is completed and the system is running stably.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
| Rules     | Click **Add Rule**. In the dialog box displayed, set the following parameters:  
  - **Name**: Enter a custom rule name.  
  - **Type**: You can select **Metric-based** or **Periodic**.  
  **Metric-based**:  
  - **Condition**: Select **CPU usage** or **Memory usage**, choose > or <, and enter a percentage. As shown in the following figure, the rule will be executed immediately when the CPU usage is greater than 50%.  
    
    **Figure 15-3 Setting a trigger condition**

```
Add Rule

<table>
<thead>
<tr>
<th>Rule Name</th>
<th>Enter a rule name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Metric-based</td>
</tr>
<tr>
<td>Condition</td>
<td>CPU usage &gt; 50 %</td>
</tr>
</tbody>
</table>
```

- **Action**: Set an action to be performed when the trigger condition is met. Multiple actions can be added. As shown below, when the CPU usage exceeds 50%, the number of pods is scaled out to 5. When the CPU usage exceeds 70%, the number of pods is scaled out to 8. When the CPU usage exceeds 90%, the number of pods is scaled out to 18 (adding 10 more pods). These rules also work for scale-in operations.  
  
  **Figure 15-4 Action**

```
Add Rule

<table>
<thead>
<tr>
<th>Rule Name</th>
<th>Enter a rule name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Metric-based</td>
</tr>
<tr>
<td>Condition</td>
<td>CPU usage &gt; 50 %</td>
</tr>
</tbody>
</table>

Action

<table>
<thead>
<tr>
<th>Scope</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>50 %</td>
<td>Scale to 5 pods</td>
</tr>
<tr>
<td>70 %</td>
<td>Scale to 8 pods</td>
</tr>
<tr>
<td>90 %</td>
<td>Increase 10 pods</td>
</tr>
</tbody>
</table>

Enable: Enable or disable the policy rule.  
Periodic:
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>● <strong>Triggered At</strong>: You can select a specific time point every day, every week, every month, or every year. As shown below, the rule is triggered at 17:00 every day.</td>
<td></td>
</tr>
</tbody>
</table>

**Figure 15-5** Periodic triggering (Daily)

<table>
<thead>
<tr>
<th>Add Rule</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rule Name</td>
</tr>
<tr>
<td>Type</td>
</tr>
<tr>
<td>Triggered At</td>
</tr>
<tr>
<td>Action</td>
</tr>
<tr>
<td>Enable</td>
</tr>
</tbody>
</table>

| ● **Action**: Set an action to be performed when the **Triggered At** value is reached. As shown below, four pods will be reduced at 17:00 every day. |

**Figure 15-6** Setting an action for periodic triggering

<table>
<thead>
<tr>
<th>Add Rule</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rule Name</td>
</tr>
<tr>
<td>Type</td>
</tr>
<tr>
<td>Triggered At</td>
</tr>
<tr>
<td>Action</td>
</tr>
<tr>
<td>Enable</td>
</tr>
</tbody>
</table>

| ● **Enable**: Enable or disable the policy rule. |

Click **OK**. You can view the added rule in the policy rule list and enable, disable, edit, or delete the rule.

You can click **Add Rule** below the policy rule list to add multiple rules.

---

**Step 5** After the configuration is complete, click **Create**. If the system displays a message indicating that the request to create workload policy *** is successfully submitted, click **Back to Workload Scaling**.

**Step 6** On the **Workload Scaling** tab page, you can view the newly created CustomedHPA policy.
15.2.4 Managing Workload Scaling Policies

Scenario

After an HPA or CustomedHPA policy is created, you can update, clone, edit, and delete the policy, as well as edit the YAML file.

Checking an HPA Policy

You can view the rules, status, and events of an HPA policy and handle exceptions based on the error information displayed.

Step 1  Log in to the CCE console. In the navigation pane, choose Auto Scaling. On the Workload Scaling tab page, click in front of the target policy.

Step 2  In the expanded area, you can view the Rules, Status, and Events tab pages. If the policy is abnormal, locate and rectify the fault based on the error information.

NOTE

You can also view the created HPA policy on the workload details page. Log in to the CCE console, choose Workloads > Deployments or Workloads > StatefulSets in the navigation pane, and choose More > Scaling in the Operation column. On the workload details page, click the Scaling tab. You can see the Auto Scaling-HPA / CustomedHPA pane, as well as the HPA policy you have configured on the Auto Scaling page.

Table 15-6  Event types and names

<table>
<thead>
<tr>
<th>Event Type</th>
<th>Event Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal</td>
<td>SuccessfulRescale</td>
<td>The scaling is performed successfully.</td>
</tr>
<tr>
<td>Abnormal</td>
<td>InvalidTargetRange</td>
<td>Invalid target range.</td>
</tr>
<tr>
<td></td>
<td>InvalidSelector</td>
<td>Invalid selector.</td>
</tr>
<tr>
<td></td>
<td>FailedGetObjectMetric</td>
<td>Objects fail to be obtained.</td>
</tr>
<tr>
<td></td>
<td>FailedGetPodsMetric</td>
<td>Pods fail to be obtained.</td>
</tr>
<tr>
<td></td>
<td>FailedGetResourceMetric</td>
<td>Resources fail to be obtained.</td>
</tr>
</tbody>
</table>
### Event List

<table>
<thead>
<tr>
<th>Event Type</th>
<th>Event Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>FailedGetExternalMetric</td>
<td>External metrics fail to be obtained.</td>
</tr>
<tr>
<td></td>
<td>InvalidMetricSourceType</td>
<td>Invalid metric source type.</td>
</tr>
<tr>
<td></td>
<td>FailedConvertHPA</td>
<td>HPA conversion failed.</td>
</tr>
<tr>
<td></td>
<td>FailedGetScale</td>
<td>The scale fails to be obtained.</td>
</tr>
<tr>
<td></td>
<td>FailedComputeMetricsReplicas</td>
<td>Failed to calculate metric-defined replicas.</td>
</tr>
<tr>
<td></td>
<td>FailedGetScaleWindow</td>
<td>Failed to obtain ScaleWindow.</td>
</tr>
<tr>
<td></td>
<td>FailedRescale</td>
<td>Failed to scale the service.</td>
</tr>
</tbody>
</table>

---

**Viewing a CustomedHPA Policy**

You can view the rules and latest status of a CustomedHPA policy and rectify faults based on the error information displayed.

**Step 1** Log in to the CCE console. In the navigation pane, choose **Auto Scaling**. On the **Workload Scaling** tab page, click ![ ] in front of the target policy.

**Step 2** In the expanded area, if the policy is abnormal on the **Rules** tab page, click **Details** in **Latest Status** and locate the fault based on the information displayed.

**NOTE**

You can also view the created CustomedHPA policy on the workload details page. Log in to the CCE console, choose **Workloads > Deployments** or **Workloads > StatefulSets** in the navigation pane, and choose **More > Scaling** in the **Operation** column. On the workload details page, click the **Scaling** tab. You can see the **Auto Scaling-HPA / CustomedHPA** pane, as well as the CustomedHPA policy you have configured on the **Auto Scaling** page.

---

**Updating an HPA or CustomedHPA Policy**

An HPA policy is used as an example.

**Step 1** Log in to the CCE console. In the navigation pane, choose **Auto Scaling**. On the **Workload Scaling** tab page, click **Update** in the **Operation** column of the policy to be updated.

**Step 2** On the **Update HPA Policy** page displayed, set the policy parameters listed in **Table 15-4**.

**Step 3** Click **Update**.

---
Cloning an HPA or CustomedHPA Policy

**Step 1** Log in to the CCE console. In the navigation pane, choose Auto Scaling. On the Workload Scaling tab page, click Clone in the Operation column of the target policy.

**Step 2** For example, for an HPA policy, on the Create HPA Policy page, you can view that parameters such as Pod Range, Cooldown Period, and Rules have been cloned. Add or modify other policy parameters as needed.

**Step 3** Click Create to complete policy cloning. On the Workload Scaling tab page, you can view the cloned policy in the policy list.

-----End

Editing the YAML File (HPA Policy)

**Step 1** Log in to the CCE console. In the navigation pane, choose Auto Scaling. On the Workload Scaling tab page, choose More > Edit YAML in the Operation column of the target policy.

**Step 2** In the Edit YAML dialog box displayed, edit or download the YAML file.

**Step 3** Click the close button in the upper right corner.

-----End

Viewing the YAML File (CustomedHPA Policy)

**Step 1** Log in to the CCE console. In the navigation pane, choose Auto Scaling. On the Workload Scaling tab page, choose More > View YAML in the Operation column of the target policy.

**Step 2** In the dialog box displayed, you can copy and download the YAML file but cannot modify it.

**Step 3** Click the close button in the upper right corner.

-----End

Deleting an HPA or CustomedHPA Policy

**Step 1** Log in to the CCE console. In the navigation pane, choose Auto Scaling. On the Workload Scaling tab page, choose More > Delete in the Operation column of the target policy.

**Step 2** In the Delete HPA Policy dialog box displayed, confirm whether to delete the HPA policy.

**Step 3** Click Yes to delete the policy.

-----End

15.3 Scaling a Cluster/Node
15.3.1 Node Scaling Mechanisms

Kubernetes HPA is designed for pods. However, if the cluster resources are insufficient, you can only add nodes. Scaling of cluster nodes could be laborious. Now with clouds, you can add or delete nodes by simply calling APIs.

**autoscaler** is a component provided by Kubernetes for auto scaling of cluster nodes based on the pod scheduling status and resource usage.

**Prerequisites**

Before using the node scaling function, you must install the **autoscaler** add-on of v1.13.8 or later.

**How autoscaler Works**

The cluster autoscaler (CA) goes through two processes.

- **Scale-out:** The CA checks all unschedulable pods every 15 seconds and selects a node group that meets the requirements for scale-out based on the policy you set.
- **Scale-in:** The CA scans all nodes every 10 seconds. If the number of pod requests on a node is less than the user-defined percentage for scale-in, the CA simulates whether the pods on the node can be migrated to other nodes. If yes, the node will be removed after an idle time window.

As described above, if a cluster node is idle for a period of time (10 minutes by default), scale-in is triggered, and the idle node is removed.

However, a node cannot be removed from a cluster if the following pods exist:

1. Pods that do not meet specific requirements set in PodDisruptionBudget
2. Pods that cannot be scheduled to other nodes due to constraints such as affinity and anti-affinity policies
3. Pods that have the "**cluster-autoscaler.kubernetes.io/safe-to-evict**: "false" annotation
4. Pods (except those created by kube-system DaemonSet) that exist in the kube-system namespace on the node
5. Pods that are not created by the controller (Deployment/ReplicaSet/job/StatefulSet)

**autoscaler Architecture**

*Figure 15-8* shows the autoscaler architecture and its core modules:
Figure 15-8 autoscaler architecture

Description

- **Estimator**: Evaluates the number of nodes to be added to each node pool to host unschedulable pods.
- **Simulator**: Finds the nodes that meet the scale-in conditions in the scale-in scenario.
- **Expander**: Selects an optimal node from the node pool picked out by the Estimator based on the user-defined policy in the scale-out scenario. Currently, the Expander has the following policies:
  - **Random**: Selects a node pool randomly. If you have not specified a policy, Random is set by default.
  - **most-Pods**: Selects the node pool that can host the largest number of unschedulable pods after the scale-out. If multiple node pools meet the requirement, a random node pool will be selected.
  - **least-waste**: Selects the node pool that has the least CPU or memory resource waste after scale-out.
  - **price**: Selects the node pool in which the to-be-added nodes cost least for scale-out.
  - **priority**: Selects the node pool with the highest weight. The weights are user-defined.

Currently, CCE supports all policies except price. By default, CCE add-ons use the least-waste policy.
15.3.2 Creating a Node Scaling Policy

CCE provides auto scaling through the autoscaler add-on. Nodes with different specifications can be automatically added across AZs on demand.

If a node scaling policy and the configuration in the autoscaler add-on take effect at the same time, for example, there are pods that cannot be scheduled and the value of a metric reaches the threshold at the same time, scale-out is performed first for the unschedulable pods.

- If the scale-out succeeds for the unschedulable pods, the system skips the metric-based rule logic and enters the next loop.
- If the scale-out fails for the unschedulable pods, the metric-based rule is executed.

Prerequisites

Before using the node scaling function, you must install the autoscaler add-on of v1.13.8 or later.

Procedure

Step 1  Log in to the CCE console. In the navigation pane, choose Auto Scaling. On the Node Scaling tab page, click Create Node Scaling Policy.

Step 2  In the Check Add-ons step:

- If Not installed is displayed next to the add-on name, click Install, set add-on parameters as required, and click Install to install the add-on.

- If Already installed is displayed next to the add-on name, the add-on has been installed.

Step 3  After the required add-ons have been installed, click Next: Policy configuration.

**NOTE**

If the add-ons have been installed, after you click Create Node Scaling Policy, you will directly land on the second step to configure the policy. The first step (checking the add-ons) has been completed almost instantly.

Step 4  On the Create Node Scaling Policy page, set the following policy parameters.

- Policy Name: name of the policy to be created, which can be customized.
- Associated Node Pool: Click Add Node Pool and select the node pool to be associated. You can associate multiple node pools to use the same scaling policy.
NOTE

Priority is now supported for node pools. CCE will select a node pool for auto scaling based on the following policies:

1. CCE uses algorithms to determine whether a node pool meets the conditions to allow scheduling of a pod in pending state, including whether the node resources are greater than requested by the pod, and whether the nodeSelect, nodeAffinity, and taints meet the conditions. In addition, the node pools that fail to be scaled (due to insufficient resources or other reasons) and are still in the 15-minute cool-down interval are filtered.

2. If multiple node pools meet the scaling requirements, the system checks the priority of each node pool and selects the node pool with the highest priority for scaling. The value ranges from 0 to 100 and the default priority is 0. The value 100 indicates the highest priority, and the value 0 indicates the lowest priority.

3. If multiple node pools have the same priority or no priority is configured for them, the system selects the node pool that will consume the least resources based on the configured VM specification.

4. If the VM specifications of multiple node pools are the same but the node pools are deployed in different AZs, the system randomly selects a node pool to trigger scaling.

5. If the resources of the preferred node pool are insufficient, the system automatically selects next node pool based on the priority.

For details about the node pool priority, see Autoscaler.

• Execution Rules: Click Add Rule. In the dialog box displayed, set the following parameters:

  Name: Enter a rule name.

  Type: You can select Metric-based or Periodic. The differences between the two types are as follows:

  - Metric-based:

    — Condition: Select CPU allocation or Memory allocation and enter a value. The value must be greater than the scale-in percentage configured in the autoscaler add-on.

    NOTE

    ○ Resource allocation (%) = Resources requested by pods in the node pool/Resources allocatable to pods in the node pool

    ○ If multiple rules meet the conditions, the rules are executed in either of the following modes:

      If rules based on the CPU allocation rate and memory allocation rate are configured and two or more rules meet the scale-out conditions, the rule that will add the most nodes will be executed.

      If a rule based on the CPU allocation rate and a periodic rule are configured and they both meet the scale-out conditions, one of them will be executed randomly. The rule executed first (rule A) changes the node pool to the scaling state. As a result, the other rule (rule B) cannot be executed. After rule A is executed and the node pool status becomes normal, rule B will not be executed.

      ○ If rules based on the CPU allocation rate and memory allocation rate are configured, the policy detection period varies with the processing logic of each loop of the autoscaler add-on. Scale-out is triggered once the conditions are met, but it is constrained by other factors such as the cool-down interval and node pool status.
- **Action**: Set an action to be performed when the trigger condition is met. As shown in Figure 15-9, five nodes will be added when the memory allocation rate exceeds 40%.

**Figure 15-9 Setting an action for metric-based triggering**

- **Periodic**:
  - **Triggered At**: You can select a specific time point every day, every week, every month, or every year. As shown in Figure 15-10, the rule is triggered at 15:00 every day.

**Figure 15-10 Setting a trigger time**

- **Action**: Set an action to be performed when the **Triggered At** value is reached. As shown in Figure 15-11, five nodes will be added at 15:00 every day.
Figure 15-11 Setting an action for periodic triggering

You can click Add Rule again to add more node scaling policies. You can add a maximum of one CPU usage-based rule and one memory usage-based rule. The total number of rules cannot exceed 10.

Step 5 After the configuration is complete, click Create. If the system displays a message indicating that the request to create a node scaling policy is submitted successfully, click Back to Node Scaling Policy List.

Step 6 On the Node Scaling tab page, you can view the created node scaling policy.

Figure 15-12 Node scaling policy

Constraints on Scale-in

CCE cannot trigger scale-in by using node scaling policies. You can set a scale-in policy when installing the autoscaler add-on.

Node scale-in can be triggered only by the resource allocation rate. When CPU and memory allocation rates in a cluster are lower than the specified thresholds (set when the autoscaler add-on is installed or modified), scale-in is triggered for nodes in the node pool (this function can be disabled), as shown in Figure 15-13.
Example YAML File

The following is a YAML example of a node scaling policy:

```yaml
apiVersion: autoscaling.cce.io/v1alpha1
group: HorizontalNodeAutoscaler
metadata:
  creationTimestamp: "2020-02-13T12:47:49Z"
generation: 1
  name: xxxx
  namespace: kube-system
  resourceVersion: "11433270"
  selfLink: /apis/autoscaling.cce.io/v1alpha1/namespaces/kube-system/horizontalnodeautoscalers/xxxx
  uid: c2bd1e1d-60aa-47b5-938c-6bf3fadbe91f
spec:
  disable: false
  rules:
  - action:
      type: ScaleUp
      unit: Node
      value: 1
cronTrigger:
    schedule: 47 20 * * *
    disable: false
    ruleName: cronrule
type: Cron
  - action:
      type: ScaleUp
      unit: Node
      value: 2
disabled: false
metricTrigger:
  metricName: Cpu
  metricOperation: '>'
  metricValue: "40"
  unit: Percent
```

Figure 15-13 Auto scale-in configuration
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>spec.disable</td>
<td>Bool</td>
<td>Whether to enable the scaling policy. This parameter takes effect for all rules in the policy.</td>
</tr>
<tr>
<td>spec.rules</td>
<td>Array</td>
<td>All rules in a scaling policy.</td>
</tr>
<tr>
<td>spec.rules[x].ruleName</td>
<td>String</td>
<td>Rule name.</td>
</tr>
<tr>
<td>spec.rules[x].type</td>
<td>String</td>
<td>Rule type. Currently, <strong>Cron</strong> and <strong>Metric</strong> are supported.</td>
</tr>
<tr>
<td>spec.rules[x].disable</td>
<td>Bool</td>
<td>Rule switch. Currently, only <strong>false</strong> is supported.</td>
</tr>
<tr>
<td>spec.rules[x].action.type</td>
<td>String</td>
<td>Rule action type. Currently, only <strong>ScaleUp</strong> is supported.</td>
</tr>
<tr>
<td>spec.rules[x].action.unit</td>
<td>String</td>
<td>Rule action unit. Currently, only <strong>Node</strong> is supported.</td>
</tr>
<tr>
<td>spec.rules[x].action.value</td>
<td>Integer</td>
<td>Rule action value.</td>
</tr>
<tr>
<td>spec.rules[x].cronTrigger</td>
<td>/</td>
<td>Optional. This parameter is valid only in periodic rules.</td>
</tr>
<tr>
<td>spec.rules[x].cronTrigger.sched ule</td>
<td>String</td>
<td>Cron expression of a periodic rule.</td>
</tr>
<tr>
<td>spec.rules[x].metricTrigger</td>
<td>/</td>
<td>Optional. This parameter is valid only in metric-based rules.</td>
</tr>
<tr>
<td>spec.rules[x].metricTrigger.metricName</td>
<td>String</td>
<td>Metric of a metric-based rule. Currently, <strong>Cpu</strong> and <strong>Memory</strong> are supported.</td>
</tr>
<tr>
<td>spec.rules[x].metricTrigger.metricOperation</td>
<td>String</td>
<td>Comparison operator of a metric-based rule. Currently, only &gt; is supported.</td>
</tr>
<tr>
<td>spec.rules[x].metricTrigger.metricValue</td>
<td>String</td>
<td>Metric threshold of a metric-based rule. The value can be any integer from 1 to 100 and must be a character string.</td>
</tr>
<tr>
<td>spec.rules[x].metricTrigger.Unit</td>
<td>String</td>
<td>Unit of the metric-based rule threshold. Currently, only % is supported.</td>
</tr>
</tbody>
</table>
### 15.3.3 Managing Node Scaling Policies

**Scenario**

After a node scaling policy is created, you can delete, edit, disable, enable, or clone the policy.

**Viewing a Node Scaling Policy**

You can view the associated node pool, rules, and scaling history of a node scaling policy and rectify faults according to the error information displayed.

**Step 1** Log in to the CCE console. In the navigation pane, choose **Auto Scaling**. On the **Node Scaling** tab page, click **✓** in front of the policy to be viewed.

**Step 2** In the expanded area, the **Associated Node Pool**, **Execution Rules**, and **Scaling Records** tab pages are displayed. If the policy is abnormal, locate and rectify the fault based on the error information.

- **NOTE**
  
  You can also enable or disable auto scaling in **Node Pools**. Log in to the CCE console. In the navigation pane, choose **Resource Management > Node Pools**, and click **Edit** in the upper right corner of the node pool to be operated. In the **Edit Node Pool** dialog box displayed, you can enable **Autoscaler** and set the limits of the number of nodes and the cooling interval for auto scaling.

----End

**Deleting a Node Scaling Policy**

**Step 1** Log in to the CCE console. In the navigation pane, choose **Auto Scaling**. On the **Node Scaling** tab page, click **Delete** in the **Operation** column of the policy to be deleted.

**Step 2** In the **Delete Node Policy** dialog box displayed, confirm whether to delete the policy.

**Step 3** Enter **DELETE** in the text box.

**Step 4** Click **OK** to delete the policy.

----End
Editing a Node Scaling Policy

Step 1  Log in to the CCE console. In the navigation pane, choose Auto Scaling. On the Node Scaling tab page, click Edit in the Operation column of the policy.

Step 2  On the Edit Node Scaling Policy page displayed, modify policy parameter values listed in Table 15-7.

Step 3  After the configuration is complete, click OK.

----End

Cloning a Node Scaling Policy

Step 1  Log in to the CCE console. In the navigation pane, choose Auto Scaling. On the Node Scaling tab page, click More > Clone in the Operation column of the policy.

Step 2  On the Create Node Scaling Policy page displayed, certain parameters have been cloned. Add or modify other policy parameters based on service requirements.

Step 3  Click Create Now to clone the policy. The cloned policy is displayed in the policy list on the Node Scaling tab page.

----End

Enabling or Disabling a Node Scaling Policy

Step 1  Log in to the CCE console. In the navigation pane, choose Auto Scaling. On the Node Scaling tab page, click More > Disable or Enable in the Operation column of the policy.

Step 2  In the dialog box displayed, confirm whether to disable or enable the node policy.

Step 3  Click Yes. The policy status is displayed in the node scaling list.

----End

15.4 Using HPA and CA for Auto Scaling of Workloads and Nodes

Issues

The best way to handle surging traffic is to automatically adjust the number of machines based on the traffic volume or resource usage, which is called scaling.

In CCE, the resources that can be used by containers are fixed during application deployment. Therefore, in auto scaling, pods are scaled first. The node resource usage increases only after the number of pods increases. Then, nodes can be scaled based on the node resource usage. How to configure auto scaling in CCE?
Solution

Two major auto scaling policies are HPA (Horizontal Pod Autoscaling) and CA (Cluster AutoScaling). HPA is for workload auto scaling and CA is for node auto scaling.

HPA and CA work with each other. HPA requires sufficient cluster resources for successful scaling. When the cluster resources are insufficient, CA is needed to add nodes. If HPA reduces workloads, the cluster will have a large number of idle resources. In this case, CA needs to release nodes to avoid resource waste.

As shown in Figure 15-14, HPA performs scale-out based on the monitoring metrics. When cluster resources are insufficient, newly created pods are in Pending state. CA then checks these pending pods and selects the most appropriate node pool based on the configured scaling policy to scale out the node pool. For details about how HPA and CA work, see Workload Scaling Mechanisms and Node Scaling Mechanisms.

Figure 15-14 HPA and CA working flows

CCE allows you to create a CA policy to scale out nodes periodically or based on the CPU or memory allocation rate. The CA policy can work with the autoscaler add-on to scales out nodes based on the pending status of pods.

Using HPA and CA can easily implement auto scaling in most scenarios. In addition, the scaling process of nodes and pods can be easily observed.

This section uses an example to describe the auto scaling process using HPA and CA policies together.

Preparations
- Prepare a compute-intensive application. When a user sends a request, the result is returned to the user only after computing. The following is a PHP page. The code is written in the index.php file. When a user sends a request, the root extraction is performed for 1,000,000 times, and then "OK!" is returned.

```php
<?php
$x = 0.0001;
for ($i = 0; $i <= 1000000; $i++) {
    $x += sqrt($x);
}
```
```bash
    echo "OK!";
?>
```

Compile a Dockerfile to create an image.

```
FROM php:5-apache
RUN chmod a+rx index.php
```

Run the following command to build an image named `hpa-example`.

```
docker build -t hpa-example:latest .
```

After the build is complete, upload the image to SWR. For details, see `Uploading an Image Through a Container Engine Client`.

- Create a cluster with one node with 2 cores of CPU and 4 GB of memory. Assign an EIP for the node to allow external access.
- Install add-ons for the cluster.
  - `cce-hpa-controller`: HPA add-on
  - `autoscaler`: CA add-on
  - `prometheus`: an open-source system monitoring and alarm framework that can collect multiple metrics
  - `metrics-server`: an aggregator of resource usage data in a Kubernetes cluster. It can collect measurement data of major Kubernetes resources, such as pods, nodes, containers, and Services.

---

**Creating a Node Pool and a CA Policy**

Create a node pool, add a node with 2 vCPUs and 4 GB memory, and enable auto scaling.

Modify the autoscaler add-on configuration, enable auto scale-in, and configure scale-in parameters. For example, when the node resource usage is less than 50%, the scale-in is triggered.
After the preceding configurations, scale-out is performed based on the pending status of the pod and scale-in is triggered when the node resource usage decreases.

When you create a CA policy on CCE, scale-out can be triggered periodically or based on the CPU or memory allocation rate. The CA policy can work with the autoscaler add-on for scale-out based on the pending status of pods. As shown in the following figure, when the cluster CPU allocation rate is greater than 70%, one node will be added. A CA policy needs to be associated with a node pool. Multiple node pools can be associated. When you need to scale nodes, node with proper specifications will be added or reduced from the node pool based on the minimum waste principle.

Creating a Workload

Use the hpa-example image to create a Deployment with one replica. The image path is related to the organization uploaded to the SWR repository and needs to be replaced with the actual value.

```yaml
kind: Deployment
apiVersion: apps/v1
metadata:
  name: hpa-example
spec:
  replicas: 1
  selector:
```
matchLabels:
  app: hpa-example

template:
  metadata:
    labels:
      app: hpa-example

spec:
  containers:
    - name: container-1
      image: 'swr.cn-east-3.myhuaweicloud.com/group/hpa-example:latest'
      resources:
        limits:
          cpu: 500m
          memory: 200Mi
        requests:
          cpu: 250m
          memory: 200Mi
      imagePullSecrets:
        - name: default-secret

Then, create a NodePort Service for the workload so that the workload can be accessed from external networks.

⚠️ NOTE

To allow external access to NodePort Services, you need to purchase an EIP for the node in the cluster. After purchasing the EIP, you need to synchronize node data. For details, see Synchronizing Node Data. If the node already has an EIP, you do not need to buy one.

Alternatively, you can create a Service with an ELB load balancer for external access. For details, see Using kubectl to Create a Service (Automatically Creating a Shared Load Balancer).

```
kind: Service
apiVersion: v1
metadata:
  name: hpa-example
spec:
  ports:
    - name: cce-service-0
      protocol: TCP
      port: 80
      targetPort: 80
      nodePort: 31144
  selector:
    app: hpa-example
  type: NodePort
```

Creating an HPA Policy

Create an HPA policy. As shown below, the policy is associated with the hpa-example workload, and the target CPU usage is 50%.

There are two other annotations. One annotation defines the CPU thresholds, indicating that scaling is not performed when the CPU usage is between 30% and 70% to prevent impact caused by slight fluctuation. The other is the scaling time window, indicating that after the policy is successfully executed, a scaling operation will not be triggered again in this cooling interval to prevent impact caused by short-term fluctuation.

```
apiVersion: autoscaling/v2beta1
kind: HorizontalPodAutoscaler
metadata:
  name: hpa-policy
annotations:
  extendedhpa.metrics: 
    - type: "Resource"
      name: "cpu"
      targetType: "Utilization"
      targetRange: 
```

```
Set the parameters as follows if you are using the console.

### Observing the Auto Scaling Process

**Step 1** Check the cluster node status. In the following example, there are two nodes.

```bash
# kubectl get node
```

<table>
<thead>
<tr>
<th>NAME</th>
<th>STATUS</th>
<th>ROLES</th>
<th>AGE</th>
<th>VERSION</th>
</tr>
</thead>
<tbody>
<tr>
<td>192.168.0.183</td>
<td>Ready</td>
<td>&lt;none&gt;</td>
<td>2m20s</td>
<td>v1.17.9-r0-CCE21.1.1.3.8001-17.36.8</td>
</tr>
<tr>
<td>192.168.0.26</td>
<td>Ready</td>
<td>&lt;none&gt;</td>
<td>55m</td>
<td>v1.17.9-r0-CCE21.1.1.3.8001-17.36.8</td>
</tr>
</tbody>
</table>

Check the HPA policy. The CPU usage of the target workload is 0%.

```bash
# kubectl get hpa hpa-policy
```

<table>
<thead>
<tr>
<th>NAME</th>
<th>REFERENCE</th>
<th>TARGETS</th>
<th>MINPODS</th>
<th>MAXPODS</th>
<th>REPLICA</th>
<th>AGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>hpa-policy</td>
<td>Deployment/hpa-example</td>
<td>0%/50%</td>
<td>1</td>
<td>100</td>
<td>1</td>
<td>4m</td>
</tr>
</tbody>
</table>

**Step 2** Run the following command to access the workload. In the following command, `{ip:port}` indicates the access address of the workload, which can be queried on the workload details page.

```bash
while true;do wget -q -O- http://{ip:port}; done
```

---

**NOTE**

If no EIP is displayed, the cluster node has not been assigned any EIP. You need to buy one, bind it to the node, and synchronize node data. For details, see [Synchronizing Node Data](#).

Observe the scaling process of the workload.
You can see that the CPU usage of the workload is 190% at 4m23s, which exceeds the target value. In this case, scaling is triggered to expand the workload to four replicas/pods. In the subsequent several minutes, the CPU usage does not decrease until 7m16s. This is because the new pods may not be successfully created. The possible cause is that resources are insufficient and the pods are in Pending state. During this period, nodes are added.

At 7m16s, the CPU usage decreases, indicating that the pods are successfully created and start to bear traffic. The CPU usage decreases to 81% at 8m, still greater than the target value (50%) and the high threshold (70%). Therefore, 7 pods are added at 9m16s, and the CPU usage decreases to 51%, which is within the range of 30% to 70%. From then on, the number of pods remains 7.

In the following output, you can see the workload scaling process and the time when the HPA policy takes effect.

Check the number of nodes. The following output shows that two nodes are added.

You can also view the scaling history on the console. For example, the CA policy is executed once when the CPU allocation rate in the cluster is greater than 70%, and the number of nodes in the node pool is increased from 2 to 3. The new node is automatically added by autoscaler based on the pending state of pods in the initial phase of HPA.
The node scaling process is as follows:

1. After the number of pods changes to 4, the pods are in Pending state due to insufficient resources. As a result, the default scale-out policy of the autoscaler add-on is triggered, and the number of nodes is increased by one.

2. The second node scale-out is triggered because the CPU allocation rate in the cluster is greater than 70%. As a result, the number of nodes is increased by one, which is recorded in the scaling history on the console. Scaling based on the allocation rate ensures that the cluster has sufficient resources.

**Step 3** Stop accessing the workload and check the number of pods.

```
# kubectl get hpa hpa-policy --watch
```

<table>
<thead>
<tr>
<th>NAME</th>
<th>REFERENCE</th>
<th>TARGETS</th>
<th>MINPODS</th>
<th>MAXPODS</th>
<th>REPLICAS</th>
<th>AGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>hpa-policy</td>
<td>Deployment/hpa-example</td>
<td>50%/50%</td>
<td>1</td>
<td>100</td>
<td>7</td>
<td>12m</td>
</tr>
<tr>
<td>hpa-policy</td>
<td>Deployment/hpa-example</td>
<td>21%/50%</td>
<td>1</td>
<td>100</td>
<td>13m</td>
<td></td>
</tr>
<tr>
<td>hpa-policy</td>
<td>Deployment/hpa-example</td>
<td>0%/50%</td>
<td>1</td>
<td>100</td>
<td>7</td>
<td>14m</td>
</tr>
<tr>
<td>hpa-policy</td>
<td>Deployment/hpa-example</td>
<td>0%/50%</td>
<td>1</td>
<td>100</td>
<td>18m</td>
<td></td>
</tr>
<tr>
<td>hpa-policy</td>
<td>Deployment/hpa-example</td>
<td>0%/50%</td>
<td>1</td>
<td>100</td>
<td>3</td>
<td>19m</td>
</tr>
<tr>
<td>hpa-policy</td>
<td>Deployment/hpa-example</td>
<td>0%/50%</td>
<td>1</td>
<td>100</td>
<td>3</td>
<td>19m</td>
</tr>
<tr>
<td>hpa-policy</td>
<td>Deployment/hpa-example</td>
<td>0%/50%</td>
<td>1</td>
<td>100</td>
<td>3</td>
<td>19m</td>
</tr>
<tr>
<td>hpa-policy</td>
<td>Deployment/hpa-example</td>
<td>0%/50%</td>
<td>1</td>
<td>100</td>
<td>1</td>
<td>23m</td>
</tr>
<tr>
<td>hpa-policy</td>
<td>Deployment/hpa-example</td>
<td>0%/50%</td>
<td>1</td>
<td>100</td>
<td>1</td>
<td>23m</td>
</tr>
</tbody>
</table>

You can see that the CPU usage is 21% at 13m. The number of pods is reduced to 3 at 18m, and then reduced to 1 at 23m.

In the following output, you can see the workload scaling process and the time when the HPA policy takes effect.

```
# kubectl describe deploy hpa-example
```

<table>
<thead>
<tr>
<th>Type</th>
<th>Reason</th>
<th>Age</th>
<th>From</th>
<th>Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal</td>
<td>ScalingReplicaSet</td>
<td>25m</td>
<td>deployment-controller</td>
<td>Scaled up replica set hpa-example-79dd795485 to 1</td>
</tr>
<tr>
<td>Normal</td>
<td>ScalingReplicaSet</td>
<td>20m</td>
<td>deployment-controller</td>
<td>Scaled up replica set hpa-example-79dd795485 to 4</td>
</tr>
<tr>
<td>Normal</td>
<td>ScalingReplicaSet</td>
<td>16m</td>
<td>deployment-controller</td>
<td>Scaled up replica set hpa-example-79dd795485 to 7</td>
</tr>
<tr>
<td>Normal</td>
<td>ScalingReplicaSet</td>
<td>6m28s</td>
<td>deployment-controller</td>
<td>Scaled down replica set hpa-example-79dd795485 from 7 to 3</td>
</tr>
<tr>
<td>Normal</td>
<td>ScalingReplicaSet</td>
<td>72s</td>
<td>deployment-controller</td>
<td>Scaled down replica set hpa-example-79dd795485 to 1</td>
</tr>
</tbody>
</table>

```
# kubectl describe hpa hpa-policy
```

<table>
<thead>
<tr>
<th>Type</th>
<th>Reason</th>
<th>Age</th>
<th>From</th>
<th>Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal</td>
<td>SuccessfulRescale</td>
<td>20m</td>
<td>horizontal-pod-autoscaler</td>
<td>New size: 4; reason: cpu resource utilization (percentage of request) above target</td>
</tr>
<tr>
<td>Normal</td>
<td>SuccessfulRescale</td>
<td>16m</td>
<td>horizontal-pod-autoscaler</td>
<td>New size: 7; reason: cpu resource utilization (percentage of request) above target</td>
</tr>
<tr>
<td>Normal</td>
<td>SuccessfulRescale</td>
<td>6m45s</td>
<td>horizontal-pod-autoscaler</td>
<td>New size: 3; reason: All metrics below target</td>
</tr>
<tr>
<td>Normal</td>
<td>SuccessfulRescale</td>
<td>90s</td>
<td>horizontal-pod-autoscaler</td>
<td>New size: 1; reason: All metrics below target</td>
</tr>
</tbody>
</table>

You can also view the HPA policy execution history on the console.
Wait until the one node is reduced.

The reason why the other two nodes in the node pool are not reduced is that they both have pods in the kube-system namespace (not pods created by DaemonSets). For details about node scale-in, see How autoscaler Works.

----End

Summary

Using HPA and CA can easily implement auto scaling in most scenarios. In addition, the scaling process of nodes and pods can be easily observed.
16 Permissions Management

16.1 Permissions Overview

CCE permissions management allows you to assign permissions to IAM users and user groups under your tenant accounts. CCE combines the advantages of Identity and Access Management (IAM) and Kubernetes Role-based Access Control (RBAC) authorization to provide a variety of authorization methods, including IAM fine-grained authorization, IAM token authorization, cluster-scoped authorization, and namespace-wide authorization.

If you need to perform refined permissions management on CCE clusters and related resources purchased on HUAWEI CLOUD, for example, to control the access of employees in different departments to cloud resources, you can perform multi-dimensional permissions management on CCE.

This section describes the CCE permissions management mechanism and related concepts. If your HUAWEI CLOUD account has met your service requirements, you can skip the configurations in this chapter.

CCE Permissions Management

CCE permissions are described as follows:

- **Cluster-level permissions**: Each user group has the same permissions. On IAM, you can configure system policies to describe which IAM user groups can perform which operations on cluster resources. For example, you can grant user group A to create and delete cluster X, add a node, or install an add-on, while granting user group B to view information about cluster X.

Cluster-level permissions involve CCE non-Kubernetes APIs and support fine-grained IAM policies and enterprise project management capabilities.

- **Namespace-level permissions**: You can regulate users’ or user groups’ access to Kubernetes resources in a single namespace based on their Kubernetes RBAC roles. CCE has also been enhanced based on open-source capabilities. It supports RBAC authorization based on IAM user or user group, and RBAC authentication on access to APIs using IAM tokens.
Namespace-level permissions involve CCE Kubernetes APIs and are enhanced based on the Kubernetes RBAC capabilities. Namespace-level permissions can be granted to IAM users or user groups for authentication and authorization, but are independent of fine-grained IAM policies.

Starting from version 1.11.7-r2, CCE clusters allow you to configure namespace permissions. Clusters earlier than v1.11.7-r2 have all namespace permissions by default.

In general, you configure CCE permissions in two scenarios. The first is creating and managing clusters and related resources, such as nodes. The second is creating and using Kubernetes resources in the cluster, such as workloads and Services.

**Figure 16-1** Illustration on CCE permissions

These permissions allow you to manage resource users at a finer granularity.

**Cluster Permissions (IAM-based) and Namespace Permissions (Kubernetes RBAC-based)**

Users with different cluster permissions (assigned using IAM) have different namespace permissions (assigned using Kubernetes RBAC). Table 16-1 lists the namespace permissions of different users.
Table 16-1 Differences in namespace permissions

<table>
<thead>
<tr>
<th>User</th>
<th>Clusters Earlier Than v1.11.7-r2</th>
<th>Clusters of v1.11.7-r2 and Later</th>
</tr>
</thead>
</table>
| User with the Tenant Administrator permissions (for example, a HUAWEI CLOUD account) | All namespace permissions | ● Has all namespace permissions when using CCE on the console.  
● Requires Kubernetes RBAC authorization when using CCE via `kubectl`.  
NOTE When such a user accesses the CCE console, an administrator group is added. Therefore, the user has all namespace permissions. |
| IAM user with the CCE Administrator role | All namespace permissions | ● Has all namespace permissions when using CCE on the console.  
● Requires Kubernetes RBAC authorization when using CCE via `kubectl`.  
NOTE When such a user accesses the CCE console, an administrator group is added. Therefore, the user has all namespace permissions. |
| IAM user with the CCE FullAccess or CCE ReadOnlyAccess role | All namespace permissions | Requires Kubernetes RBAC authorization. |
| IAM user with the Tenant Guest role | All namespace permissions | Requires Kubernetes RBAC authorization. |

**kubectl Permissions**

You can use `kubectl` to access Kubernetes resources in a cluster.

When you access a cluster using `kubectl`, CCE uses the `kubeconfig.json` file generated on the cluster for authentication. This file contains user information, based on which CCE determines which Kubernetes resources can be accessed by `kubectl`. The permissions recorded in a `kubeconfig.json` file vary from user to user. The permissions that a user has are listed in Table 16-1.

### 16.2 Cluster Permissions (IAM-based)

CCE cluster permissions are assigned based on IAM system policies and custom policies. You can use user groups to assign permissions to IAM users.
Cluster permissions are configured only for cluster-related resources (such as clusters and nodes). You must also configure namespace permissions to operate Kubernetes resources (such as workloads and Services).

Prerequisites

- Before granting permissions to user groups, get familiar with the system policies listed in Permissions Management. For the system policies of other services, see System Permissions.
- A user with the Security Administrator role (for example, a HUAWEI CLOUD account) has all IAM permissions except role switching. Only these users can view user groups and their permissions on the Permissions Management page on the CCE console.

Configuration

On the CCE console, when you choose Permissions Management > Cluster-Level Permissions to create a user group, you will be directed to the IAM console to complete the process. After the user group is created and its permissions are configured, you can view the information on the Cluster-Level Permissions tab page. This section describes the operations in IAM.

Process Flow

**Figure 16-2 Process of assigning CCE permissions**

1. **Create a user group and assign permissions** to it.
   
   Create a user group on the IAM console, and assign CCE permissions, for example, the CCEReadOnlyAccess policy to the group.
2. **Create a user and add it to a user group.**
   Create a user on the IAM console and add the user to the group created in 1.

3. **Log in** and verify permissions.
   Log in to the management console as the user you created, and verify that the user has the assigned permissions.
   - Log in to the management console and switch to the CCE console. Click **Buy Cluster** in the upper right corner. If you fail to do so (assuming that only the CCEReadOnlyAccess role is assigned), the permission control policy takes effect.
   - Switch to the console of any other service. If a message appears indicating that you do not have the required permissions to access the service, the CCEReadOnlyAccess policy takes effect.

### Custom Policies

Custom policies can be created as a supplement to the system-defined policies of CCE. For the actions that can be added to custom policies, see [Permissions Policies and Supported Actions](#).

You can create custom policies in either of the following ways:

- Visual editor: Select cloud services, actions, resources, and request conditions. This does not require knowledge of policy syntax.
- JSON: Edit JSON policies from scratch or based on an existing policy.

For details, see [Creating a Custom Policy](#). This section provides examples of common custom CCE policies.

#### Example Custom Policies:

- **Example 1: Creating a cluster named test**
  ```json
  
  "Version": "1.1",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": ["cce:cluster:create"
    }
  ]
  
  ```

- **Example 2: Denying node deletion**
  A policy with only "Deny" permissions must be used in conjunction with other policies to take effect. If the permissions assigned to a user contain both "Allow" and "Deny", the "Deny" permissions take precedence over the "Allow" permissions.

  The following method can be used if you need to assign permissions of the **CCEFullAccess** policy to a user but you want to prevent the user from deleting nodes (**cce:node:delete**). Create a custom policy for denying node deletion, and attach both policies to the group to which the user belongs.

  Then, the user can perform all operations on CCE except deleting nodes. The following is an example of a deny policy:

  ```json
  
  "Version": "1.1",
  
  ```
• Example 3: Defining permissions for multiple services in a policy

A custom policy can contain the actions of multiple services that are of the global or project-level type. The following is an example policy containing actions of multiple services:

```json
{
  "Version": "1.1",
  "Statement": [
    {
      "Action": [
        "ecs:cloudServers:resize",
        "ecs:cloudServers:delete",
        "ecs:cloudServers:delete",
        "ims:images:list",
        "ims:serverImages:create"
      ],
      "Effect": "Allow"
    }
  ]
}
```

CCE Cluster Permissions and Enterprise Projects

CCE supports resource management and permission allocation by cluster and enterprise project.

Note that:

- IAM projects are based on physical isolation of resources, whereas enterprise projects provide global logical groups of resources, which better meet the actual requirements of enterprises. In addition, IAM policies can be managed based on enterprise projects. Therefore, you are advised to use enterprise projects for permissions management. For details, see Creating an Enterprise Project.

- When there are both IAM projects and enterprise projects, IAM preferentially matches the IAM project policies.

- When creating a cluster or node using purchased cloud resources, ensure that IAM users have been granted the required permissions in the enterprise project to use these resources. Otherwise, the cluster or node may fail to be created.

CCE Cluster Permissions and IAM RBAC

CCE is compatible with IAM system roles for permissions management. You are advised to use fine-grained policies provided by IAM to simplify permissions management.

CCE supports the following roles:

- Basic IAM roles:
- **te_admin (Tenant Administrator):** Users with this role can call all APIs of all services except IAM.
- **readonly (Tenant Guest):** Users with this role can call APIs with the read-only permissions of all services except IAM.

- **Custom CCE administrator role:** CCE Administrator
- **CCE APIs** are designed compatible with three legacy system roles of **ServiceStage** (SvcStg Administrator, SvcStg Developer, and SvcStg Operator). Currently, CCE and ServiceStage have fully adapted to fine-grained policies of IAM for permissions management. Therefore, you are not advised to use these legacy roles for permissions management. These ServiceStage roles are described as follows:
  - **SvcStg Administrator:** This role has the same permissions as the CCE Administrator role except that users with this role do not have the namespace-level permissions by default (Kubernetes RBAC).
  - **SvcStg Developer:** This role has the same permissions as the CCE Administrator role except that the user with this role does not have the namespace-level permission by default (Kubernetes RBAC).
  - **SvcStg Operator:** This role has the read-only permissions on CCE, excluding the namespace-level permissions of clusters by default.

For more information about ServiceStage and CCE permissions management, see [ServiceStage Permissions Management](#) and [CCE Permissions Management](#).

**NOTE**

- Tenant Administrator and Tenant Guest are special IAM system roles. After any system or custom policy is configured, Tenant Administrator and Tenant Guest take effect as system policies to achieve compatibility with IAM RBAC and ABAC scenarios.
- If a user has the Tenant Administrator or CCE Administrator system role, the user has the cluster-admin permissions in Kubernetes RBAC and the permissions cannot be removed after the cluster is created.
  
  If the user is the cluster creator, the cluster-admin permissions in Kubernetes RBAC are granted to the user by default. The permissions can be manually removed after the cluster is created.

  - **Method 1:** Choose [Permissions Management > Namespace-Level Permissions > Delete](#) at the same role as cluster-creator on the CCE console.
  - **Method 2:** Delete [ClusterRoleBinding: cluster-creator](#) through the API or kubectl.

When RBAC and IAM policies co-exist, the backend authentication logic for open APIs or console operations on CCE is as follows:
16.3 Namespace Permissions (Kubernetes RBAC-based)

Namespace Permissions (Kubernetes RBAC-based)

You can regulate users' or user groups' access to Kubernetes resources in a single namespace based on their Kubernetes RBAC roles. The RBAC API declares four kinds of Kubernetes objects: Role, ClusterRole, RoleBinding, and ClusterRoleBinding, which are described as follows:

- **Role**: defines a set of rules for accessing Kubernetes resources in a namespace.
- **RoleBinding**: defines the relationship between users and roles.
- **ClusterRole**: defines a set of rules for accessing Kubernetes resources in a cluster (including all namespaces).
- **ClusterRoleBinding**: defines the relationship between users and cluster roles.

Role and ClusterRole specify actions that can be performed on specific resources. RoleBinding and ClusterRoleBinding bind roles to specific users, user groups, or ServiceAccounts. Illustration:

---

**CAUTION**

Certain CCE APIs involve namespace-level permissions or key operations and therefore, they require special permissions:

Using clusterCert to obtain the cluster kubeconfig: cceadm/teadmin
On the CCE console, you can assign permissions to a user or user group to access resources in one or multiple namespaces. By default, the CCE console provides the following five ClusterRoles:

- **view**: has the permission to view namespace resources.
- **edit**: has the permission to modify namespace resources.
- **admin**: has all permissions on the namespace.
- **cluster-admin**: has all permissions on the cluster.
- **psp-global**: controls sensitive security aspects of the pod specification. For details, see [Pod Security Policies](#).

### Cluster Permissions (IAM-based) and Namespace Permissions (Kubernetes RBAC-based)

Users with different cluster permissions (assigned using IAM) have different namespace permissions (assigned using Kubernetes RBAC). Table 16-2 lists the namespace permissions of different users.

<table>
<thead>
<tr>
<th>User</th>
<th>Clusters Earlier Than v1.11.7-r2</th>
<th>Clusters of v1.11.7-r2 and Later</th>
</tr>
</thead>
</table>
| User with the Tenant Administrator permissions (for example, a HUAWEI CLOUD account) | All namespace permissions | • Has all namespace permissions when using CCE on the console.  
• Requires Kubernetes RBAC authorization when using CCE via `kubectl`.  
**NOTE**  
When such a user accesses the CCE console, an administrator group is added. Therefore, the user has all namespace permissions. |

![Figure 16-3 Role binding](image)
<table>
<thead>
<tr>
<th>User</th>
<th>Clusters Earlier Than v1.11.7-r2</th>
<th>Clusters of v1.11.7-r2 and Later</th>
</tr>
</thead>
<tbody>
<tr>
<td>IAM user with the CCE Administrator role</td>
<td>All namespace permissions</td>
<td>• Has all namespace permissions when using CCE on the console.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Requires Kubernetes RBAC authorization when using CCE via <code>kubectl</code>.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>NOTE</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>When such a user accesses the CCE console, an administrator group is added. Therefore, the user has all namespace permissions.</td>
</tr>
<tr>
<td>IAM user with the CCE FullAccess or CCE ReadOnlyAccess role</td>
<td>All namespace permissions</td>
<td>Requires Kubernetes RBAC authorization.</td>
</tr>
<tr>
<td>IAM user with the Tenant Guest role</td>
<td>All namespace permissions</td>
<td>Requires Kubernetes RBAC authorization.</td>
</tr>
</tbody>
</table>

### Prerequisites

- Kubernetes RBAC authorization can be used for clusters of v1.11.7-r2 and later. Ensure that you have deployed a supported cluster version. For details about upgrading a cluster, see [Upgrading a Cluster Using the Console](#).
- After you create a cluster of v1.11.7-r2 or later, CCE automatically assigns the cluster-admin permissions of all namespaces in the cluster to you, which means you have full control on the cluster and all resources in all namespaces.
- A user with the Security Administrator role has all IAM permissions except role switching. For example, a HUAWEI CLOUD account in the admin user group has this role by default. Only these users can assign permissions on the Permissions Management page on the CCE console.

### Configuring Namespace Permissions (on the Console)

You can regulate users' or user groups' access to Kubernetes resources in a single namespace based on their Kubernetes RBAC roles.

**Step 1** Log in to the CCE console. In the navigation pane, choose **Permissions Management**.

**Step 2** On the displayed page, click the **Namespace-Level Permissions** tab. In the upper right corner of the namespace permissions list, select the cluster that contains the namespace whose access will be managed, and click **Add Permissions**.

**Step 3** Confirm the cluster name and select the namespace to assign permissions for. For example, select **All namespaces**, the target user or user group, and select the permissions, as shown in the following figure..
Figure 16-4 Configuring namespace permissions

Step 4  Click Create.

----End

Using kubectl to Configure Namespace Permissions

NOTE

When you access a cluster using kubectl, CCE uses the kubeconfig.json file generated on the cluster for authentication. This file contains user information, based on which CCE determines which Kubernetes resources can be accessed by kubectl. The permissions recorded in a kubeconfig.json file vary from user to user. The permissions that a user has are listed in Cluster Permissions (IAM-based) and Namespace Permissions (Kubernetes RBAC-based).

In addition to cluster-admin, admin, edit, and view, you can define Roles and RoleBindings to configure the permissions to add, delete, modify, and query resources, such as pods, Deployments, and Services, in the namespace.

The procedure for creating a Role is very simple. To be specific, specify a namespace and then define rules. The rules in the following example are to allow GET and LIST operations on pods in the default namespace.

```
kind: Role
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  namespace: default  # Namespace
  name: role-example
rules:
  - apiGroups: ["""]
    resources: ["pods"]  # The pod can be accessed.
    verbs: ["get", "list"]  # The GET and LIST operations can be performed.
```

- **apiGroups** indicates the API group to which the resource belongs.
- **resources** indicates the resources that can be operated. Pods, Deployments, ConfigMaps, and other Kubernetes resources are supported.
- **verbs** indicates the operations that can be performed. **get** indicates querying a specific object, and **list** indicates listing all objects of a certain type. Other value options include **create**, **update**, and **delete**.

For details, see Using RBAC Authorization.

After creating a Role, you can bind the Role to a specific user, which is called RoleBinding. The following is an example.
kind: RoleBinding
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: RoleBinding-example
  namespace: default
  annotations:
    CCE.com/IAM: 'true'
roleRef:
  kind: Role
  name: role-example
  apiGroup: rbac.authorization.k8s.io
subjects:
  - kind: User
    name: 0c97ac3cb280f4d91fa7c0096739e1f8 # User ID of the user-example
    apiGroup: rbac.authorization.k8s.io

The **subjects** section binds a Role with an IAM user so that the IAM user can obtain the permissions defined in the Role, as shown in the following figure.

**Figure 16-5** A RoleBinding binds the Role to the user.

You can also specify a user group in the **subjects** section. In this case, all users in the user group obtain the permissions defined in the Role.

subjects:
  - kind: Group
    name: 0c96fad22880f32a3f84c009862af6f7 # User group ID
    apiGroup: rbac.authorization.k8s.io

Use the IAM user user-example to connect to the cluster and obtain the pod information. The following is an example of the returned pod information.

```
# kubectl get pod
```

<table>
<thead>
<tr>
<th>NAME</th>
<th>READY</th>
<th>STATUS</th>
<th>RESTARTS</th>
<th>AGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>deployment-389584-2-6f6bd4c574-2n9rk</td>
<td>1/1</td>
<td>Running</td>
<td>0</td>
<td>4d7h</td>
</tr>
<tr>
<td>deployment-389584-2-6f6bd4c574-7s5qw</td>
<td>1/1</td>
<td>Running</td>
<td>0</td>
<td>4d7h</td>
</tr>
<tr>
<td>deployment-3895841-746b97b455-86g77</td>
<td>1/1</td>
<td>Running</td>
<td>0</td>
<td>4d7h</td>
</tr>
<tr>
<td>deployment-3895841-746b97b455-twvpn</td>
<td>1/1</td>
<td>Running</td>
<td>0</td>
<td>4d7h</td>
</tr>
<tr>
<td>nginx-658dff48ff-7rkhph</td>
<td>1/1</td>
<td>Running</td>
<td>0</td>
<td>4d9h</td>
</tr>
<tr>
<td>nginx-658dff48ff-njdjh</td>
<td>1/1</td>
<td>Running</td>
<td>0</td>
<td>4d9h</td>
</tr>
</tbody>
</table>

Try querying Deployments and Services in the namespace. The output shows **user-example** does not have the required permissions. Try querying the pods in namespace kube-system. The output shows **user-example** does not have the required permissions, neither. This indicates that the IAM user **user-example** has only the GET and LIST Pod permissions in the default namespace, which is the same as expected.

```
# kubectl get deploy
```

Error from server (Forbidden): deployments.apps is forbidden: User "0c97ac3cb280f4d91fa7c0096739e1f8"
cannot list resource "deployments" in API group "apps" in the namespace "default"

# kubectl get svc
Error from server (Forbidden): services is forbidden: User "0c97ac3cb280f4d91fa7c0096739e1f8" cannot list resource "services" in API group "" in the namespace "default"

# kubectl get pod --namespace=kube-system
Error from server (Forbidden): pods is forbidden: User "0c97ac3cb280f4d91fa7c0096739e1f8" cannot list resource "pods" in API group "" in the namespace "kube-system"

Example: Assigning All Cluster Permissions (cluster-admin)

You can use the cluster-admin role to assign all permissions on a cluster. This role contains the permissions for cluster resources (such as PVs and StorageClasses).

Figure 16-6 Assigning all cluster permissions (cluster-admin)

In the following example kubectl output, a ClusterRoleBinding has been created and binds the cluster-admin role to the user group cce-role-group.

```yaml
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  annotations:
    CCE.com/IAM: "true"
  creationTimestamp: "2021-06-23T09:15:22Z"
  name: clusterrole_cluster-admin_group0c96fad22880f32a3f84c009862af6f7
  resourceVersion: "36659058"
  selfLink: /apis/rbac.authorization.k8s.io/v1/clusterrolebindings/clusterrole_cluster-admin_group0c96fad22880f32a3f84c009862af6f7
  uid: d6cd43e9-b4ca-4b56-bc52-e36346fc1320
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: cluster-admin
subjects:
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: 0c96fad22880f32a3f84c009862af6f7
```

Connect to the cluster as an authorized user. If the PVs and StorageClasses can be queried, the permission configuration takes effect.

```bash
# kubectl get pv
No resources found
# kubectl get sc
NAME PROVISIONER ALLOWVOLUMEEXPANSION AGE RECLAIMPOLICY VOLUMEBINDINGMODE
csi-disk everest-csi-provisioner Delete Delete Immediate true true 75d
csi-disk-topology everest-csi-provisioner Delete Delete Immediate true true 75d
csi-nas everest-csi-provisioner Delete Delete Immediate true true 75d
csi-obs everest-csi-provisioner Delete Delete Immediate false false 75d
csi-sfsturbo everest-csi-provisioner Delete Delete Immediate true true 75d
```
Example: Assigning All Namespace Permissions (admin)

The admin role contains all permissions on a namespace. You can assign permissions to users to access one or multiple namespaces.

**Figure 16-7 Assigning all namespace permissions (admin)**

In the following example kubectl output, a RoleBinding has been created, the admin role is bound to the user group **cce-role-group**, and the target namespace is the default namespace.

```
# kubectl get rolebinding
NAME                                                      ROLE                AGE
clusterrole_admin_group0c96fad22880f32a3f84c009862af6f7   ClusterRole/admin   18s
# kubectl get rolebinding clusterrole_admin_group0c96fad22880f32a3f84c009862af6f7 -o yaml
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  annotations:
    CCE.com/IAM: "true"
  creationTimestamp: "2021-06-24T01:30:08Z"
  name: clusterrole_admin_group0c96fad22880f32a3f84c009862af6f7
  namespace: default
  resourceVersion: "36963685"
  selfLink: /apis/rbac.authorization.k8s.io/v1/namespaces/default/rolebindings/
  uid: 6c6f46a6-8584-47d9-83f9-9eeef17b75d6
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: admin
subjects:
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: 0c96fad22880f32a3f84c009862af6f7

Connect to a cluster as an authorized user. In this example, you can create and query resources in the default namespace, but cannot query resources in the kube-system namespace or cluster resources.

```
# kubectl get pod
NAME                    READY   STATUS    RESTARTS   AGE
test-568d96f4f8-brdp   1/1     Running   0          33m
test-568d96f4f8-cgjqp   1/1     Running   0          33m
# kubectl get pod -nkube-system
Error from server (Forbidden): pods is forbidden: User "0c97ac3cb280f4d91fa7c0096739e1f8" cannot list resource "pods" in API group "" in the namespace "kube-system"
# kubectl get pv
Error from server (Forbidden): persistentvolumes is forbidden: User "0c97ac3cb280f4d91fa7c0096739e1f8" cannot list resource "persistentvolumes" in API group "" at the cluster scope
```

Example: Assigning Read-Only Namespace Permissions (view)

The view role has the read-only permissions on a namespace. You can assign permissions to users to view one or multiple namespaces.
In the following example kubectl output, a RoleBinding has been created, the view role is bound to the user group `cce-role-group`, and the target namespace is the default namespace.

```
# kubectl get rolebinding
NAME                                                     ROLE               AGE
clusterrole_view_group0c96fad22880f32a3f84c009862af6f7   ClusterRole/view   7s
# kubectl get rolebinding clusterrole_view_group0c96fad22880f32a3f84c009862af6f7 -oyaml
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  annotations:
    CCE.com/IAM: "true"
  creationTimestamp: "2021-06-24T01:36:53Z"
  name: clusterrole_view_group0c96fad22880f32a3f84c009862af6f7
  namespace: default
  resourceVersion: "36965800"
  selfLink: /apis/rbac.authorization.k8s.io/v1/namespaces/default/rolebindings/
  clusterrole_view_group0c96fad22880f32a3f84c009862af6f7
  uid: b86e2507-e735-494c-be55-c41a0c4ef0dd
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: view
subjects:
- apiGroup: rbac.authorization.k8s.io
  kind: Group
  name: 0c96fad22880f32a3f84c009862af6f7
```

Connect to the cluster as an authorized user. In this example, you can query resources in the default namespace but cannot create resources.

```
# kubectl get pod
NAME                    READY   STATUS    RESTARTS   AGE
test-568d96f4f8-brdrp   1/1     Running   0          40m
# kubectl run -i --tty --image tutum/dnsutils dnsutils --restart=Never --rm /bin/sh
Error from server (Forbidden): pods is forbidden: User "0c97ac3cb280f4d91f7a7c0096739e1f8" cannot create resource "pods" in API group "" in the namespace "default"
```

**Example: Assigning Permissions for a Specific Kubernetes Resource Object**

You can assign permissions on a specific Kubernetes resource object, such as pod, Deployment, and Service. For details, see [Using kubectl to Configure Namespace Permissions](#).

---

**Figure 16-8 Assigning read-only namespace permissions (view)**

![Figure 16-8 Assigning read-only namespace permissions (view)](image-url)
16.4 Example: Designing and Configuring Permissions for Users in a Department

Overview

The conventional distributed task scheduling mode is being replaced by Kubernetes. HUAWEI CLOUD CCE allows you to easily deploy, manage, and scale containerized applications in the cloud by providing support for you to use Kubernetes.

To help enterprise administrators manage resource permissions in clusters, CCE provides multi-dimensional, fine-grained permission policies and management measures. CCE permissions are described as follows:

- **Cluster-level permissions**: allowing a user group to perform operations on clusters, nodes, node pools, charts, and add-ons. These permissions are assigned based on IAM system policies.

- **Namespace-level permissions**: allowing a user or user group to perform operations on Kubernetes resources, such as workloads, networking, storage, and namespaces. These permissions are assigned based on Kubernetes RBAC.

Cluster permissions and namespace permissions are independent of each other but must be used together. The permissions set for a user group apply to all users in the user group. When multiple permissions are added to a user or user group, they take effect at the same time (the union set is used).

Permission Design

The following uses company X as an example.

Generally, a company has multiple departments or projects, and each department has multiple members. Therefore, you need to design how permissions are to be assigned to different groups and projects, and set a user name for each member to facilitate subsequent user group and permissions configuration.

The following figure shows the organizational structure of a department in a company and the permissions to be assigned to each member:
Director: David

David is a department director of company X. To assign him all CCE permissions (both cluster and namespace permissions), you need to create the **cce-admin** user group for David on the IAM console and assign the CCE Administrator role.

### NOTE

**CCE Administrator**: This role has all CCE permissions. You do not need to assign other permissions.

**CCE FullAccess and CCE ReadOnlyAccess**: These policies are related to cluster management permissions and configured only for cluster-related resources (such as clusters and nodes). You must also configure namespace permissions to perform operations on Kubernetes resources (such as workloads and Services).

Figure 16-9 Assigning permissions to the user group to which David belongs
**O&M Leader: James**

James is the O&M team leader of the department. He needs the cluster permissions for all projects and the read-only permissions for all namespaces.

To assign the permissions, create a user group named `cce-sre` on the IAM console and add James to this user group. Then, assign CCE FullAccess to the user group `cce-sre` to allow it to perform operations on clusters in all projects.

*Figure 16-10* Assigning permissions to the user group to which James belongs

---

**Assigning Read-only Permissions on All Clusters and Namespaces to All Team Leaders and Engineers**

You can create a read-only user group named `read_only` on the IAM console and add users to the user group.

- Although the development engineers Linda and Peter do not require cluster management permissions, they still need to view data on the CCE console. Therefore, the read-only cluster permission is required.
- For the O&M engineer William, assign the read-only permission on clusters to him in this step.
- The O&M team leader James already has the management permissions on all clusters. You can add him to the `read_only` user group to assign the read-only permission on clusters to him.

As shown in the following figure, users James, Robert, William, Linda, and Peter are added to the `read_only` user group.
Assign the read-only permission on clusters to the user group **read_only**.

**Figure 16-12 Assigning the read-only permission on clusters to the user group**

Return to the CCE console, and add the read-only permission on namespaces to the user group **read_only** to which the five users belong. Choose **Permissions Management > Namespace-Level Permissions** on the CCE console, and assign the view policy to the user group **read_only** for each cluster.
Figure 16-13 Assigning the read-only permission on namespaces to the user group

![Add Permissions](image)

After the setting is complete, James has the cluster management permissions for all projects and the read-only permissions on all namespaces, and the Robert, William, Linda, and Peter have the read-only permission on all clusters and namespaces.

**Development Team Leader: Robert**

In the previous steps, Robert has been assigned the read-only permission on all clusters and namespaces. Now, assign the management permissions on all namespaces to Robert.

Log in to the CCE console and choose Permissions Management > Namespace-Level Permissions to assign the admin policy to Robert for each cluster.

Figure 16-14 Assigning the management permissions on namespaces to Robert

![Add Permissions](image)
O&M Engineer: William

In the previous steps, William has been assigned the read-only permission on all clusters and namespaces. He also requires the cluster management permissions in region Beijing4. Therefore, you can log in to the IAM console, create a user group named `cce-sre-b4` and assign CCE FullAccess to William for region Beijing4.

**Figure 16-15** Assigning the cluster management permissions for Beijing4 region to the user group to which WILLIAM belongs

Now, William has the cluster management permissions for Beijing4 region and the read-only permission on all namespaces.

Development Engineers: Linda and Peter

In the previous steps, Linda and Peter have been assigned the read-only permission on clusters and namespaces. Therefore, you only need to assign the edit policy to them by choosing Permissions Management > Namespace-Level Permissions on the CCE console.

**Figure 16-16** Assigning the edit policy on namespaces
By now, all the required permissions are assigned to the department members.

FAQs

1. **Can I configure only namespace permissions without cluster management permissions?**
   
   No. The permissions you have to perform operations on the console are determined by the IAM system policy. Therefore, if the cluster management permissions are not configured, you do not have the permissions to access the CCE console.

2. **Can I use CCE APIs if the cluster management permissions are not configured?**

   No. API calls must be authenticated by IAM using tokens.

3. **Can I use kubectl if the cluster management permissions are not configured?**

   Yes. The prerequisite is that the kubectl configuration file has been downloaded from the console. Therefore, if you have configured cluster permissions, after you download the kubectl configuration file from the CCE console and then delete the cluster permissions (reserving the namespace permissions), you can still use kubectl to perform operations on Kubernetes clusters.

### 16.5 Permission Dependency of the CCE Console

Some CCE permissions policies depend on the policies of other cloud services. To view or use other cloud resources on the CCE console, you need to enable the system policy access control feature of IAM and assign dependency policies for the other cloud services.

- Dependency policies are assigned based on the CCE FullAccess or CCE ReadOnlyAccess policy you configure. For details, see [Cluster Permissions (IAM-based)](#).
- Only users and user groups with namespace permissions can gain the view access to resources in clusters of v1.11.7-r2 and later.
  - If a user is granted the view access to all namespaces of a cluster, the user can view all namespaced resources (except secrets) in the cluster. To view secrets in the cluster, the user must gain the **admin** or **edit** role in all namespaces of the cluster.
  - CustomedHPA and HPA policies take effect only after the cluster-admin permissions are configured for the namespace.
  - The **view** role within a single namespace allows users to view resources only in the specified namespace.

**Dependency Policy Configuration**

To grant an IAM user the permissions to view or use resources of other cloud services on the CCE console, you must first grant the CCE Administrator, CCE FullAccess, or CCE ReadOnlyAccess policy to the user group to which the user belongs and then grant the dependency policies listed in [Table 16-3](#) to the user.
These dependency policies will allow the IAM user to access resources of other cloud services.

**NOTE**

**Enterprise projects** can group and manage resources across different projects of an enterprise. Resources are thereby isolated. IAM allows you to implement fine-grained authorization. It is strongly recommended that you use IAM for permissions management.

If you use an enterprise project to set permissions for IAM users, the following restrictions apply:

- Storage resources are not available to enterprise projects. Therefore, when an enterprise project user queries the storage resources of clusters in an enterprise project other than Default on the CCE console, all buttons on the page are in gray and resources are not displayed.
- On the CCE console, enterprise projects cannot call the API used to obtain AOM monitoring data for cluster monitoring. Therefore, IAM users in these enterprise projects cannot query monitoring data.
- On the CCE console, enterprise projects cannot call the API to query the key pair created during node creation. Therefore, IAM users in these enterprise projects cannot use the key pair login mode. Only the password login mode is supported.
- After assigning the **CCE FullAccess** permission to an enterprise project, you need to configure the `ecs:availabilityZones:list` action on the IAM console so that the enterprise project users can create nodes. Otherwise, the system will prompt that they do not have the permission.

CCE supports fine-grained permissions configuration, but has the following restrictions:

- AOM does not support resource-level monitoring. After operation permissions on specific resources are configured using IAM's fine-grained cluster resource management function, IAM users can view cluster monitoring information on the **Dashboard** page of the CCE console, but cannot view the data on non-fine-grained metrics.

---

### Table 16-3 Dependency policies

<table>
<thead>
<tr>
<th>Console Function</th>
<th>Dependent Services</th>
<th>Roles or Policies Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dashboard</td>
<td>Application Operations Management (AOM)</td>
<td>• An IAM user with CCE Administrator assigned can use this function only after AOM FullAccess policy is assigned.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• IAM users with IAM ReadOnlyAccess, CCE FullAccess, or CCE ReadOnlyAccess assigned can directly use this function.</td>
</tr>
<tr>
<td>Console Function</td>
<td>Dependent Services</td>
<td>Roles or Policies Required</td>
</tr>
<tr>
<td>--------------------------</td>
<td>---------------------------------------------------------</td>
<td>------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Workload management</td>
<td>Elastic Load Balance (ELB)</td>
<td>Except in the following cases, the user does not require any additional role to create workloads.</td>
</tr>
<tr>
<td></td>
<td>Application Performance Management (APM)</td>
<td>• To create a Service using ELB, you must have ELB FullAccess or ELB Administrator plus VPC Administrator assigned.</td>
</tr>
<tr>
<td></td>
<td>Application Operations Management (AOM)</td>
<td>• To use a Java probe, you must have AOM FullAccess and APM FullAccess assigned.</td>
</tr>
<tr>
<td></td>
<td>NAT Gateway</td>
<td>• To create a Service using NAT Gateway, you must have NAT Gateway Administrator assigned.</td>
</tr>
<tr>
<td></td>
<td>Object Storage Service (OBS)</td>
<td>• To use OBS, you must have OBS Administrator globally assigned.</td>
</tr>
<tr>
<td></td>
<td>Scalable File Service (SFS)</td>
<td><strong>NOTE</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Because of the cache, it takes about 13 minutes for the RBAC policy to take effect after being granted to users, user groups, and enterprise projects. After an OBS-related system policy is granted, it takes about 5 minutes for the policy to take effect.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• To use SFS, you must have SFS FullAccess assigned.</td>
</tr>
<tr>
<td>Cluster management</td>
<td>Application Operations Management (AOM)</td>
<td>• Auto scale-out or scale-up requires the AOM FullAccess policy.</td>
</tr>
<tr>
<td></td>
<td>Billing Center (BSS)</td>
<td>• Changing the billing mode to yearly/monthly requires the BSS Administrator role.</td>
</tr>
<tr>
<td>Node management</td>
<td>Elastic Cloud Server (ECS)</td>
<td>If the permission assigned to an IAM user is CCE Administrator, creating or deleting a node requires the ECS FullAccess or ECS Administrator policy and the VPC Administrator policy.</td>
</tr>
<tr>
<td>Network management</td>
<td>Elastic Load Balance (ELB)</td>
<td>Except in the following cases, the user does not require any additional role to create a Service.</td>
</tr>
<tr>
<td></td>
<td>NAT Gateway</td>
<td>• To create a Service using ELB, you must have ELB FullAccess or ELB Administrator plus VPC Administrator assigned.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• To create a Service using NAT Gateway, you must have NAT Administrator assigned.</td>
</tr>
<tr>
<td>Console Function</td>
<td>Dependent Services</td>
<td>Roles or Policies Required</td>
</tr>
<tr>
<td>------------------</td>
<td>--------------------</td>
<td>-----------------------------</td>
</tr>
</tbody>
</table>
| Storage management | Object Storage Service (OBS) Scalable File Service (SFS) | • To use OBS, you must have OBS Administrator globally assigned.  
**NOTE** Because of the cache, it takes about 13 minutes for the RBAC policy to take effect after being granted to users, user groups, and enterprise projects. After an OBS-related system policy is granted, it takes about 5 minutes for the policy to take effect.  
• To use SFS, you must have SFS FullAccess assigned. |
| Namespace management | / | / |
| Chart management | / | HUAWEI CLOUD accounts and the IAM users with CCE Administrator assigned can use this function. |
| Add-on management | / | HUAWEI CLOUD accounts and the IAM users with CCE Administrator, CCE FullAccess, or CCE ReadOnlyAccess assigned can use this function. |
| Permissions management | / | • HUAWEI CLOUD accounts can use this function.  
• IAM users with CCE Administrator or global Security Administrator assigned can use this function.  
• IAM users with IAM ReadOnlyAccess plus CCE FullAccess or CCE ReadOnlyAccess assigned can use this function. |
| Configuration center | / | • Creating ConfigMaps does not require any additional policy.  
• Viewing secrets requires that the cluster-admin, admin, or edit permission be configured for the namespace. The DEW KeypairFullAccess or DEW KeypairReadOnlyAccess policy must be assigned for dependent services. |
| Help center | / | / |
The CCE console provides links to other related services. To view or use these services, an IAM user must be assigned required permissions for the services.

<table>
<thead>
<tr>
<th>Console Function</th>
<th>Dependent Services</th>
<th>Roles or Policies Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>Switching to other related services</td>
<td>Software Repository for Container (SWR)</td>
<td>The CCE console provides links to other related services. To view or use these services, an IAM user must be assigned required permissions for the services.</td>
</tr>
<tr>
<td></td>
<td>Application Operations Management (AOM)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Multi-Cloud Container Platform (MCP)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Cloud Eye (for viewing metrics of nodes and storage systems)</td>
<td></td>
</tr>
</tbody>
</table>

**Example**

This example describes how to authorize the user James to view monitoring information on the **Dashboard** page of the CCE console.

Assume that the IAM user James belongs to the user group Developers and Developers is already granted the CCE Administrator role. According to **Table 16-3**, to view monitoring information on the **Dashboard** page of the CCE console, the AOM FullAccess policy is additionally required.

**Figure 16-17** Process for granting dependency policies

**Step 1: Verify User's Current Permissions**

**Step 1** Log in to the CCE console as the IAM user **James**. For details on how to log in to the CCE console as an IAM user, see **Logging In as an IAM User**.
Step 2  In the navigation pane, click Dashboard. The monitoring information on the Dashboard page is not available to the user James.

Figure 16-18  Viewing monitoring information (no data available)

----End

Step 2: Grant Permissions to the User Group

Step 1  Log out as IAM user James and log in to the CCE console using your HUAWEI CLOUD account.

Step 2  In the navigation pane, choose Permissions Management. Click the Cluster-Level Permissions tab. In the same row as the user group Developers to which the IAM user James belongs, click Configure Policy.

Figure 16-19  Granting permissions to a user group

Step 3  In the Configure Policy dialog box, click Configure Now.
Step 4 In the navigation pane of the IAM console, choose User Groups. In the user group list, click the user group name Developers to show user group details. On the Permissions tab page, click Assign Permissions.

Step 5 On the page displayed, select the application scope. In this case, select Region-specific projects and then one or more projects in the drop-down list.

Step 6 Search for AOM in the search box above the permissions list, select the AOM FullAccess policy, and click OK.
Figure 16-23 Adding an AOM FullAccess policy

Step 7  Verify that the AOM FullAccess policy is displayed on the Permissions tab page.

Figure 16-24 Verifying whether a permissions policy is successfully added

Step 8  In the navigation pane of the CCE console, choose Permissions Management. Click the Cluster-Level Permissions tab. In the upper right corner of the page, click the refresh icon to refresh the page. The AOM FullAccess policy is displayed in the policy list.

Figure 16-25 Success in adding a policy

----End

Step 3: Verify User's New Permissions

Step 1  Log in to the CCE console as the IAM user James.

Step 2  In the navigation pane, choose Dashboard. The monitoring information on the Dashboard page is now available to James.
Figure 16-26 Monitoring information on the Dashboard page

Step 3 Verify that the Dashboard page also displays information about dependent services to James.

---End

16.6 Pod Security Policies

A pod security policy (PSP) is a cluster-level resource that controls sensitive security aspects of the pod specification. The PodSecurityPolicy object in Kubernetes defines a group of conditions that a pod must comply with to be accepted by the system, as well as the default values of related fields.

By default, the PSP access control component is enabled for clusters of v1.17.17 and a global default PSP named psp-global is created. You can modify the default policy (but not delete it). You can also create a PSP and bind it to the RBAC configuration.

NOTE

In addition to the global default PSP, the system configures independent PSPs for system components in namespace kube-system. Modifying the psp-global configuration does not affect pod creation in namespace kube-system.

Modifying the Global Default PSP

Before modifying the global default PSP, ensure that a CCE cluster has been created and connected by using kubectl.

Step 1 Run the following command:

```
kubectl edit psp psp-global
```

Step 2 Modify the parameters as required. For details, see PodSecurityPolicy.

---End
Example of Enabling Unsafe Sysctls in Pod Security Policy

You can configure allowed-unsafe-sysctls for a node pool. For CCE v1.17.17 and later versions, add configurations in `allowedUnsafeSysctls` of the pod security policy to make the configuration take effect. For details, see PodSecurityPolicy.

In addition to modifying the global pod security policy, you can add new pod security policies. For example, enable the `net.core.somaxconn` unsafe sysctls. The following is an example of adding a pod security policy:

```yaml
apiVersion: policy/v1beta1
kind: PodSecurityPolicy
metadata:
  annotations:
    seccomp.security.alpha.kubernetes.io/allowedProfileNames: "*"
  name: sysctl-psp
spec:
  allowedUnsafeSysctls:
  - net.core.somaxconn
    allowPrivilegeEscalation: true
    allowedCapabilities: 
    - "*"
  fsGroup:
    rule: RunAsAny
  hostIPC: true
  hostNetwork: true
  hostPID: true
  hostPorts:
    - max: 65535
      min: 0
    privileged: true
  runAsGroup:
    rule: RunAsAny
  runAsUser:
    rule: RunAsAny
  seLinux:
    rule: RunAsAny
  supplementalGroups:
    rule: RunAsAny
  volumes:
    - "*"
---
kind: ClusterRole
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: sysctl-psp
rules:
- apiGroups: 
  - ""
  resources: 
  - podsecuritypolicies
  resourceNames:
  - sysctl-psp
  verbs: 
  - use
---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: sysctl-psp
roleRef:
  kind: ClusterRole
  name: sysctl-psp
  apiGroup: rbac.authorization.k8s.io
subjects:
- kind: Group
```
If you have modified the default pod security policy and want to restore the original pod security policy, perform the following operations.

**Step 1** Create a policy description file named `policy.yaml`. `policy.yaml` is an example file name. You can rename it as required.

```
vi policy.yaml
```

The content of the description file is as follows:

```yaml
apiVersion: policy/v1beta1
class: PodSecurityPolicy
description: "The PSP definition is as follows."

metadata:
  name: psp-global
  annotations:
    seccomp.security.alpha.kubernetes.io/allowedProfileNames: "**"

spec:
  privileged: true
  allowPrivilegeEscalation: true
  allowedCapabilities: 
  - "*"
  volumes: 
  - "*"
  hostNetwork: true
  hostPorts:
  - min: 0
  - max: 65535
  hostIPC: true
  hostPID: true
  runAsUser:
    rule: 'RunAsAny'
  seLinux:
    rule: 'RunAsAny'
  supplementalGroups:
    rule: 'RunAsAny'
  fsGroup:
    rule: 'RunAsAny'

---

kind: ClusterRole
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: psp-global
rules:
- apiGroups:
  - "*"
  resources:
  - podsecuritypolicies
  resourceNames:
  - psp-global
  verbs:
  - use

---

apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: psp-global
roleRef:
  kind: ClusterRole
  name: psp-global
  apiGroup: rbac.authorization.k8s.io
subjects:
```
Step 2  Run the following commands:

```bash
kubectl apply -f policy.yaml
```

-----End
17 System Steward

17.1 System Check

Scenario

System Steward consists of system check and system hardening. This topic describes the system check function.

System check detects faults or exceptions on nodes in real time.

Prerequisites

- Before using the system check function, you must install the npd add-on, which is used to detect node exceptions.
- Before using the system check function, you must install the prometheus add-on, which is used to obtain abnormal metrics reported by the npd add-on.

Procedure

Step 1  Log in to the CCE console. In the navigation pane on the left, choose System Steward > System Check.

Step 2  In the left pane of the System Check page, choose the node for which you want to perform a system check. The Indicator Check, Behavior Statistics, and Kubernetes Events tab pages are displayed.

Required add-ons have not been installed:

If the npd and prometheus add-ons are not installed, install them as prompted.

After the add-ons are installed, choose System Steward > System Check again to view the check information.
Figure 17-1 Installing add-ons required for system check

Required add-ons have been installed:

If the add-ons have been installed, you can click the Indicator Check, Behavior Statistics, and Kubernetes Events tabs to view the system check information.

Figure 17-2 Viewing the system check information

Step 3 In the Indicator Check tab page, you can view system resources, system components, abnormal behaviors, and other information, and then perform operations as prompted.

Table 17-1 Precautions for creating a cluster

<table>
<thead>
<tr>
<th>Check Item</th>
<th>Check Sub-item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>System resources</td>
<td>Disk</td>
<td>Node disk usage.</td>
</tr>
<tr>
<td></td>
<td>Memory</td>
<td>Node memory usage.</td>
</tr>
<tr>
<td></td>
<td>PID</td>
<td>Node PID usage.</td>
</tr>
<tr>
<td>System components</td>
<td>CNI</td>
<td>CNI component running status</td>
</tr>
<tr>
<td></td>
<td>Docker</td>
<td>Docker component running status</td>
</tr>
<tr>
<td></td>
<td>kubelet</td>
<td>kubelet component running status</td>
</tr>
<tr>
<td></td>
<td>kube-proxy</td>
<td>kube-proxy component running status</td>
</tr>
<tr>
<td></td>
<td>NTP</td>
<td>Docker component running status</td>
</tr>
</tbody>
</table>
### Recovery Suggestion

- If system resources are insufficient, expand system resources on the node or increase the upper limit of kernel parameters. If the node cannot be recovered, you can add a taint to the node so that pods will not be scheduled to the node or the pods on the node are evicted to isolate the node.

- A taint can be also added if a system component is abnormal or other exceptions occur.

### Reference

- Adding a taint to a node: [Taints and Tolerations](#)
- Safe eviction: [Safely Drain a Node while Respecting the PodDisruptionBudget](#)
- The following three commands can be used to smoothly migrate services from a node to another node during node maintenance, ensuring that services are not affected:

<table>
<thead>
<tr>
<th>Table 17-2 Marking a node as schedulable or unschedulable</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Command</strong></td>
</tr>
<tr>
<td>cordon</td>
</tr>
<tr>
<td>Command</td>
</tr>
<tr>
<td>---------</td>
</tr>
<tr>
<td>uncordon</td>
</tr>
<tr>
<td>drain</td>
</tr>
</tbody>
</table>

## 17.2 System Hardening

### Scenario

System Steward consists of system check and system hardening. This topic describes the system hardening function.

In system hardening, system components, such as the CoreDNS add-on, are hardened. Currently, CCE supports automatic horizontal scaling using the CoreDNS add-on. The number of pods is automatically scaled based on the number of CoreDNS requests to prevent CoreDNS resolution performance deterioration or resolution timeout due to excessive requests.

You can also view the monitoring metrics of the Horizontal Pod Autoscaler (HPA), autoscaler, and Prometheus on System Steward > System Hardening.

### Prerequisites

- Before using the system hardening function, you need to install the prometheus add-on, which is used to monitor the system and report alarms. Horizontal scaling is performed by the coredns add-on based on the custom Prometheus metrics coredns_dns_request_count_total.
- Before using coredns for horizontal scaling, you need to install the coredns add-on, which is a DNS server that provides the domain name resolution service for Kubernetes. coredns chains plug-ins to provide additional features. This add-on is mandatory when you create a cluster. If you have manually deleted it, reinstall it.
- Before viewing HPA monitoring metrics, you need to install the cce-hpa-controller add-on. cce-hpa-controller is a Huawei-developed add-on, which can be used to flexibly scale in or out Deployments based on metrics such as CPU usage and memory usage.
- Before viewing the monitoring metrics of the autoscaler, you need to install the autoscaler add-on. The autoscaler add-on is used to automatically scale in or out nodes in a Kubernetes cluster.

### Configuring Horizontal Scaling Policies Based on coredns

**Step 1** Log in to the CCE console. In the navigation pane on the left, choose System Steward > System Hardening.

**Step 2** On the System Hardening page, the Horizontal coredns Scaling tab page is displayed.
If the `coredns` and `prometheus` add-ons are not installed, install them as prompted on the tab page. After the add-ons are installed, choose **System Steward > System Hardening** and perform operations as required.

If the add-ons have been installed, you can perform operations as required on the **System Hardening** page.

**Step 3** In the **Horizontal coredns Scaling Policy** area, configure the following parameters (parameters marked with an asterisk (*) are mandatory):

**NOTICE**

If the `coredns` and `prometheus` add-ons have not been installed, horizontal scaling policies using the coredns add-on cannot be configured.

**Table 17-3** Configuration Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>* Instances</td>
<td>Set the pod scaling scope based on service requirements.</td>
</tr>
<tr>
<td>Metric</td>
<td>Number of CoreDNS requests, which cannot be changed.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>* Trigger value</td>
<td>You are advised to set the threshold based on the specifications selected during the CoreDNS add-on installation. For example, if the CoreDNS add-on specification is 2500 (concurrent domain name resolution capability: 2500 QPS for external domain names and 10000 QPS for internal domain names), the recommended threshold is 10000.</td>
</tr>
<tr>
<td>Tolerance</td>
<td>The default value is 10%, which cannot be changed. This parameter indicates the fluctuation range of the actual value of monitoring metrics compared to their target value. When the actual value exceeds the fluctuation range, scaling is triggered.</td>
</tr>
</tbody>
</table>

**Step 4** Click **Configure Now**.

-----End

**Related Operations**

After the horizontal coredns scaling policy is configured, you can view the average number of coredns requests per second, policy configuration information, and Kubernetes events.

![相关政策画面](image)

Click **Modify** at the row of **Policy Configuration** to modify the pod quantity range and triggering threshold. After the modification is complete, click **OK** to update the scaling policy configuration.
Click **Delete** at the row of **Policy Configuration**. In the dialog box displayed, click **OK** to delete the scaling policy. Deleted policies cannot be recovered. Exercise caution when performing this operation.
18 Cloud Eye Service (CES)

18.1 Monitoring Metrics

CCE monitors nodes by using Cloud Eye. Each node corresponds to an ECS.

Monitoring is the key to ensure CCE node performance, reliability, and availability. Using monitored data, you can determine node resource usage. You can use Cloud Eye to automatically monitor nodes in real time and manage alarms and notifications, so that you can keep track of node performance metrics.

Table 18-1 Major monitoring metrics

<table>
<thead>
<tr>
<th>Metric</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU usage</td>
<td>CPU usage of the instance (unit: %)</td>
</tr>
<tr>
<td>Disk read rate</td>
<td>Disk read rate of the instance (unit: byte/s)</td>
</tr>
<tr>
<td>Disks Read Requests</td>
<td>Indicates the number of read requests sent to an ECS per second.</td>
</tr>
<tr>
<td>Disk write rate</td>
<td>Indicates the number of bytes written to an ECS per second.</td>
</tr>
<tr>
<td>Disk Write Requests</td>
<td>Indicates the number of write requests sent to an ECS per second.</td>
</tr>
<tr>
<td>Outband Incoming Rate</td>
<td>Indicates the number of incoming bytes on an ECS per second at the virtualization layer.</td>
</tr>
<tr>
<td>Outband Outgoing Rate</td>
<td>Indicates the number of outgoing bytes on an ECS per second at the virtualization layer.</td>
</tr>
</tbody>
</table>

Related Operations

Cloud Eye is a multi-dimensional monitoring platform for resources such as ECSs and bandwidth. With Cloud Eye, you can view the resource usage and service
running status in the cloud, and respond to exceptions in a timely manner to ensure smooth running of services.

## 18.2 Setting Alarm Rules

### Scenario

You can set node alarm rules to customize the monitored objects and notification policies. Then, you can learn node running status in a timely manner.

Node alarm rules include alarm rule names, monitored objects, metric, thresholds, monitoring intervals, and whether to send notifications. This section describes how to set alarm rules.

### Procedure

1. **Step 1** Log in to the management console.
2. **Step 2** Under Management & Deployment, select Cloud Eye.
3. **Step 3** In the navigation pane on the left, choose Alarm Management > Alarm Rules.
4. **Step 4** Click Create Alarm Rule in the upper right corner.
5. **Step 5** On the Create Alarm Rule page, set parameters as prompted.
   1. Set **Name** and **Description**.

   ![Figure 18-1 Creating an alarm rule](image)

### Table 18-2 Configuration parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Name of the alarm rule. The system generates a name randomly but you can change it. Example value: <code>alarm-b6al</code></td>
</tr>
<tr>
<td>Description</td>
<td>Alarm rule description. This parameter is optional.</td>
</tr>
</tbody>
</table>

2. Select a monitored object and set alarm content parameters.
Figure 18-2 Configuring the alarm content

Table 18-3 Parameters for configuring the alarm content

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Example Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Resource Type</td>
<td>Name of the service for which the alarm rule is configured</td>
<td>Elastic Cloud Server</td>
</tr>
<tr>
<td>Dimensions</td>
<td>Metric dimension of the alarm rule.</td>
<td>ECSs</td>
</tr>
</tbody>
</table>
| Monitoring Scope| Resource range to which the alarm rule applies. You can select Resource groups or Specific resources.  
|                 | **NOTE**                                                                   |                                    |
|                 | - If Resource groups is selected and any resource in the group meets the alarm policy, an alarm is triggered.  
|                 | - If you select Specific resources, select one or more monitored objects and click to synchronize the monitored objects to the dialog box on the right.  
| Group           | This parameter is mandatory when Monitoring Scope is set to Resource groups. | N/A                                |
| Method          | Select Use template or Create manually as required.                         | Create manually                    |
| Template        | Select the template to be imported.                                         | N/A                                |
### Parameter Description Example Value

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Example Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alarm Policy</td>
<td>Policy that triggers an alarm. When Resource Type is set to Website Monitoring, Log Monitoring, Custom Monitoring, or a specific cloud service, the alarm policy takes effect periodically. When Resource Type is set to Event Monitoring, the specific event is an instant operation instead of a periodic action. For example, an alarm is triggered if the average CPU usage is 80% or more for three consecutive periods of 5 minutes.</td>
<td>N/A</td>
</tr>
<tr>
<td>Mount Point</td>
<td>This parameter is mandatory when the metric is a fine-grained disk metric. For the Windows OS, enter a drive letter, such as C, D, or E. For the Linux OS, enter a mount point, such as /dev or /opt.</td>
<td>/dev</td>
</tr>
<tr>
<td>Alarm Severity</td>
<td>Severity of an alarm. Valid values are Critical, Major, Minor, and Informational.</td>
<td>Major</td>
</tr>
</tbody>
</table>

3. Set **Alarm Notification** parameters.

**Figure 18-3** Configuring alarm notification

**Table 18-4** Parameters for configuring alarm notification

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Send Notification</td>
<td>Specifies whether to notify users when alarms are triggered. Notifications can be sent by emails or text messages, or HTTP/HTTPS requests sent to the servers.</td>
</tr>
<tr>
<td>Validity Period</td>
<td>Cloud Eye sends notifications only within the validity period specified in the alarm rule. If Validity Period is set to 08:00-20:00, Cloud Eye sends notifications only within 08:00-20:00.</td>
</tr>
</tbody>
</table>
### 18.3 Viewing Metrics

#### Scenario

Cloud Eye provided by the cloud platform monitors node running statuses. You can obtain the node monitoring metrics on the management console.

Monitored data requires a period of time for transmission and display. The node status displayed on the Cloud Eye page is the status obtained 5 to 10 minutes before. You can view the monitored data of a newly created node after 5 to 10 minutes.

#### Procedure

- **Step 1** Log in to the CCE console. In the navigation pane, choose Resource Management > Nodes.
- **Step 2** Click Monitoring next to the node to be monitored and go to the Cloud Eye console to view node monitoring details.

For details, see Viewing Server Monitoring Metrics.

---

Cloud Container Engine
User Guide

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Notification Object</td>
<td>Name of the topic to which the alarm notification is sent. If you enable the notification function, you need to select a topic. If no desired topics are available, you need to create one first, whereupon the SMN service is invoked. For details about how to create a topic, see the Simple Message Notification User Guide.</td>
</tr>
<tr>
<td>Trigger Condition</td>
<td>Condition for triggering the alarm. You can select Generated alarm, Cleared alarm, or both.</td>
</tr>
</tbody>
</table>
Cloud Trace Service (CTS) records operations on cloud service resources, allowing users to query, audit, and backtrack the resource operation requests initiated from the management console or open APIs as well as responses to the requests.

### Table 19-1 CCE operations supported by CTS

<table>
<thead>
<tr>
<th>Operation</th>
<th>Resource Type</th>
<th>Event Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Creating an agency</td>
<td>Cluster</td>
<td>createUserAgencies</td>
</tr>
<tr>
<td>Creating a cluster</td>
<td>Cluster</td>
<td>createCluster</td>
</tr>
<tr>
<td>Creating a yearly/monthly-billed cluster</td>
<td>Cluster</td>
<td>createCluster/createPeriodicCluster</td>
</tr>
<tr>
<td>Updating the description of a cluster</td>
<td>Cluster</td>
<td>updateCluster</td>
</tr>
<tr>
<td>Upgrading a cluster</td>
<td>Cluster</td>
<td>clusterUpgrade</td>
</tr>
<tr>
<td>Deleting a cluster</td>
<td>Cluster</td>
<td>claimCluster/deleteCluster</td>
</tr>
<tr>
<td>Downloading a cluster certificate</td>
<td>Cluster</td>
<td>getClusterCertByUID</td>
</tr>
<tr>
<td>Binding and unbinding an EIP</td>
<td>Cluster</td>
<td>operateMasterEIP</td>
</tr>
<tr>
<td>Waking up a cluster and resetting node management (V2)</td>
<td>Cluster</td>
<td>operateCluster</td>
</tr>
<tr>
<td>Hibernating a cluster (V3)</td>
<td>Cluster</td>
<td>hibernateCluster</td>
</tr>
<tr>
<td>Operation</td>
<td>Resource Type</td>
<td>Event Name</td>
</tr>
<tr>
<td>------------------------------------------------</td>
<td>---------------</td>
<td>-----------------------</td>
</tr>
<tr>
<td>Waking up a cluster (V3)</td>
<td>Cluster</td>
<td>awakeCluster</td>
</tr>
<tr>
<td>Changing the specifications of a pay-per-use cluster</td>
<td>Cluster</td>
<td>resizeCluster</td>
</tr>
<tr>
<td>Changing the specifications of a yearly/monthly-billed cluster</td>
<td>Cluster</td>
<td>resizePeriodCluster</td>
</tr>
<tr>
<td>Modifying configurations of a cluster</td>
<td>Cluster</td>
<td>updateConfiguration</td>
</tr>
<tr>
<td>Creating a node pool</td>
<td>Node pool</td>
<td>createNodePool</td>
</tr>
<tr>
<td>Updating a node pool</td>
<td>Node pool</td>
<td>updateNodePool</td>
</tr>
<tr>
<td>Deleting a node pool</td>
<td>Node pool</td>
<td>claimNodePool</td>
</tr>
<tr>
<td>Migrating a node pool</td>
<td>Node pool</td>
<td>migrateNodepool</td>
</tr>
<tr>
<td>Modifying node pool configurations</td>
<td>Node pool</td>
<td>updateConfiguration</td>
</tr>
<tr>
<td>Creating a node</td>
<td>Node</td>
<td>createNode</td>
</tr>
<tr>
<td>Creating a yearly/monthly-billed node</td>
<td>Node</td>
<td>createPeriodNode</td>
</tr>
<tr>
<td>Deleting all the nodes from a specified cluster</td>
<td>Node</td>
<td>deleteAllHosts</td>
</tr>
<tr>
<td>Deleting a single node</td>
<td>Node</td>
<td>deleteOneHost/claimOneHost</td>
</tr>
<tr>
<td>Updating the description of a node</td>
<td>Node</td>
<td>updateNode</td>
</tr>
<tr>
<td>Creating an add-on instance</td>
<td>Add-on instance</td>
<td>createAddonInstance</td>
</tr>
<tr>
<td>Deleting an add-on instance</td>
<td>Add-on instance</td>
<td>deleteAddonInstance</td>
</tr>
<tr>
<td>Uploading a chart</td>
<td>Chart</td>
<td>uploadChart</td>
</tr>
<tr>
<td>Updating a chart</td>
<td>Chart</td>
<td>updateChart</td>
</tr>
<tr>
<td>Deleting a chart</td>
<td>Chart</td>
<td>deleteChart</td>
</tr>
<tr>
<td>Operation</td>
<td>Resource Type</td>
<td>Event Name</td>
</tr>
<tr>
<td>----------------------</td>
<td>------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>Creating a release</td>
<td>Release</td>
<td>createRelease</td>
</tr>
<tr>
<td>Upgrading a release</td>
<td>Release</td>
<td>updateRelease</td>
</tr>
<tr>
<td>Deleting a release</td>
<td>Release</td>
<td>deleteRelease</td>
</tr>
<tr>
<td>Creating a ConfigMap</td>
<td>Kubernetes resource</td>
<td>createConfigmaps</td>
</tr>
<tr>
<td>Creating a DaemonSet</td>
<td>Kubernetes resource</td>
<td>createDaemonsets</td>
</tr>
<tr>
<td>Creating a Deployment</td>
<td>Kubernetes resource</td>
<td>createDeployments</td>
</tr>
<tr>
<td>Creating an event</td>
<td>Kubernetes resource</td>
<td>createEvents</td>
</tr>
<tr>
<td>Creating an Ingress</td>
<td>Kubernetes resource</td>
<td>createIngresses</td>
</tr>
<tr>
<td>Creating a job</td>
<td>Kubernetes resource</td>
<td>createJobs</td>
</tr>
<tr>
<td>Creating a namespace</td>
<td>Kubernetes resource</td>
<td>createNamespaces</td>
</tr>
<tr>
<td>Creating a node</td>
<td>Kubernetes resource</td>
<td>createNodes</td>
</tr>
<tr>
<td>Creating a PersistentVolume-Claim</td>
<td>Kubernetes resource</td>
<td>createPersistentvolumeclaims</td>
</tr>
<tr>
<td>Creating a pod</td>
<td>Kubernetes resource</td>
<td>createPods</td>
</tr>
<tr>
<td>Creating a replica set</td>
<td>Kubernetes resource</td>
<td>createReplicasets</td>
</tr>
<tr>
<td>Creating a resource quota</td>
<td>Kubernetes resource</td>
<td>createResourcequotas</td>
</tr>
<tr>
<td>Creating a secret</td>
<td>Kubernetes resource</td>
<td>createSecrets</td>
</tr>
<tr>
<td>Creating a service</td>
<td>Kubernetes resource</td>
<td>createServices</td>
</tr>
<tr>
<td>Creating a StatefulSet</td>
<td>Kubernetes resource</td>
<td>createStatefulsets</td>
</tr>
<tr>
<td>Creating a volume</td>
<td>Kubernetes resource</td>
<td>createVolumes</td>
</tr>
<tr>
<td>Operation</td>
<td>Resource Type</td>
<td>Event Name</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>--------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>Deleting a ConfigMap</td>
<td>Kubernetes resource</td>
<td>deleteConfigmaps</td>
</tr>
<tr>
<td>Deleting a DaemonSet</td>
<td>Kubernetes resource</td>
<td>deleteDaemonsets</td>
</tr>
<tr>
<td>Deleting a Deployment</td>
<td>Kubernetes resource</td>
<td>deleteDeployments</td>
</tr>
<tr>
<td>Deleting an event</td>
<td>Kubernetes resource</td>
<td>deleteEvents</td>
</tr>
<tr>
<td>Deleting an Ingress</td>
<td>Kubernetes resource</td>
<td>deleteIngresses</td>
</tr>
<tr>
<td>Deleting a job</td>
<td>Kubernetes resource</td>
<td>deleteJobs</td>
</tr>
<tr>
<td>Deleting a namespace</td>
<td>Kubernetes resource</td>
<td>deleteNamespaces</td>
</tr>
<tr>
<td>Deleting a node</td>
<td>Kubernetes resource</td>
<td>deleteNodes</td>
</tr>
<tr>
<td>Deleting a Pod</td>
<td>Kubernetes resource</td>
<td>deletePods</td>
</tr>
<tr>
<td>Deleting a replica set</td>
<td>Kubernetes resource</td>
<td>deleteReplicaset</td>
</tr>
<tr>
<td>Deleting a resource quota</td>
<td>Kubernetes resource</td>
<td>deleteResourcequotas</td>
</tr>
<tr>
<td>Deleting a secret</td>
<td>Kubernetes resource</td>
<td>deleteSecrets</td>
</tr>
<tr>
<td>Deleting a service</td>
<td>Kubernetes resource</td>
<td>deleteServices</td>
</tr>
<tr>
<td>Deleting a StatefulSet</td>
<td>Kubernetes resource</td>
<td>deleteStatefulsets</td>
</tr>
<tr>
<td>Deleting volumes</td>
<td>Kubernetes resource</td>
<td>deleteVolumes</td>
</tr>
<tr>
<td>Replacing a specified ConfigMap</td>
<td>Kubernetes resource</td>
<td>updateConfigmaps</td>
</tr>
<tr>
<td>Replacing a specified DaemonSet</td>
<td>Kubernetes resource</td>
<td>updateDaemonsets</td>
</tr>
<tr>
<td>Replacing a specified Deployment</td>
<td>Kubernetes resource</td>
<td>updateDeployments</td>
</tr>
<tr>
<td>Replacing a specified event</td>
<td>Kubernetes resource</td>
<td>updateEvents</td>
</tr>
<tr>
<td>Operation</td>
<td>Resource Type</td>
<td>Event Name</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>------------------------</td>
<td>---------------------------</td>
</tr>
<tr>
<td>Replacing a specified ingress</td>
<td>Kubernetes resource</td>
<td>updateIngresses</td>
</tr>
<tr>
<td>Replacing a specified job</td>
<td>Kubernetes resource</td>
<td>updateJobs</td>
</tr>
<tr>
<td>Replacing a specified namespace</td>
<td>Kubernetes resource</td>
<td>updateNamespaces</td>
</tr>
<tr>
<td>Replacing a specified node</td>
<td>Kubernetes resource</td>
<td>updateNodes</td>
</tr>
<tr>
<td>Replacing a specified PersistentVolume-Claim</td>
<td>Kubernetes resource</td>
<td>updatePersistentvolumeclaims</td>
</tr>
<tr>
<td>Replacing a specified pod</td>
<td>Kubernetes resource</td>
<td>updatePods</td>
</tr>
<tr>
<td>Replacing a specified replica set</td>
<td>Kubernetes resource</td>
<td>updateReplicasets</td>
</tr>
<tr>
<td>Replacing a specified resource quota</td>
<td>Kubernetes resource</td>
<td>updateResourcequotas</td>
</tr>
<tr>
<td>Replacing a specified secret</td>
<td>Kubernetes resource</td>
<td>updateSecrets</td>
</tr>
<tr>
<td>Replacing a specified service</td>
<td>Kubernetes resource</td>
<td>updateServices</td>
</tr>
<tr>
<td>Replacing a specified StatefulSet</td>
<td>Kubernetes resource</td>
<td>updateStatefulsets</td>
</tr>
<tr>
<td>Replacing the specified status</td>
<td>Kubernetes resource</td>
<td>updateStatus</td>
</tr>
<tr>
<td>Uploading a chart</td>
<td>Kubernetes resource</td>
<td>uploadChart</td>
</tr>
<tr>
<td>Updating a component template</td>
<td>Kubernetes resource</td>
<td>updateChart</td>
</tr>
<tr>
<td>Deleting a chart</td>
<td>Kubernetes resource</td>
<td>deleteChart</td>
</tr>
<tr>
<td>Creating a template application</td>
<td>Kubernetes resource</td>
<td>createRelease</td>
</tr>
<tr>
<td>Updating a template application</td>
<td>Kubernetes resource</td>
<td>updateRelease</td>
</tr>
<tr>
<td>Deleting a template application</td>
<td>Kubernetes resource</td>
<td>deleteRelease</td>
</tr>
</tbody>
</table>
19.2 Querying CTS Logs

Scenario

After you enable CTS, the system starts recording operations on CCE resources. Operation records of the last 7 days can be viewed on the CTS management console.

Procedure

**Step 1** Log in to the management console.

**Step 2** Click in the upper left corner and select a region.

**Step 3** Choose Service List from the main menu. Choose Management & Deployment > Cloud Trace Service.

**Step 4** In the navigation pane of the CTS console, choose Cloud Trace Service > Trace List.

**Step 5** On the Trace List page, query operation records based on the search criteria. Currently, the trace list supports trace query based on the combination of the following search criteria:

- **Trace Source, Resource Type, and Search By**
  Select the search criteria from the drop-down lists. Select CCE from the Trace Source drop-down list.
  If you select Trace name from the Search By drop-down list, specify the trace name.
  If you select Resource ID from the Search By drop-down list, select or enter a specific resource ID.
  If you select Resource name from the Search By drop-down list, select or enter a specific resource name.

- **Operator**: Select a specific operator (at user level rather than account level).

- **Trace Status**: Set this parameter to any of the following values: All trace statuses, normal, warning, and incident.

- **Time range**: You can query traces generated during any time range in the last seven days.

**Step 6** Click on the left of a trace to expand its details, as shown below.

![Figure 19-1 Expanding trace details](image)

**Step 7** Click View Trace in the Operation column. The trace details are displayed.
Figure 19-2 Viewing event details

```json
{
  "service_type": "cct",
  "user": {
    "domain": {
      "name": "admin@smile",
      "id": "cc9c0076108042a6e53b7428d756c7f9f"
    },
    "id": "90ed6ed6bdc5e2436eb973935b49bb5ed",
    "name": "wuhongliu"
  },
  "time": "2018/10/11 09:24:56 UTC+00:00",
  "code": 200,
  "resource_type": "clusters-action",
  "source_ip": "20.215.108.72",
  "trace_name": "operateCluster",
  "trace_type": "ConsoleAction",
  "message": "UserName: wuhongliu; Operation : POST /api/v2/project/6244f4651baa4b5db6a54b33b7ac6e77/clusters/3ce",
  "record_time": "2018/10/11 09:24:56 UTC+00:00",
  "trace_id": "766e4e0-ccf4-11e8-3f5f-2b6ed480c9e3",
  "trace_status": "normal"
}

----End
20.1 Software Repository for Container (SWR)

SoftWare Repository for Container (SWR) provides easy, secure, and reliable management over container images throughout their lifecycle, facilitating the deployment of containerized applications. You can push, pull, and manage container images through SWR console, SWR APIs, or community Command Line Interface (CLI).

SWR can work with Cloud Container Engine (CCE) to facilitate image deployment, apart from serving as an image repository.

**Figure 20-1 How SWR works**

![Image showing the process of how SWR works]

**Features**

- **Lifecycle management over container images**
  SWR manages the whole lifecycle of your container images, including push, pull, and deletion.

- **Private image repository and access control**
  Private image repository and fine-grained permission management allow you to grant different access permissions, namely, read, write, and edit, to different users.

- **P2P acceleration of large scale image distribution**
  Acceleration technology developed by Huawei brings faster image pull for CCE clusters during high concurrency.
Automatic deployment update through triggers

Image deployment can be triggered automatically upon image update. Simply set a trigger to the desired image. Every time the image is updated, the application deployed with this image will be automatically updated.

Uploading an Image

For details about how to upload images to SWR, see Uploading an Image Through a Container Engine Client.

Using an Image

When creating a workload on CCE, you can choose pushed images from My Images. This section describes how to choose a pushed image when creating a workload.

Step 1 Log in to the CCE console. In the navigation pane, choose Workloads > Deployments. On the page displayed, click Create Deployment.

Step 2 Set the following parameters, and retain the default settings for other parameters:

- **Workload Name**: Set it to game.
- **Cluster Name**: Select the cluster in which the application will run.
- **Instances**: Set it to 1.

Step 3 Click Next: Add Container.

Click Add Container. On the My Images tab page, select the pushed image and click OK.

Step 4 For details about how to create a workload, see Creating a Deployment or Creating a StatefulSet.

----End

20.2 Application Operations Management (AOM)

AOM is a one-stop and multi-dimensional O&M management platform for cloud applications. It monitors applications and related cloud resources in real time, collects and associates resource metrics, logs, and events to analyze application health status, and provides flexible alarm reporting and data visualization. With AOM, you can detect and locate faults in a timely manner and monitor running status of applications, resources, and services. For details, see AOM Progressive Knowledge.

After creating workloads on CCE, you can operate and maintain the workloads using the AOM service.

Common Operations

- You can create threshold rules for certain resource metrics. For details, see Creating Static Threshold Rules.
- Use the dashboard to learn comprehensive information in real time during routine O&M. You can create and add metrics to the dashboard for real-time monitoring. For details, see Dashboard.
AOM collects and displays container service logs on the AOM console. For details about how to configure a log policy on CCE, see Configuring Container Log Collection Paths.

Perform routine inspection on applications. For details, see Application Monitoring.