

Cloud Container Engine

FAQs

Issue 01
Date 2024-11-08



Copyright © Huawei Cloud Computing Technologies Co., Ltd. 2024. All rights reserved.

No part of this document may be reproduced or transmitted in any form or by any means without prior written consent of Huawei Cloud Computing Technologies Co., Ltd.

Trademarks and Permissions



HUAWEI and other Huawei trademarks are the property of Huawei Technologies Co., Ltd.

All other trademarks and trade names mentioned in this document are the property of their respective holders.

Notice

The purchased products, services and features are stipulated by the contract made between Huawei Cloud and the customer. All or part of the products, services and features described in this document may not be within the purchase scope or the usage scope. Unless otherwise specified in the contract, all statements, information, and recommendations in this document are provided "AS IS" without warranties, guarantees or representations of any kind, either express or implied.

The information in this document is subject to change without notice. Every effort has been made in the preparation of this document to ensure accuracy of the contents, but all statements, information, and recommendations in this document do not constitute a warranty of any kind, express or implied.

Contents

1 Common FAQ	1
2 Billing	2
2.1 How Is CCE Billed?	2
2.2 How Do I Change the Billing Mode of a CCE Cluster from Pay-per-Use to Yearly/Monthly?	3
2.3 Can I Change the Billing Mode of CCE Nodes from Pay-per-Use to Yearly/Monthly?	3
2.4 Can I Delete a Yearly/Monthly-Billed CCE Cluster Directly When It Expires?	4
3 Cluster	5
3.1 Cluster Creation	5
3.1.1 Why Cannot I Create a CCE Cluster?	5
3.1.2 Is Management Scale of a Cluster Related to the Number of Master Nodes?	6
3.1.3 How Do I Update the Root Certificate When Creating a CCE Cluster?	6
3.1.4 Which Resource Quotas Should I Pay Attention To When Using CCE?	7
3.2 Cluster Running	8
3.2.1 How Do I Locate the Fault When a Cluster Is Unavailable?	8
3.2.2 How Do I Retrieve Data After a CCE Cluster Is Deleted?	9
3.3 Cluster Deletion	9
3.3.1 What Can I Do If a Cluster Deletion Fails Due to Residual Resources in the Security Group?	10
3.3.2 How Do I Clear Residual Resources After Deleting a Non-Running Cluster?	11
3.4 Cluster Upgrade	12
3.4.1 What Do I Do If a Cluster Add-On Fails to be Upgraded During the CCE Cluster Upgrade?	12
4 Node	15
4.1 Node Creation	15
4.1.1 How Do I Troubleshoot Problems Occurred When Adding Nodes to a CCE Cluster?	15
4.1.2 How Do I Troubleshoot Problems Occurred When Accepting Nodes into a CCE Cluster?	17
4.1.3 What Should I Do If a Node Fails to Be Accepted Because It Fails to Be Installed?	19
4.2 Node Running	19
4.2.1 What Should I Do If a Cluster Is Available But Some Nodes Are Unavailable?	20
4.2.2 How Do I Log In to a Node Using a Password and Reset the Password?	24
4.2.3 How Do I Collect Logs of Nodes in a CCE Cluster?	24
4.2.4 What Should I Do If the vdb Disk of a Node Is Damaged and the Node Cannot Be Recovered After Reset?	26
4.2.5 What Should I Do If I/O Suspension Occasionally Occurs When SCSI EVS Disks Are Used?	27

4.2.6 How Do I Fix an Abnormal Container or Node Due to No Thin Pool Disk Space?.....	28
4.2.7 How Do I Rectify Failures When the NVIDIA Driver Is Used to Start Containers on GPU Nodes?.....	32
4.3 Specification Change.....	33
4.3.1 How Do I Change the Node Specifications in a CCE Cluster?.....	33
4.3.2 What Should I Do If I Fail to Restart or Create Workloads on a Node After Modifying the Node Specifications?.....	33
4.4 OSs.....	34
4.4.1 What Should I Do If There Is a Service Access Failure After a Backend Service Upgrade or a 1-Second Latency When a Service Accesses a CCE Cluster?.....	34
5 Node Pool.....	38
5.1 What Should I Do If a Node Pool Is Abnormal?.....	38
5.2 What Should I Do If No Node Creation Record Is Displayed When the Node Pool Is Being Expanding?.....	39
5.3 What Should I Do If a Node Pool Scale-Out Fails?.....	40
5.4 How Do I Modify ECS Configurations When an ECS Cannot Be Managed by a Node Pool?.....	42
6 Workload.....	47
6.1 Workload Exception Troubleshooting.....	47
6.1.1 How Can I Find the Fault for an Abnormal Workload?.....	47
6.1.2 What Should I Do If Pod Scheduling Fails?.....	49
6.1.3 What Should I Do If a Pod Fails to Pull the Image?.....	58
6.1.4 What Should I Do If Container Startup Fails?.....	67
6.1.5 What Should I Do If a Pod Fails to Be Evicted?.....	75
6.1.6 What Should I Do If a Storage Volume Cannot Be Mounted or the Mounting Times Out?.....	80
6.1.7 What Should I Do If a Workload Remains in the Creating State?.....	81
6.1.8 What Should I Do If a Pod Remains in the Terminating State?.....	83
6.1.9 What Should I Do If a Workload Is Stopped Caused by Pod Deletion?.....	84
6.1.10 What Should I Do If an Error Occurs When I Deploy a Service on the GPU Node?.....	85
6.1.11 How Can I Locate Faults Using an Exit Code?.....	86
6.2 Container Configuration.....	89
6.2.1 When Is Pre-stop Processing Used?.....	89
6.2.2 How Do I Set an FQDN for Accessing a Specified Container in the Same Namespace?.....	89
6.2.3 What Should I Do If Health Check Probes Occasionally Fail?.....	90
6.2.4 How Do I Set the umask Value for a Container?.....	90
6.2.5 What Is the Retry Mechanism When CCE Fails to Start a Pod?.....	91
6.3 Scheduling Policies.....	91
6.3.1 How Do I Evenly Distribute Multiple Pods to Each Node?.....	91
6.3.2 How Do I Prevent a Container on a Node from Being Evicted?.....	92
6.3.3 Why Are Pods Not Evenly Distributed on Nodes?.....	93
6.3.4 How Do I Evict All Pods on a Node?.....	94
6.3.5 Why Cannot a Pod Be Scheduled to a Node?.....	95
6.4 Others.....	95
6.4.1 What Should I Do If a Cron Job Cannot Be Restarted After Being Stopped for a Period of Time?.....	96

6.4.2 What Is a Headless Service When I Create a StatefulSet?.....	96
6.4.3 What Should I Do If Error Message "Auth is empty" Is Displayed When a Private Image Is Pulled?	97
6.4.4 What Is the Image Pull Policy for Containers in a CCE Cluster?.....	98
6.4.5 What Can I Do If a Layer Is Missing During Image Pull?.....	98
7 Networking.....	100
7.1 Network Exception Troubleshooting.....	100
7.1.1 How Do I Locate a Workload Networking Fault?.....	100
7.1.2 Why Does the Browser Return Error Code 404 When I Access a Deployed Application?.....	102
7.1.3 What Should I Do If a Container Fails to Access the Internet?.....	103
7.1.4 What Should I Do If a Node Fails to Connect to the Internet (Public Network)?.....	104
7.1.5 What Should I Do If Nginx Ingress Access in the Cluster Is Abnormal After the NGINX Ingress Controller Add-on Is Upgraded?.....	104
7.1.6 What Could Cause Access Exceptions After Configuring an HTTPS Certificate for a LoadBalancer Ingress?.....	106
7.2 Network Planning.....	107
7.2.1 What Is the Relationship Between Clusters, VPCs, and Subnets?.....	107
7.2.2 How Can I Configure a Security Group Rule in a Cluster?.....	108
7.3 Security Hardening.....	118
7.3.1 How Do I Prevent Cluster Nodes from Being Exposed to Public Networks?.....	118
7.3.2 How Do I Configure an Access Policy for a Cluster?.....	118
7.3.3 How Do I Change the Security Group of Nodes in a Cluster in Batches?.....	119
7.4 Network Configuration.....	120
7.4.1 How Can Container IP Addresses Survive a Container Restart?.....	120
7.4.2 How Can I Check Whether an ENI Is Used by a Cluster?.....	120
7.4.3 How Can I Delete a Security Group Rule Associated with a Deleted Subnet?.....	121
7.4.4 How Can I Synchronize Certificates When Multiple Ingresses in Different Namespaces Share a Listener?.....	122
7.4.5 How Can I Determine Which Ingress the Listener Settings Have Been Applied To?.....	124
8 Storage.....	129
8.1 How Do I Expand the Storage Capacity of a Container?.....	129
8.2 What Are the Differences Among CCE Storage Classes in Terms of Persistent Storage and Multi-Node Mounting?.....	130
8.3 Can I Create a CCE Node Without Adding a Data Disk to the Node?.....	132
8.4 What Should I Do If the Host Cannot Be Found When Files Need to Be Uploaded to OBS During the Access to the CCE Service from a Public Network?.....	132
8.5 How Can I Achieve Compatibility Between ExtendPathMode and Kubernetes client-go?.....	133
8.6 Can CCE PVCs Detect Underlying Storage Faults?.....	135
8.7 What Should I Do If a Yearly/Monthly EVS Disk Cannot Be Automatically Created?.....	136
9 Namespace.....	138
9.1 What Should I Do If a Namespace Fails to Be Deleted Due to an APIService Object Access Failure?.....	138
10 Chart and Add-on.....	140

10.1 What Should I Do If Installation of an Add-on Fails and "The release name is already exist" Is Displayed?.....	140
10.2 How Do I Configure the Add-on Resource Quotas Based on Cluster Scale?.....	142
10.3 How Can I Clean Up Residual Resources After the NGINX Ingress Controller Add-on in the Unknown State Is Deleted?.....	145
10.4 Why TLS v1.0 and v1.1 Cannot Be Used After the NGINX Ingress Controller Add-on Is Upgraded?.	146
11 API & kubectl FAQs.....	148
11.1 How Can I Access a Cluster API Server?.....	148
11.2 Can the Resources Created Using APIs or kubectl Be Displayed on the CCE Console?.....	148
11.3 How Do I Download kubeconfig for Connecting to a Cluster Using kubectl?.....	149
11.4 How Do I Rectify the Error Reported When Running the kubectl top node Command?.....	149
11.5 Why Is "Error from server (Forbidden)" Displayed When I Use kubectl?.....	150
12 DNS FAQs.....	151
12.1 What Should I Do If Domain Name Resolution Fails in a CCE Cluster?.....	151
12.2 Why Does a Container in a CCE Cluster Fail to Perform DNS Resolution?.....	153
12.3 How Do I Optimize the Configuration If the External Domain Name Resolution Is Slow or Times Out?.....	154
12.4 How Do I Configure a DNS Policy for a Container?.....	155
13 Image Repository FAQs.....	157
13.1 How Do I Upload My Images to CCE?.....	157
14 Permissions.....	158
14.1 Can I Configure Only Namespace Permissions Without Cluster Management Permissions?.....	158
14.2 Can I Use CCE APIs If the Cluster Management Permissions Are Not Configured?.....	158
14.3 Can I Use kubectl If the Cluster Management Permissions Are Not Configured?.....	159

1 Common FAQ

Cluster Management

- [Why Cannot I Create a CCE Cluster?](#)
- [Is Management Scale of a Cluster Related to the Number of Master Nodes?](#)
- [How Do I Locate the Fault When a Cluster Is Unavailable?](#)

Node/Node Pool Management

- [What Should I Do If a Cluster Is Available But Some Nodes Are Unavailable?](#)
- [What Should I Do If I/O Suspension Occasionally Occurs When SCSI EVS Disks Are Used?](#)

Workload Management

- [What Should I Do If Pod Scheduling Fails?](#)
- [What Should I Do If a Pod Fails to Pull the Image?](#)
- [What Should I Do If Container Startup Fails?](#)
- [What Should I Do If a Pod Remains in the Terminating State?](#)
- [What Is the Image Pull Policy for Containers in a CCE Cluster?](#)

Network Management

- [Why Does the Browser Return Error Code 404 When I Access a Deployed Application?](#)
- [What Should I Do If a Node Fails to Connect to the Internet \(Public Network\)?](#)
- [How Do I Optimize the Configuration If the External Domain Name Resolution Is Slow or Times Out?](#)

2 Billing

2.1 How Is CCE Billed?

Billing Modes

There are yearly/monthly and pay-per-use billing modes to meet your requirements. For details, see [Billing Modes](#).

- Yearly/Monthly is a prepaid billing mode. You pay in advance for a subscription term. Before purchasing yearly/monthly resources, ensure that your account has sufficient balance.
- Pay-per-use is a postpaid billing mode. You pay as you go and just pay for what you use.

After purchasing CCE clusters or cluster resources, you can change their billing modes if the current billing mode cannot meet your service requirements. For details, see [Billing Mode Changes](#).

Billing Items

You will be billed for clusters, nodes, and other cloud service resources. For details about the billing factors and formulas for each billed item, see [Billed Items](#).

1. **Clusters:** the cost of resources used by master nodes. It varies with the cluster type (VMs or BMSs and the number of master nodes) and size (the number of worker nodes).

For more details, see [CCE Pricing Details](#).

2. **Other cloud resources:** the cost of IaaS resources in use. Such resources, which are created either manually or automatically during cluster creation, include ECSs, EVS disks, EIPs, bandwidth, and load balancers.

For more pricing details, see [Product Pricing Details](#).

For more information about the billing samples and the billing for each item, see [Billing Examples](#).

2.2 How Do I Change the Billing Mode of a CCE Cluster from Pay-per-Use to Yearly/Monthly?

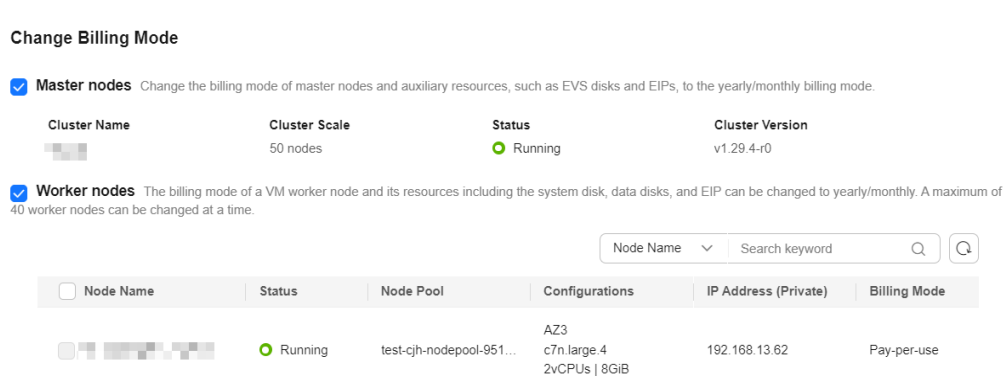
Currently, clusters support **pay-per-use** and **yearly/monthly** billing modes. A pay-per-use cluster can be converted to a yearly/monthly-billed cluster.

Changing the Billing Mode of a Cluster

To change the billing mode of a cluster from pay-per-use to yearly/monthly, perform the following steps:

- Step 1** Log in to the CCE console. In the navigation pane, choose **Clusters**.
- Step 2** Locate the target cluster, click ... to view more operations on the cluster, and choose **Change Billing Mode**.
- Step 3** On the page displayed, select the target cluster. You can also select the nodes whose billing modes you want to change.

Figure 2-1 Changing the billing mode of a cluster to yearly/monthly



- Step 4** Click **OK**. Wait until the order is processed and the payment is complete.
- End

2.3 Can I Change the Billing Mode of CCE Nodes from Pay-per-Use to Yearly/Monthly?

Currently, nodes support **pay-per-use** and **yearly/monthly** billing modes.

Notes and Constraints

- To change the billing mode of a node in a pay-per-use node pool to yearly/monthly, you need to upgrade the cluster to v1.19.16-r40, v1.21.11-r0, v1.23.0-r0, v1.25.4-r0, or later.
- After a node in a pay-per-use node pool is changed to a yearly/monthly node, the node does not support elastic scale-in.

Changing the Billing Mode of a Node

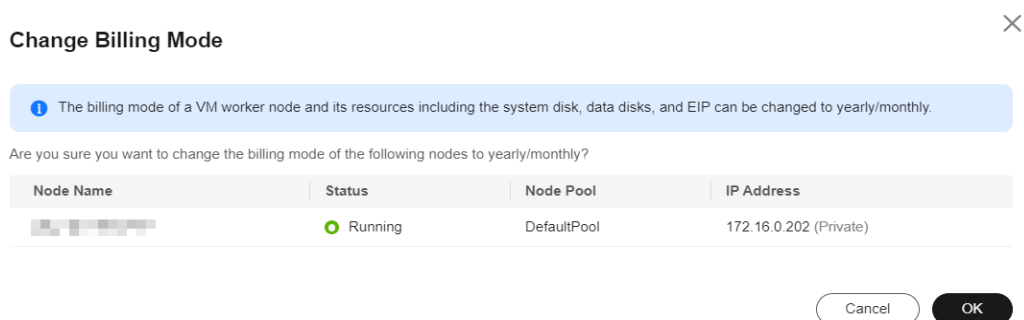
NOTICE

- The billing modes of resources like EVS disks and EIPs used by pay-per-use nodes cannot be changed simultaneously. For details, see [Pay-per-Use to Yearly/Monthly](#).
- To change a pay-per-use node in a node pool to a yearly/monthly one, locate the target node in the node list, choose **More > Forbid node pool scale-in** above the list, and change the billing mode to yearly/monthly.

To change the billing mode of a pay-per-use node to yearly/monthly, perform the following steps:

- Step 1** Log in to the CCE console and click the cluster name to access the cluster console.
- Step 2** In the navigation pane, choose **Nodes**. Click the **Nodes** tab, locate the row containing the target node, and choose **More > Change Billing Mode** in the **Operation** column.

Figure 2-2 Changing the billing mode of a node to yearly/monthly



- Step 3** Click **OK**. Wait until the order is processed and the payment is complete.

----End

2.4 Can I Delete a Yearly/Monthly-Billed CCE Cluster Directly When It Expires?

After a yearly/monthly-billed cluster expires, you can delete the cluster after all data is backed up.

If you do not renew or delete the cluster after it expires, the system will delete the cluster based on the resource expiration time. You are advised to renew the cluster and back up data in a timely manner.

3 Cluster

3.1 Cluster Creation

3.1.1 Why Cannot I Create a CCE Cluster?

Overview

This section describes how to locate and rectify the fault if you fail to create a CCE cluster.

Details

Possible causes:

1. The Network Time Protocol daemon (ntpd) is not installed or fails to be installed, Kubernetes components fail to pass the pre-verification, or the disk partition is incorrect. The current solution is to create a cluster again. For details about how to locate the fault, see [Locating the Failure Cause](#).

Locating the Failure Cause

View the cluster logs to identify the cause and rectify the fault.

Step 1 Log in to the CCE console and click **Operation Records** above the cluster list to view operation records.

Step 2 Click the record of the **Failed** status to view error information.

Figure 3-1 Viewing the operation details

Operation Records ×

All Actions Failed ↻

Cluster Name	Operation Type	Status	Time
^ r30027646-new	Create Cluster	● Failed	May 07, 2022 11:39:40 GMT+08:00

	Project	Start Time	End Time	Status
Create	Create security group rule for cluster communication	May 07, 2022 11:39:41 GMT+08:00	May 07, 2022 11:39:41 GMT+08:00	Completed
	Create security group rule for master node	May 07, 2022 11:39:41 GMT+08:00	May 07, 2022 11:39:41 GMT+08:00	Completed
Group	Create security group rules for worker nodes	May 07, 2022 11:39:41 GMT+08:00	May 07, 2022 11:39:46 GMT+08:00	Completed
	Create master node network	May 07, 2022 11:39:41 GMT+08:00	May 07, 2022 11:39:41 GMT+08:00	Completed
Create	Create control node subnet	May 07, 2022 11:39:41	May 07, 2022 11:39:44	Completed
	Create master node (5 minutes)[1/3]	May 07, 2022 11:39:44 GMT+08:00	--	Failed

Expected HTTP response code [200 201 202 203 204] when accessing [POST https://ecs-internal.cn-north-7.myhuaweicloud.com/v1/0524ea9c1a00d57e2fddc0190fc7dd97/cloudservers], but got 400 instead {"error":{"message":"The volume type[SSD] cannot be used with the specified flavor in the AZ [cn-north-7b].","code":"Ecs.0044"}}

Step 3 Rectify the fault based on the error information and create a cluster again.

----End

3.1.2 Is Management Scale of a Cluster Related to the Number of Master Nodes?

Management scale indicates the maximum number of nodes that can be managed by a cluster. If you select **50 nodes**, the cluster can manage a maximum of 50 nodes.

The number of master nodes varies according to the cluster specification, but is not affected by the management scale.

After the multi-master node mode is enabled, three master nodes will be created. If one of them is faulty, the cluster can still run properly. The services will not be affected.

3.1.3 How Do I Update the Root Certificate When Creating a CCE Cluster?

The root certificate of CCE clusters is the basic certificate for Kubernetes authentication. Both the Kubernetes cluster control plane and the certificate are hosted on Huawei Cloud CCE. CCE will periodically update the certificate. This certificate is not open to users but will not expire.

The X.509 certificate is enabled on Kubernetes clusters by default. CCE will automatically maintain and update the X.509 certificate.

Obtaining a Cluster Certificate

You can obtain a cluster certificate on the CCE console to access Kubernetes. For details, see [Obtaining a Cluster Certificate](#).

3.1.4 Which Resource Quotas Should I Pay Attention To When Using CCE?

CCE restricts **only the number of clusters**. However, when using CCE, you may also be using other cloud services, such as Elastic Cloud Server (ECS), Elastic Volume Service (EVS), Virtual Private Cloud (VPC), Elastic Load Balance (ELB), and SoftWare Repository for Containers (SWR).

What Is Quota?

Quotas can limit the number or amount of resources available to users, such as the maximum number of ECS or EVS disks that can be created.

If the existing resource quota cannot meet your service requirements, you can apply for a higher quota.

How Do I View My Quota?


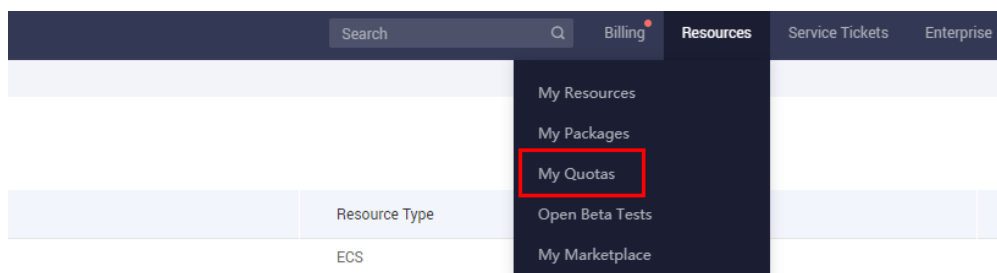
1. Log in to the management console.
2. Click  in the upper left corner to select a region and a project.
3. In the upper right corner of the page, choose **Resources > My Quotas**.
The **Service Quota** page is displayed.

Figure 3-2 My Quotas

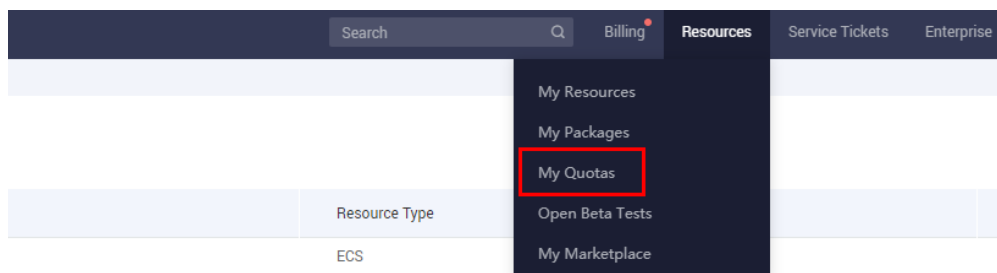


4. On this page, view the total quota and used quota of resources.
If a quota cannot meet your service requirements, click **Increase Quota**.

How Do I Increase My Quota?

1. Log in to the management console.
2. In the upper right corner of the page, choose **Resources > My Quotas**.
The **Service Quota** page is displayed.

Figure 3-3 My Quotas



3. Click **Increase Quota**.
4. On the **Create Service Ticket** page, configure parameters as required and submit a service ticket.

In the **Problem Description** area, enter the required quota and reason for the adjustment.

5. Select **I have read and agree to the Ticket Service Protocol and Privacy Statement**, and click **Submit**.

3.2 Cluster Running

3.2.1 How Do I Locate the Fault When a Cluster Is Unavailable?

If a cluster is unavailable, perform the following operations to locate the fault.

Troubleshooting Process

The issues here are described in order of how likely they are to occur.

Check these causes one by one until you find the cause of the fault.

- [Check Item 1: Whether the Security Group Is Modified](#)
- [Check Item 2: Whether There Are Residual Listeners and Backend Server Groups on the Load Balancer](#)

If the fault persists, contact the customer service to help you locate the fault.

Check Item 1: Whether the Security Group Is Modified

- Step 1** Log in to the management console and choose **Service List > Networking > Virtual Private Cloud**. In the navigation pane, choose **Access Control > Security Groups** to find the security group of the master node in the cluster.

The name of this security group is in the format of *Cluster name-cce-control-ID*.

- Step 2** Click the security group. On the details page displayed, ensure that the security group rules of the master node are correct.

For details, see [How Can I Configure a Security Group Rule in a Cluster?](#)

----End

Check Item 2: Whether There Are Residual Listeners and Backend Server Groups on the Load Balancer

Reproducing the Problem

A cluster exception occurs when a LoadBalancer Service is being created or deleted. After the fault is rectified, the Service is deleted successfully, but there are residual listeners and backend server groups.

- Step 1** Pre-create a CCE cluster. In the cluster, use the official Nginx image to create a workload, preset a load balancer, a Service, and an ingress.
- Step 2** Ensure that the cluster is running properly and the Nginx workload is stable.
- Step 3** Create and delete 10 LoadBalancer Services every 20 seconds.
- Step 4** Verify that an injection exception occurs in the cluster. For example, the etcd is unavailable or the cluster is hibernated.

----End

Possible Causes

There are residual listeners and backend server groups on the load balancer.

Solution

Manually clear residual listeners and backend server groups.

- Step 1** Log in to the management console and choose **Networking > Elastic Load Balance** from the service list.
- Step 2** In the load balancer list, click the name of the target load balancer to go to the details page. On the **Listeners** tab, locate the target listener and delete it.
- Step 3** On the **Backend Server Groups** page, locate the target backend server group and delete it.

----End

3.2.2 How Do I Retrieve Data After a CCE Cluster Is Deleted?

Question

How do I retrieve data after a CCE cluster is deleted?

Answer:

After a cluster is deleted, the workload on the cluster will also be deleted and cannot be restored. Therefore, exercise caution when deleting a cluster.

3.3 Cluster Deletion

3.3.1 What Can I Do If a Cluster Deletion Fails Due to Residual Resources in the Security Group?

When deleting a cluster, CCE obtains the cluster's resources through kube-apiserver of the cluster. If the cluster is unavailable, frozen, or hibernated, the resources may fail to be obtained, and the cluster may not be deleted.

Symptom

The cluster cannot be deleted, and the following error information is displayed:

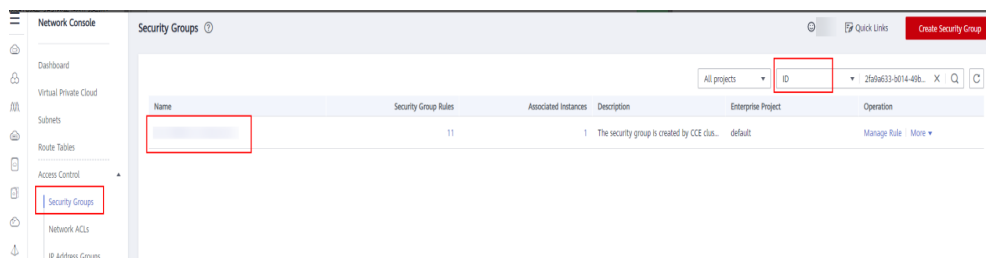
```
Expected HTTP response code [200 202 204 404] when accessing [DELETE https://vpc.***.com/v2.0/security-groups/46311976-7743-4c7c-8249-ccd293bcae91], but got 409 instead
{"code":"VPC.0602","message":{"NeutronError":{"message":"Security Group 46311976-7743-4c7c-8249-ccd293bcae91 in use"},"type":"SecurityGroupInUse"},"detail":{"\"\"}}
```

Possible Causes

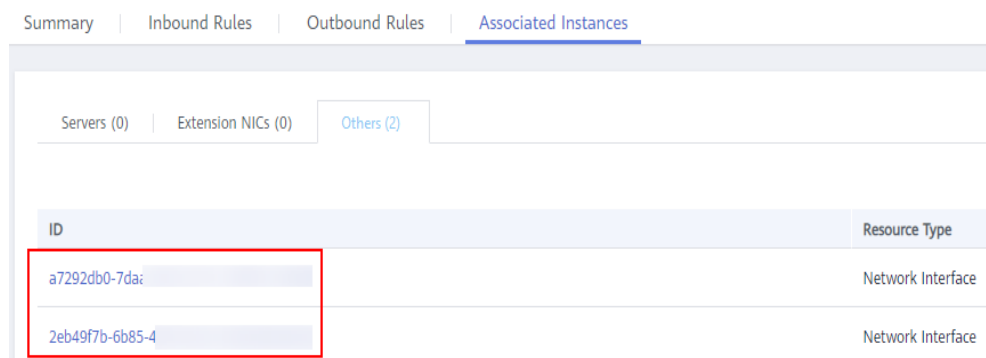
The cluster's security group has undeleted resources, preventing its deletion and causing the creation of the cluster to fail.

Procedure

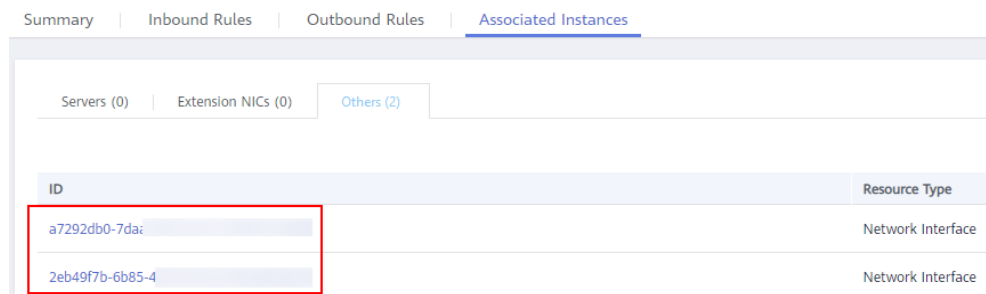
- Step 1** Copy the resource ID in the error information, go to the **Security Groups** page of the VPC console, and obtain security groups by ID.



- Step 2** Click the security group to view its details, and click the **Associated Instances** tab.

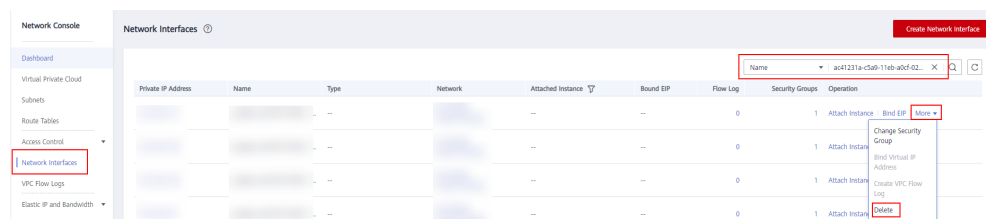


Obtain other resources associated with the security group, such as servers, ENIs, and sub-ENIs. You can delete residual resources. The sub ENIs will be automatically deleted.



Step 3 For a residual ENI, go to the **Network Interfaces** page and delete the ENI obtained in the previous step.

You can search for the ENIs to be deleted by IDs or names.



Step 4 Go to the **Security Groups** page to confirm that the security group is not associated with any instance. Then, go to the CCE console to delete the cluster.

----End

3.3.2 How Do I Clear Residual Resources After Deleting a Non-Running Cluster?

If a cluster is not in the running state (for example, frozen or unavailable), its resources such as PVCs, Services, and Ingresses cannot be obtained. After the cluster is deleted, residual network and storage resources may exist. In this case, manually delete these resources on their respective service console.

Deleting Residual ELB Resources

- Step 1** Log in to the ELB console.
- Step 2** Search for load balancers in the VPC by VPC ID used in the cluster.
- Step 3** View the listener details of a load balancer. If the description contains the cluster ID and Service ID, the listener is created in the cluster.
- Step 4** Delete the residual load balancer-related resources from the cluster based on the preceding information.

----End

Deleting Residual EVS Resources

An EVS disk dynamically created using a PVC is named in the format of **pvc-*{UID}***. The **metadata** field in the API contains the cluster ID. You can use this cluster ID to obtain these EVS disks automatically created in the cluster and delete them as required.

- Step 1** Go to the EVS console.
- Step 2** Search for EVS disks by **pvc-*{UID}*** to get all automatically created EVS disks in the cluster.
- Step 3** Press **F12** to open the developer tools. Check whether the **metadata** field in the **detail** API contains the cluster ID. If yes, the EVS disks are automatically created in this cluster.
- Step 4** Delete the residual EVS disk-related resources from the cluster based on the preceding information.

 **NOTE**

Deleted data cannot be restored. Exercise caution when performing this operation.

----End

Deleting Residual SFS Resources

An SFS file system dynamically created using a PVC is named in the format of **pvc-*{UID}***. The **metadata** field in the API contains the cluster ID. You can use this cluster ID to obtain these SFS file systems automatically created in the cluster and delete them as required.

- Step 1** Log in to the SFS console.
- Step 2** Search for SFS file systems **pvc-*{UID}*** to get all automatically created SFS file systems in the cluster.
- Step 3** Press **F12** to open the developer tools. Check whether the **metadata** field in the **detail** API contains the cluster ID. If yes, the SFS file systems are automatically created in the cluster.
- Step 4** Delete the residual SFS file system-related resources from the cluster based on the preceding information.

 **NOTE**

Deleted data cannot be restored. Exercise caution when performing this operation.

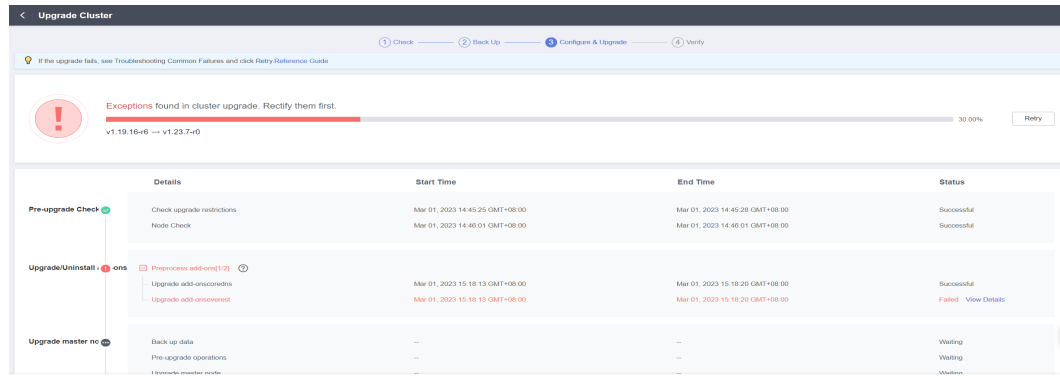
----End

3.4 Cluster Upgrade

3.4.1 What Do I Do If a Cluster Add-On Fails to be Upgraded During the CCE Cluster Upgrade?

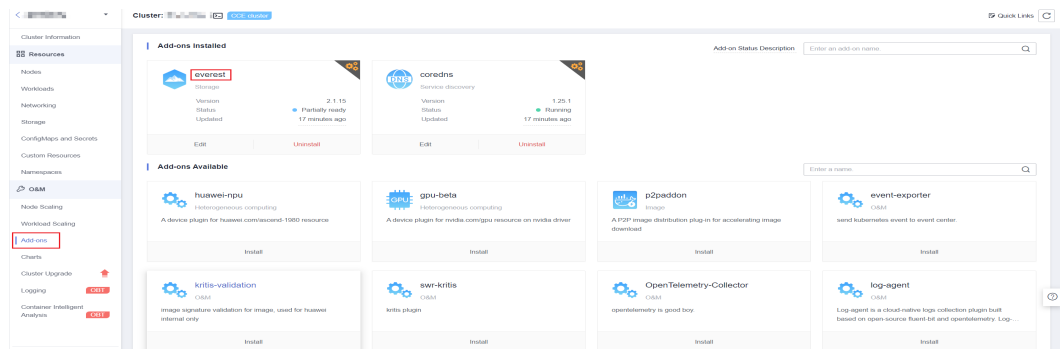
Overview

This section describes how to locate and rectify the fault if you fail to upgrade an add-on during the CCE cluster upgrade.



Procedure

- Step 1** If the add-on fails to be upgraded, try again first. If the retry fails, perform the following steps to rectify the fault.
- Step 2** If a failure message is displayed on the upgrade page, go to the **Add-ons** page to view the add-on status. For an abnormal add-on, click the add-on name to view details.



- Step 3** On the pod details page, click **View Events** in the **Operation** column of the abnormal pod.

Pods

Pod Name	Status	Names...	Pod IP	Node	Rest...	CPU Request/Limit/Usa	Memory Request/Limit/Usa	Created	Operation
everest-csi-driver- Host network	Running	kube-system	192.168.0.100	192.168.0.100	0	0.1 Cores 0.5 Cores 1.00%	300 MiB 300 MiB 26.25%	3 hours ago	Monitor View Events More
everest-csi-driver- Host network	Running	kube-system	192.168.0.230	192.168.0.230	0	0.1 Cores 0.5 Cores 0.80%	300 MiB 300 MiB 26.03%	3 hours ago	Monitor View Events More
everest-csi-control	Running	kube-system	10.0.0.3	192.168.0.230	0	0.25 Cores 0.25 Cores 0.80%	0.59 GiB 1.46 GiB 5.13%	3 hours ago	Monitor View Events More
everest-csi-control	Abnormal	kube-system	10.0.0.131	192.168.0.100	10	0.25 Cores 0.25 Cores 0.40%	0.59 GiB 1.46 GiB 3.92%	3 hours ago	Monitor View Events More

- Step 4** Rectify the fault based on the exception information. For example, delete the pod that is not started or restart it.

Events

X


💡 Event data is stored only for one hour and then automatically cleared.

Start Date – End Date Enter a Kubernetes event name

Kubernetes...	Event ...	Occurr...	Event Name	Kubernetes Event	First Occurred	Last Occurred
kubelet	Alarm	121	Failed to restar...	the failed container exited with ExitCod...	Mar 01, 2023 15:08:2...	Mar 01, 2023 15:33:1...
kubelet	Alarm	74	Failed to restar...	Back-off restarting failed container	Mar 01, 2023 15:08:2...	Mar 01, 2023 15:23:1...
kubelet	Alarm	4	PodsStart failed	Error: failed to start container "everest-...	Mar 01, 2023 15:08:0...	Mar 01, 2023 15:08:5...
kubelet	Normal	5	Image pulled	Container image "100.79.1.215:20202/...	Mar 01, 2023 11:53:5...	Mar 01, 2023 15:08:5...
kubelet	Normal	5	PodsCreated	Created container everest-csi-controller	Mar 01, 2023 11:53:5...	Mar 01, 2023 15:08:5...
kubelet	Normal	4	PodsVolume ...	Successfully mounted volumes for pod ...	Mar 01, 2023 11:53:5...	Mar 01, 2023 15:08:2...

Step 5 After the processing is successful, the add-on status changes to **Running**. Ensure that all add-ons are in the **Running** status.


Add-ons Installed



coredns
Service discovery

Version: 1.25.1
Status: ● **Running**
Updated: 8 days ago

[Edit](#) | [Uninstall](#)



everest
Storage

Version: 2.1.15
Status: ● **Running**
Updated: 8 days ago

[Edit](#) | [Uninstall](#)

Step 6 Go to the cluster upgrade page and click **Retry**.

① Check — ② Back Up — ③ Configure & Upgrade — ④ Verify

⚠️ If the upgrade fails, see [Troubleshooting Common Failures](#) and click [Retry Reference Guide](#).

Exceptions found in cluster upgrade. Rectify them first.

30.00% Retry

v1.19.16-k8s → v1.23.7-0

	Details	Start Time	End Time	Status
Pre-upgrade Check ●	Check upgrade restrictions	Mar 01, 2023 14:45:25 GMT+08:00	Mar 01, 2023 14:45:28 GMT+08:00	Successful
	Node Check	Mar 01, 2023 14:46:01 GMT+08:00	Mar 01, 2023 14:46:01 GMT+08:00	Successful
Upgrade/Uninstall ● ons	■ Preprocess add-ons [1/2] ⊙			
	Upgrade add-ons/coredns	Mar 01, 2023 15:18:13 GMT+08:00	Mar 01, 2023 15:18:20 GMT+08:00	Successful
	Upgrade add-ons/everest	Mar 01, 2023 15:18:13 GMT+08:00	Mar 01, 2023 15:18:20 GMT+08:00	Failed View Details
Upgrade master node nc	Back up data	—	—	Waiting
	Pre-upgrade operations	—	—	Waiting

----End

4 Node

4.1 Node Creation

4.1.1 How Do I Troubleshoot Problems Occurred When Adding Nodes to a CCE Cluster?

Notes

- The node images in the same cluster must be the same. Pay attention to this when creating, adding, or accepting nodes in a cluster.
- If you need to allocate user space from the data disk when creating a node, do not set the data storage path to any key directory. For example, to store data in the **/home** directory, set the directory to **/home/test** instead of **/home**.

NOTE

Do not set **Path inside a node** to the root directory **/**. Otherwise, the mounting fails. Set **Path inside a node** to any of the following:

- **/opt/xxxx** (excluding **/opt/cloud**)
- **/mnt/xxxx** (excluding **/mnt/paas**)
- **/tmp/xxx**
- **/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 it to **/home/paas**, **/var/paas**, **/var/lib**, **/var/script**, **/mnt/paas**, or **/opt/cloud**. Otherwise, the system or node installation will fail.

Check Item 1: Subnet Quota

Symptom

New nodes cannot be added to a CCE cluster, and a message is displayed indicating that the subnet quota is insufficient.

Cause Analysis

Example:

VPC CIDR block: 192.168.66.0/24

Subnet CIDR block: 192.168.66.0/24

In 192.168.66.0/24, all 251 private IP addresses have been used.

Solution

Step 1 Expand the VPC.

Log in to the console and choose **Virtual Private Cloud** from the service list. On the page displayed, locate the row containing the target VPC and click **Edit CIDR Block** in the **Operation** column.

Step 2 Change the subnet mask to **16** and click **OK**.

Step 3 Click the VPC name. On the **Summary** tab page, click the number next to **Subnets** on the right and click **Create Subnet** to create a subnet.

Step 4 Return to the page for adding a node on the CCE console, and select the newly created subnet.

NOTE

1. Adding subnets to the VPC does not affect the use of the existing 192.168.66.0/24 CIDR block.
You can select a new subnet when creating a CCE node. The new subnet has a maximum of 251 private IP addresses. If the number of private IP addresses cannot meet service requirements, you can add more subnets.
2. Subnets in the same VPC can communicate with each other.

----End

Check Item 2: EIP Quota

Symptom

When a node is added, **EIP** is set to **Auto create**. The node cannot be created, and a message indicating that EIPs are insufficient is displayed.

Solution

Two methods are available to solve the problem.

- **Method 1:** Unbind the VMs bound with EIPs and add a node again.
 - a. Log in to the management console.
 - b. Choose **Service List > Compute > Elastic Cloud Server**.
 - c. In the ECS list, locate the target ECS and click its name.
 - d. On the page displayed, click the **EIPs** tab. In the EIP list, locate the row containing the target EIP, click **Unbind**, and click **Yes**.
 - e. Return to the page for adding a node on the CCE console, select **Use existing** for **EIP**, and add the node again.

- **Method 2:** Increase the EIP quota.

Check Item 3: Security Group

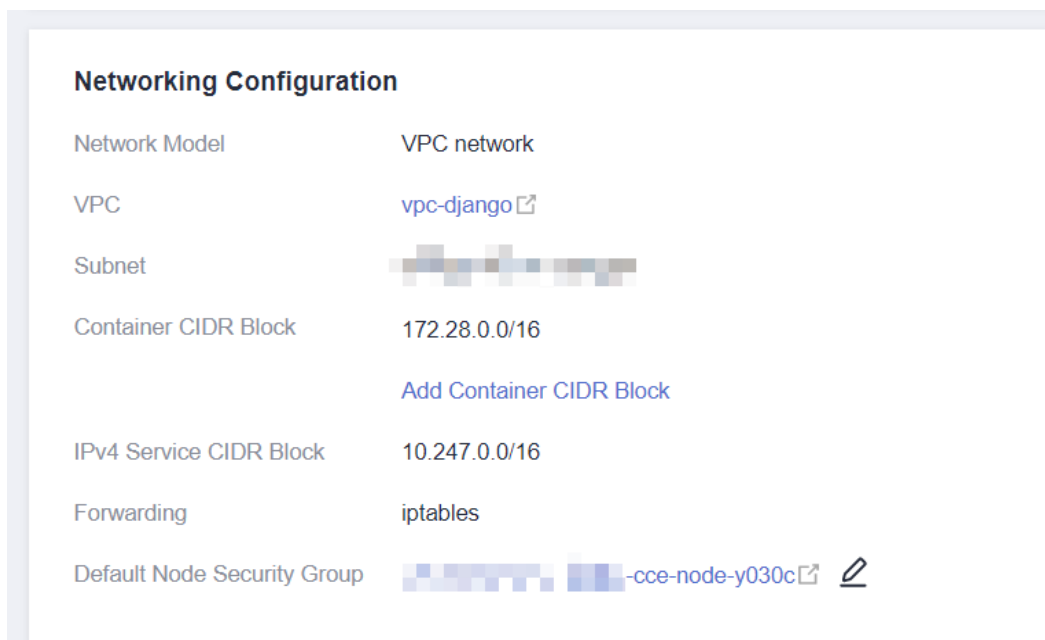
Symptom

A node cannot be added to a CCE cluster.

Solution

You can click the cluster name to view the cluster details. In the **Networking Configuration** area, click the icon next to the value of **Default Node Security Group** to check whether the default security group is deleted and whether the security group rules comply with [How Can I Configure a Security Group Rule in a Cluster?](#)

If your account has multiple clusters and you need to manage network security policies of nodes in a unified manner, you can specify custom security groups.



4.1.2 How Do I Troubleshoot Problems Occurred When Accepting Nodes into a CCE Cluster?

Overview

This section describes how to troubleshoot the problems occurred when you accept or add existing ECSs to a CCE cluster.

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 ECS system and data disks will be formatted while the ECS is being accepted into a cluster. Ensure that data 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

- BMSs and ECSs can be managed.

Prerequisites

The cloud servers to be managed must meet the following requirements:

- The node to be accepted must be in the **Running** state and not used by other clusters. In addition, the node to be accepted does not carry the CCE-Dynamic-Provisioning-Node tag.
- The node to be accepted and the 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.)
- Data disks must be attached to the nodes to be managed. A local disk (disk-intensive disk) or a data disk of at least 20 GiB can be attached to the node, and any data disks already attached cannot be smaller than 10 GiB.
- The node to be accepted has 2-core or higher CPU, 4 GiB or larger memory, and only one NIC.
- If an enterprise project is used, the node to be accepted and the cluster must be in the same enterprise project. Otherwise, resources cannot be identified during management. As a result, the node cannot be accepted.
- Only cloud servers with the same data disk configuration can be accepted in batches for management.
- Nodes in a CCE Turbo cluster must support sub-ENIs or be bound to at least 16 ENIs. For details about the node flavors, see the node flavors that can be selected on the console when you create a node.
- Data disks that have been partitioned will be ignored during node management. Ensure that there is at least one unpartitioned data disk meeting the specifications is attached to the node.

Procedure

View the cluster log information to locate the failure cause and rectify the fault.

Step 1 Log in to the CCE console and click **Operation Records** above the cluster list to view operation records.

Step 2 Click the record of the **Failed** status to view error information.

Step 3 Rectify the fault based on the error information and accept the node into a cluster again.

----End

Common Issues

If a node fails to be managed, a message will be displayed, indicating that the disk partitioning does not work:

```
Install config-prepare failed: exit status 1, output: [ Mon Jul 17 14:26:10 CST 2023 ] start install config-prepare\nNAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT\nsda 8:0 0 40G 0 disk \n└─sda1 8:1 0 40G 0 part \nnsdb 8:16 0 100G 0 disk \n└─sdb1 8:17 0 100G 0 part disk /dev/sda has been partition, will skip this device\nRaw disk /dev/sdb has been partition, will skip this device\nwarning: selector can not match any evs volume
```

To resolve this issue, attach an unpartitioned data disk of 20 GiB or higher to the node. After the node is managed, the unpartitioned data disk is used to store the container engine and kubelet. You can perform operations on the partitioned data disk that does not work as required.

4.1.3 What Should I Do If a Node Fails to Be Accepted Because It Fails to Be Installed?

Symptom

A node fails to be accepted into a cluster.

Possible Causes

Log in to the node and check the `/var/paas/sys/log/baseagent/baseagent.log` installation log. The following error information is displayed:

```
net.core.somaxconn=32768
net.ipv4.tcp_max_syn_backlog=8096
PEERDNS=
failed because of no tenant.conf
10310 10:17:41.075997 6872 baseagent.go:330] install failed
E0310 10:17:41.076179 6872 install.go:181] Install Failed: Install Version(v1.13.7-r0) failed: Exec component plugins/config-prepare Install failed: exit status 1
, output: [ Tue Mar 10 10:17:35 CST 2020 ] start install plugins/config-prepare
net.ipv4.ip_forward = 1
net.ipv4.neigh.default.gc_thresh1 = 2048
net.ipv4.neigh.default.gc_thresh2 = 4096
net.ipv4.neigh.default.gc_thresh3 = 8192
net.ipv4.ip_forward=
```

Check the Logical Volume Manager (LVM) settings of the node. It is found that the LVM logical volume is not created in `/dev/vdb`.

Solution

Run the following command to manually create a logical volume:

```
pvcreate /dev/vdb
vgcreate vgpaas /dev/vdb
```

After the node is reset on the GUI, the node becomes normal.

4.2 Node Running

4.2.1 What Should I Do If a Cluster Is Available But Some Nodes Are Unavailable?

If the cluster status is available but some nodes in the cluster are unavailable, perform the following operations to rectify the fault.

Mechanism for Detecting Node Unavailability

Kubernetes provides the heartbeat mechanism to help you determine node availability. For details about the mechanism and interval, see [Heartbeats](#).

Troubleshooting Process

The issues here are described in order of how likely they are to occur.

Check these causes one by one until you find the cause of the fault.

- [Check Item 1: Whether the Node Is Overloaded](#)
- [Check Item 2: Whether the ECS Is Deleted or Faulty](#)
- [Check Item 3: Whether You Can Log In to the ECS](#)
- [Check Item 4: Whether the Security Group Is Modified](#)
- [Check Item 5: Whether the Security Group Rules Contain the Security Group Policy for the Communication Between the Master Node and the Worker Node](#)
- [Check Item 6: Whether the Disk Is Abnormal](#)
- [Check Item 7: Whether Internal Components Are Normal](#)
- [Check Item 8: Whether the DNS Address Is Correct](#)
- [Check Item 9: Whether the vdb Disk on the Node Is Deleted](#)
- [Check Item 10: Whether the Docker Service Is Normal](#)
- [Check Item 11: Whether a Yearly/Monthly Node Is Being Unsubscribed](#)

Check Item 1: Whether the Node Is Overloaded

Symptom

The node connection in the cluster is abnormal. Multiple nodes report write errors, but services are not affected.

Fault Locating

Step 1 Log in to the CCE console and click the cluster name to access the cluster console. In the navigation pane, choose **Nodes** and click the **Nodes** tab. Locate the row that contains the unavailable node and click **Monitor**.

Step 2 On the top of the displayed page, click **View More** to go to the AOM console and view historical monitoring records.

A too high CPU or memory usage of the node will result in a high network latency or trigger system OOM. Therefore, the node is displayed as unavailable.

----End

Solution

1. Migrate services to reduce the workloads on the node and configure resource limits for the workloads.
2. Clear data on the CCE nodes in the cluster.
3. Limit the CPU and memory quotas of each container.
4. Add more nodes to the cluster.
5. Restart the node on the ECS console.
6. Add nodes to deploy memory-intensive containers separately.
7. Reset the nodes.

After the nodes become available, the workload is restored.

Check Item 2: Whether the ECS Is Deleted or Faulty

Step 1 Check whether the cluster is available.

Log in to the CCE console and check whether the cluster is available.

- If the cluster is unavailable, for example, an error occurs, perform operations described in [How Do I Locate the Fault When a Cluster Is Unavailable?](#)
- If the cluster is running but some nodes in the cluster are unavailable, go to [Step 2](#).

Step 2 Log in to the ECS console and view the ECS status.

- If the ECS status is **Deleted**, go back to the CCE console, delete the corresponding node from the node list of the cluster, and then create another one.
- If the ECS status is **Stopped** or **Frozen**, restore the ECS first. It takes about 3 minutes to restore the ECS.
- If the ECS is **Faulty**, restart the ECS to rectify the fault.
- If the ECS status is **Running**, log in to the ECS to locate the fault according to [Check Item 7: Whether Internal Components Are Normal](#).

----End

Check Item 3: Whether You Can Log In to the ECS

Step 1 Log in to the ECS console.

Step 2 Check whether the node name displayed on the page is the same as that on the VM and whether the password or key can be used to log in to the node.

If the node names are inconsistent and the password and key cannot be used to log in to the node, Cloud-Init problems occurred when an ECS was created. In this case, restart the node and submit a service ticket to the ECS personnel to locate the root cause.

----End

Check Item 4: Whether the Security Group Is Modified

Log in to the VPC console. In the navigation pane, choose **Access Control** > **Security Groups** and locate the security group of the cluster master node.

The name of this security group is in the format of *Cluster name-cce-control-ID*. You can search for the security group by cluster name and *-cce-control-*.

Check whether the security group rules have been modified. For details about security groups, see [How Can I Configure a Security Group Rule in a Cluster?](#)

Check Item 5: Whether the Security Group Rules Contain the Security Group Policy for the Communication Between the Master Node and the Worker Node

Check whether such a security group policy exists.

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.)

For details about security groups, see [How Can I Configure a Security Group Rule in a Cluster?](#)

Check Item 6: Whether the Disk Is Abnormal

A 100-GiB data disk dedicated for Docker is attached to the new node. If the data disk is uninstalled or damaged, the Docker service becomes abnormal and the node becomes unavailable.

Click the node name to check whether the data disk mounted to the node is uninstalled. If the disk is uninstalled, mount a data disk to the node again and restart the node. Then the node can be recovered.

Check Item 7: Whether Internal Components Are Normal

Step 1 Log in to the ECS where the unavailable node is located.

Step 2 Run the following command to check whether the PaaS components are normal:

systemctl status kubelet

If the command is successfully executed, the status of each component is displayed as **active**, as shown in the following figure.

```
root@bms-cce-00406059-11044:~# systemctl status kubelet
kubelet.service - Cloud Container Engine Kubelet Service
Loaded: loaded (/usr/lib/systemd/system/kubelet.service; enabled; vendor preset: disabled)
Active: active (running) since Mon 2019-08-05 14:38:22 CST; 3 days ago
Main PID: 17029 (sudo)
Memory: 139.6M
CGroup: /system.slice/system-hostos.slice/kubelet.service
├─17029 sudo /var/paas/kubernetes/kubelet/srvkubelet start
├─17030 /bin/sh /var/paas/kubernetes/kubelet/srvkubelet start
└─17422 /usr/local/bin/kubelet --bootstrap-kubeconfig=/var/paas/kubernetes/kubelet/boot.conf --cert-dir=/var/paas/kubernetes/kubelet/pki --rotate-certificates=true ...
Aug 05 14:38:22 systemd[1]: Started Cloud Container Engine Kubelet Service.
Aug 05 14:38:22 systemd[1]: Starting Cloud Container Engine Kubelet Service.
Aug 05 14:38:22 sudo[17029]:   pass : TTY=unknown ; PWD=/ ; USER=root ; COMMAND=/var/paas/kubernetes/kubelet/srvkubelet start
Aug 05 14:38:22 sudo[17051]:   pass : TTY=unknown ; PWD=/ ; USER=root ; COMMAND=/bin/sh -c cat > /etc/resolv.conf <<EOF
nameserver 100.79.1.250
options timeout:2 attempts:3 single-request-reopen...
Aug 05 14:38:20 bms-cce-00406059-11044 sh[17029]: 5 Aug 14:38:20 ntpdate[17054]: adjust time server 100.79.0.250 offset 0.014749 sec
hint: Some lines were ellipsized, use -l to show in full.
```

If the component status is not **active**, run the following commands (using the faulty component **canal** as an example):

Run **systemctl restart canal** to restart the component.

After restarting the component, run **systemctl status canal** to check the status.

Step 3 If the restart command fails to be run, run the following command to check the running status of the monitrc process:

```
ps -ef | grep monitrc
```

If the monitrc process exists, run the following command to kill this process. The monitrc process will be automatically restarted after it is killed.

```
kill -s 9 `ps -ef | grep monitrc | grep -v grep | awk '{print $2}'`
```

----End

Check Item 8: Whether the DNS Address Is Correct

Step 1 After logging in to the node, check whether any domain name resolution failure is recorded in the `/var/log/cloud-init-output.log` file.

```
cat /var/log/cloud-init-output.log | grep resolv
```

If the command output contains the following information, the domain name cannot be resolved:

```
Could not resolve host: Unknown error
```

Step 2 On the node, ping the domain name that cannot be resolved in the previous step to check whether the domain name can be resolved on the node.

- If not, the DNS cannot resolve the IP address. Check whether the DNS address in the `/etc/resolv.conf` file is the same as that configured on the VPC subnet. In most cases, the DNS address in the file is incorrectly configured. As a result, the domain name cannot be resolved. Correct the DNS configuration of the VPC subnet and reset the node.
- If yes, the DNS address configuration is correct. Check whether there are other faults.

----End

Check Item 9: Whether the vdb Disk on the Node Is Deleted

If the vdb disk on a node is deleted, you can refer to [this topic](#) to restore the node.

Check Item 10: Whether the Docker Service Is Normal

Step 1 Run the following command to check whether the Docker service is running:

```
systemctl status docker
```

```
● docker.service - Docker Application Container Engine
   Loaded: loaded (/usr/lib/systemd/system/docker.service; enabled; vendor preset: disabled)
   Active: active (running) since Wed 2021-02-03 16:07:02 CST; 1 day 23h ago
     Docs: https://docs.docker.com
   Main PID: 3673 (dockerd)
    Tasks: 46 (limit: 24004)
   Memory: 491.2M
   CGroup: /system.slice/docker.service
           └─3673 /usr/bin/dockerd --live-restore --log-opt max-size=50m --log-opt max-file=20 --log-driver=json-fil
             └─3680 containerd --config /var/run/docker/containerd/containerd.toml --log-level info
               └─5961 containerd-shim -namespace moby -workdir /var/lib/docker/containerd/daemon/io.containerd.runtime.v
                 └─6811 containerd-shim -namespace moby -workdir /var/lib/docker/containerd/daemon/io.containerd.runtime.v
Warning: Journal has been rotated since unit was started. Log output is incomplete or unavailable.
```

If the command fails or the Docker service status is not active, locate the cause or contact technical support if necessary.

Step 2 Run the following command to check the number of containers on the node:

```
docker ps -a | wc -l
```

If the command is suspended, the command execution takes a long time, or there are more than 1000 abnormal containers, check whether workloads are repeatedly created and deleted. If a large number of containers are frequently created and deleted, there may be a large number of abnormal containers, and these containers cannot be cleared in a timely manner.

In this case, stop repeated creation and deletion of the workload or use more nodes to share the workload. Generally, the nodes will be restored after a period of time. If necessary, run the **docker rm** *{container_id}* command to manually clear abnormal containers.

----End

Check Item 11: Whether a Yearly/Monthly Node Is Being Unsubscribed

Once a node is unsubscribed, it will take some time to process the order, rendering the node unavailable during this period. Typically, the node is expected to be automatically cleared within 5 to 10 minutes.

4.2.2 How Do I Log In to a Node Using a Password and Reset the Password?

Context

When creating a node on CCE, you selected a key pair or specified a password for login. If you forget your key pair or password, you can log in to the ECS console to reset the password of the node. After the password is reset, you can log in to the node using the password.

Procedure

Step 1 Log in to the ECS console.

Step 2 In the ECS list, select the cloud server type of the node. In the same row as the node, choose **More > Stop**.

Step 3 After the node is stopped, choose **More > Reset Password**, and follow on-screen prompts to reset the password.

Step 4 After the password is reset, choose **More > Start**, and click **Remote Login** to log in to the node using the password.

----End

4.2.3 How Do I Collect Logs of Nodes in a CCE Cluster?

The following tables list log files of CCE nodes.

Table 4-1 Node logs

Name	Path
kubelet log	<ul style="list-style-type: none"> For clusters of v1.21 or later: <code>/var/log/cce/kubernetes/kubelet.log</code> For clusters of v1.19 or earlier: <code>/var/paas/sys/log/kubernetes/kubelet.log</code>
kube-proxy log	<ul style="list-style-type: none"> For clusters of v1.21 or later: <code>/var/log/cce/kubernetes/kube-proxy.log</code> For clusters of v1.19 or earlier: <code>/var/paas/sys/log/kubernetes/kube-proxy.log</code>
yangtse log (networking)	<ul style="list-style-type: none"> For clusters of v1.21 or later: <code>/var/log/cce/yangtse</code> For clusters of v1.19 or earlier: <code>/var/paas/sys/log/yangtse</code>
canal log	<ul style="list-style-type: none"> For clusters of v1.21 or later: <code>/var/log/cce/canal</code> For clusters of v1.19 or earlier: <code>/var/paas/sys/log/canal</code>
System logs	<code>/var/log/messages</code>
Container engine Logs	<ul style="list-style-type: none"> For Docker nodes: <code>/var/lib/docker</code> For containerd nodes: <code>/var/log/cce/containerd</code>

Table 4-2 Add-on logs

Name	Path
everest log	<ul style="list-style-type: none"> For v2.1.41 or later: <ul style="list-style-type: none"> everest-csi-driver: <code>/var/log/cce/kubernetes</code> everest-csi-controller: <code>/var/paas/sys/log/kubernetes</code> For version earlier than v2.1.41: <ul style="list-style-type: none"> everest-csi-driver: <code>/var/log/cce/everest-csi-driver</code> everest-csi-controller: <code>/var/paas/sys/log/everest-csi-controller</code>
npd log	<ul style="list-style-type: none"> For v1.18.16 or later: <code>/var/paas/sys/log/kubernetes</code> For versions earlier than v1.18.16: <code>/var/paas/sys/log/cceaddon-npd</code>
cce-hpa-controller log	<ul style="list-style-type: none"> For v1.3.12 or later: <code>/var/paas/sys/log/kubernetes</code> For versions earlier than v1.3.12: <code>/var/paas/sys/log/ccehpa-controller</code>

4.2.4 What Should I Do If the vdb Disk of a Node Is Damaged and the Node Cannot Be Recovered After Reset?

Symptom

The vdb disk of a node is damaged and the node cannot be recovered after reset.

Error Scenarios

- On a normal node, delete the LV and VG. The node is unavailable.
- Reset an abnormal node, and a syntax error is reported. The node is unavailable.

The following figure shows the details.

```
vgcreate VG_new PV ...
create volume group error
, skip pause's work in case of failed dependency docker, skip fuxi's work in case of failed dependency docker, sk
work in case of failed dependency kubelet, skip kube-proxy's work in case of failed dependency config-prepare, sk
ork in case of failed dependency config-prepare, skip canal-agent's work in case of failed dependency fuxi, skip c
work in case of failed dependency config-prepare, skip docker's work in case of failed dependency config-prepare,
s work in case of failed dependency config-prepare]
18525 17:22:55.835685 7116 install.go:361 install failed
Install Failed: [Install config-prepare failed: exit status 1, output: [ Mon May 25 17:22:53 CST 2020 ] start inst
pare
success download the file
success download the file
success download the file
success download the file
success download the file
success download the file
success download the file
success download the file
Checking device: /dev/vda
Raw disk /dev/vda has been partition, will skip this device
Checking device: /dev/vdb
Detected paas disk: /dev/vdb
Use to config lv(eg. docker(direct-lvm),kubelet,user)
No command with matching syntax recognised. Run 'vgcreate --help' for more information.
Correct command syntax is:
vgcreate VG_new PV ...

create volume group error
, skip pause's work in case of failed dependency docker, skip fuxi's work in case of failed dependency docker, sk
work in case of failed dependency kubelet, skip kube-proxy's work in case of failed dependency config-prepare, sk
ork in case of failed dependency config-prepare, skip canal-agent's work in case of failed dependency fuxi, skip c
work in case of failed dependency config-prepare, skip docker's work in case of failed dependency config-prepare,
s work in case of failed dependency config-prepare]
```

Fault Locating

If the volume group (VG) on the node is deleted or damaged and cannot be identified, you need to manually restore the VG first to prevent your data disks from being formatted by mistake during the reset.

Solution

Step 1 Log in to the node.

Step 2 Create a PV and a VG again. In this example, the following error message is displayed:

```
root@host1:~# pvcreate /dev/vdb
Device /dev/vdb excluded by a filter
```

This is because the added disk is created on another VM and has a partition table. The current VM cannot identify the partition table of the disk. You need to run the **parted** commands for three times to re-create the partition table.

```
root@host1:~# parted /dev/vdb
GNU Parted 3.2
Using /dev/vdb
Welcome to GNU Parted! Type 'help' to view a list of commands.
```



```
(parted) mklabel msdos
Warning: The existing disk label on /dev/vdb will be destroyed and all data on this disk will be lost. Do you
want to continue?
Yes/No? yes
(parted) quit
Information: You may need to update /etc/fstab.
```

Run **pvcreate** again. When the system asks you whether to erase the DOS signature, enter **y**. The disk is created as a PV.

```
root@host1:~# pvcreate /dev/vdb
WARNING: dos signature detected on /dev/vdb at offset 510. Wipe it? [y/n]: y
Wiping dos signature on /dev/vdb.
Physical volume "/dev/vdb" successfully created
```

Step 3 Create a VG.

Check the Docker disks of the node. If the disks are **/dev/vdb** and **/dev/vdc**, run the following command:

```
root@host1:~# vgcreate vgpaas /dev/vdb /dev/vdc
```

If there is only the **/dev/vdb** disk, run the following command:

```
root@host1:~# vgcreate vgpaas /dev/vdb
```

After the creation is complete, reset the node.

----End

4.2.5 What Should I Do If I/O Suspension Occasionally Occurs When SCSI EVS Disks Are Used?

Symptom

When SCSI EVS disks are used and containers are created and deleted on a CentOS node, the disks are frequently mounted and unmounted. The read/write rate of the system disk may instantaneously surge. As a result, the system is suspended, affecting the normal node running.

When this problem occurs, the following information is displayed in the dmesg log:

```
Attached SCSI disk
task jdb2/xxx blocked for more than 120 seconds.
```

Example:

```
1128163.173120] sd 2:0:0:0: [sda] write protect is off
1128163.173457] sd 2:0:0:0: [sda] Mode Sense: 69 00 00 08
1128163.173573] sd 2:0:0:0: [sda] Write cache: disabled, read cache: enabled, doesn't support DPO or FUA
1128163.176426] sd 2:0:0:0: [sda] Attached SCSI disk
1128350.437941] INFO: task jbd2/dm-1-8:1604 blocked for more than 120 seconds.
1128350.438267] "echo 0 > /proc/sys/kernel/hung_task_timeout_secs" disables this message.
1128350.438564] jbd2/dm-1-8 D ffff9ede7f8420e0 0 1604 2 0x00000000
1128350.438829] Call Trace:
1128350.439120] [<ffffffffffa5a585>] ? blk_mq_dispatch_rq_list+0x325/0x620
1128350.439394] [<ffffffffffaaf7f229>] schedule+0x29/0x70
```

Possible Causes

After a PCI device is hot added to BUS 0, the Linux OS kernel will traverse all the PCI bridges mounted to BUS 0 for multiple times, and these PCI bridges cannot work properly during this period. During this period, if the PCI bridge used by the device is updated, due to a kernel defect, the device considers that the PCI bridge

is abnormal, and the device enters a fault mode and cannot work normally. If the front end is writing data into the PCI configuration space for the back end to process disk I/Os, the write operation may be deleted. As a result, the back end cannot receive notifications to process new requests on the I/O ring. Finally, the front-end I/O suspension occurs.

Impact

CentOS Linux kernels of versions earlier than 3.10.0-1127.el7 are affected.

Solution

Upgrade the kernel to a later version **by resetting the node**.

4.2.6 How Do I Fix an Abnormal Container or Node Due to No Thin Pool Disk Space?

Problem Description

When the disk space of a thin pool on a node is about to be used up, the following exceptions occasionally occur:

Files or directories fail to be created in the container, the file system in the container is read-only, the node is tainted disk-pressure, or the node is unavailable.

You can run the **docker info** command on the node to view the used and remaining thin pool space to locate the fault. The following figure is an example.

```
Storage Driver: devicemapper
Pool Name: vgpaas-thinpool
Pool Blocksize: 524.3kB
Base Device Size: 10.74GB
Backing Filesystem: ext4
Udev Sync Supported: true
Data Space Used: 7.794GB
Data Space Total: 71.94GB
Data Space Available: 64.15GB
Metadata Space Used: 3.076MB
Metadata Space Total: 3.221GB
Metadata Space Available: 3.218GB
Thin Pool Minimum Free Space: 7.194GB
Deferred Removal Enabled: true
Deferred Deletion Enabled: true
Deferred Deleted Device Count: 0
Library Version: 1.02.146-BHEL7 (2018-01-22)
```

Possible Cause

When Docker device mapper is used, although you can configure the **basesize** parameter to 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 only after a while. If **the number of service 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.

Solution

When the thin pool space of a node is used up, some services can be migrated to other nodes to quickly recover services. But you are advised to use the following solutions to resolve the root cause:

Solution 1:

Properly plan the service distribution and data plane disk space to avoid the scenario where **the number of service containers multiplied by basesize** is greater than the thin pool size of the node. To expand the thin pool size, perform the following steps:

Step 1 Expand the capacity of a data disk on the EVS console.

Only the storage capacity of the EVS disk is expanded. You also need to perform the following steps to expand the capacity of the logical volume and file system.

Step 2 Log in to the CCE console and click the cluster. In the navigation pane, choose **Nodes**. Click **More > Sync Server Data** in the row containing the target node.

Step 3 Log in to the target node.

Step 4 Run the **lsblk** command to check the block device information of the node.

A data disk is divided depending on the container storage **Rootfs**:

Overlayfs: No independent thin pool is allocated. Image data is stored in **dockersys**.

1. Check the disk and partition sizes of the device.

```
# lsblk
NAME        MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda          8:0  0  50G  0 disk
└─sda1       8:1  0  50G  0 part /
sdb          8:16  0 150G  0 disk # The data disk has been expanded to 150 GiB, but 50 GiB
space is not allocated.
└─vgpaas-dockersys 253:0  0  90G  0 lvm  /var/lib/containerd
└─vgpaas-kubernetes 253:1  0  10G  0 lvm  /mnt/paas/kubernetes/kubelet
```

2. Expand the disk capacity.

Add the new disk capacity to the **dockersys** logical volume used by the container engine.

a. Expand the PV capacity so that LVM can identify the new EVS capacity. **/dev/sdb** specifies the physical volume where dockersys is located.

```
pvresize /dev/sdb
```

Information similar to the following is displayed:

```
Physical volume "/dev/sdb" changed
1 physical volume(s) resized or updated / 0 physical volume(s) not resized
```

- b. Expand 100% of the free capacity to the logical volume. *vgpaas/dockersys* specifies the logical volume used by the container engine.
`lvextend -l+100%FREE -n vgpaas/dockersys`

Information similar to the following is displayed:

```
Size of logical volume vgpaas/dockersys changed from <90.00 GiB (23039 extents) to 140.00 GiB (35840 extents).
Logical volume vgpaas/dockersys successfully resized.
```

- c. Adjust the size of the file system. */dev/vgpaas/dockersys* specifies the file system path of the container engine.
`resize2fs /dev/vgpaas/dockersys`

Information similar to the following is displayed:

```
Filesystem at /dev/vgpaas/dockersys is mounted on /var/lib/containerd; on-line resizing required
old_desc_blocks = 12, new_desc_blocks = 18
The filesystem on /dev/vgpaas/dockersys is now 36700160 blocks long.
```

3. Check whether the capacity is expanded.

```
# lsblk
NAME                MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda                  8:0  0  50G  0 disk
└─sda1                8:1  0  50G  0 part /
sdb                  8:16  0 150G  0 disk
├─vgpaas-dockersys 253:0  0 140G  0 lvm  /var/lib/containerd
└─vgpaas-kubernetes 253:1  0  10G  0 lvm  /mnt/paas/kubernetes/kubelet
```

Devicemapper: A thin pool is allocated to store image data.

1. Check the disk and partition sizes of the device.

```
# lsblk
NAME                MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
vda                  8:0  0  50G  0 disk
└─vda1                8:1  0  50G  0 part /
vdb                  8:16  0 200G  0 disk
├─vgpaas-dockersys 253:0  0  18G  0 lvm  /var/lib/docker
├─vgpaas-thinpool_tmeta 253:1  0   3G  0 lvm
├─vgpaas-thinpool    253:3  0  67G  0 lvm          # Space used by thinpool
├─...
├─vgpaas-thinpool_tdata 253:2  0  67G  0 lvm
├─vgpaas-thinpool    253:3  0  67G  0 lvm
├─...
└─vgpaas-kubernetes 253:4  0  10G  0 lvm  /mnt/paas/kubernetes/kubelet
```

2. Expand the disk capacity.

Option 1: Add the new disk capacity to the thin pool disk.

- a. Expand the PV capacity so that LVM can identify the new EVS capacity. */dev/vdb* specifies the physical volume where thinpool is located.
`pvresize /dev/vdb`

Information similar to the following is displayed:

```
Physical volume "/dev/vdb" changed
1 physical volume(s) resized or updated / 0 physical volume(s) not resized
```

- b. Expand 100% of the free capacity to the logical volume. *vgpaas/thinpool* specifies the logical volume used by the container engine.
`lvextend -l+100%FREE -n vgpaas/thinpool`

Information similar to the following is displayed:

```
Size of logical volume vgpaas/thinpool changed from <67.00 GiB (23039 extents) to <167.00 GiB (48639 extents).
Logical volume vgpaas/thinpool successfully resized.
```

- c. Do not need to adjust the size of the file system, because the thin pool is not mounted to any devices.

- d. Check whether the capacity is expanded. Run the **lsblk** command to check the disk and partition sizes of the device. If the new disk capacity has been added to the thin pool, the capacity is expanded.

```
# lsblk
NAME                                MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
vda                                  8:0  0  50G  0 disk
├─vda1                               8:1  0  50G  0 part /
└─vdb                                 8:16 0 200G  0 disk
   └─vgpaas-dockersys                 253:0  0  18G  0 lvm  /var/lib/docker
      └─vgpaas-thinpool_tmeta          253:1  0   3G  0 lvm
         └─vgpaas-thinpool             253:3  0 167G  0 lvm      # Thin pool space after
            capacity expansion
               ...
            └─vgpaas-thinpool_tdata     253:2  0  67G  0 lvm
               └─vgpaas-thinpool       253:3  0  67G  0 lvm
                  ...
            └─vgpaas-kubernetes         253:4  0  10G  0 lvm  /mnt/paas/kubernetes/kubelet
```

Option 2: Add the new disk capacity to the **dockersys** disk.

- a. Expand the PV capacity so that LVM can identify the new EVS capacity. */dev/vdb* specifies the physical volume where dockersys is located.

```
pvresize /dev/vdb
```

Information similar to the following is displayed:

```
Physical volume "/dev/vdb" changed
1 physical volume(s) resized or updated / 0 physical volume(s) not resized
```

- b. Expand 100% of the free capacity to the logical volume. *vgpaas/dockersys* specifies the logical volume used by the container engine.

```
lvextend -l+100%FREE -n vgpaas/dockersys
```

Information similar to the following is displayed:

```
Size of logical volume vgpaas/dockersys changed from <18.00 GiB (4607 extents) to <118.00 GiB (30208 extents).
Logical volume vgpaas/dockersys successfully resized.
```

- c. Adjust the size of the file system. */dev/vgpaas/dockersys* specifies the file system path of the container engine.

```
resize2fs /dev/vgpaas/dockersys
```

Information similar to the following is displayed:

```
Filesystem at /dev/vgpaas/dockersys is mounted on /var/lib/docker; on-line resizing required
old_desc_blocks = 3, new_desc_blocks = 15
The filesystem on /dev/vgpaas/dockersys is now 30932992 blocks long.
```

- d. Check whether the capacity is expanded. Run the **lsblk** command to check the disk and partition sizes of the device. If the new disk capacity has been added to the dockersys, the capacity is expanded.

```
# lsblk
NAME                                MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
vda                                  8:0  0  50G  0 disk
├─vda1                               8:1  0  50G  0 part /
└─vdb                                 8:16 0 200G  0 disk
   └─vgpaas-dockersys                 253:0  0 118G  0 lvm  /var/lib/docker  # dockersys after
      capacity expansion
         └─vgpaas-thinpool_tmeta          253:1  0   3G  0 lvm
            └─vgpaas-thinpool             253:3  0  67G  0 lvm
               ...
         └─vgpaas-thinpool_tdata          253:2  0  67G  0 lvm
            └─vgpaas-thinpool             253:3  0  67G  0 lvm
               ...
         └─vgpaas-kubernetes             253:4  0  10G  0 lvm  /mnt/paas/kubernetes/kubelet
```

----End

Solution 2:

Create and delete files in service containers in the local storage (such as emptyDir and hostPath) or cloud storage directory mounted to the container. Such files do not occupy the thin pool space.

Solution 3:

If the OS uses OverlayFS, services can be deployed on such nodes to prevent the problem that the disk space occupied by files created or deleted in the container is not released immediately.

4.2.7 How Do I Rectify Failures When the NVIDIA Driver Is Used to Start Containers on GPU Nodes?

Did a Resource Scheduling Failure Event Occur on a Cluster Node?

Symptom

A node is running properly and has GPU resources. However, the following error information is displayed:

```
0/9 nodes are available: 9 insufficient nvidia.com/gpu
```

Analysis

1. Check whether the node is attached with NVIDIA label.

```
unknown flag: --show-labels
root@chengyindu-test-98835 ~]# kubectl get node --show-labels
NAME          STATUS    ROLES    AGE   VERSION   LABELS
172.16.0.180  Ready    <none>   6h26m v1.13.10-r1-CCE2.0.ZB.B001  accelerator=nvidia-p100, beta.kubernetes.io/arch=amd64, beta.kubernetes.io/os=linux, failure-domain.beta.kubernetes.io/is-baremetal=false, failure-domain.beta.kubernetes.io/region=cn-east-2, failure-domain.beta.kubernetes.io/zone=cn-east-2b, kubernetes.io/availablezone=cn-east-2b, kubernetes.io/eni-quota=12, kubernetes.io/hostname=172.16.0.180, node.kubernetes.io/subnetid=4883a3c2-f09f-412d-bd3a-5a2092c5033a, os.architecture=amd64, os.name=EulerOS_2.0_SP5, os.version=3.10.0-862.14.1.Z.h249.eulerosv2r7.x86_64
root@chengyindu-test-98835 ~]#
```

2. Check whether the NVIDIA driver is running properly.

Log in to the node where the add-on is running and view the driver installation log in the following path:

```
/opt/cloud/cce/nvidia/nvidia_installer.log
```

View standard output logs of the NVIDIA container.

Filter the container ID by running the following command:

```
docker ps -a | grep nvidia
```

View logs by running the following command:

```
docker logs Container ID
```

What Should I Do If the NVIDIA Version Reported by a Service and the CUDA Version Do Not Match?

Run the following command to check the CUDA version in the container:

```
cat /usr/local/cuda/version.txt
```

Check whether the CUDA version supported by the NVIDIA driver version of the node where the container is located contains the CUDA version of the container.

Helpful Links

[What Should I Do If an Error Occurs When I Deploy a Service on the GPU Node?](#)

4.3 Specification Change

4.3.1 How Do I Change the Node Specifications in a CCE Cluster?

Solution

CAUTION

If the node whose specifications need to be changed is accepted into the cluster for management, remove the node from the cluster and then change the node specifications to avoid affecting services.

- Step 1** Log in to the CCE console and click the cluster. In the navigation pane, choose **Nodes**. Click the name of the node to display the ECS details page.
- Step 2** In the upper right corner of the ECS details page, click **Stop**. After the ECS is stopped, choose **More > Modify Specifications**.
- Step 3** On the **Modify ECS Specifications** page, select a flavor name and click **Submit** to finish the specification modification. Return to ECS list page and choose **More > Start** to start the ECS.
- Step 4** Log in to the CCE console and click the cluster name to access the cluster console. In the navigation pane, choose **Nodes**. Locate the target node in the node list, and click **Sync Server Data** in the **Operation** column. After the synchronization is complete, you can view that the node specifications are the same as the modified specifications of the ECS.

----End

Common Issues

After the specifications of a node configured with CPU management policies are changed, the node may fail to be rebooted or workloads may fail to be created. In this case, see [What Should I Do If I Fail to Restart or Create Workloads on a Node After Modifying the Node Specifications?](#) to rectify the fault.

4.3.2 What Should I Do If I Fail to Restart or Create Workloads on a Node After Modifying the Node Specifications?

Context

The kubelet option **cpu-manager-policy** defaults to **static**, allowing pods with certain resource characteristics to be granted increased CPU affinity and exclusivity on the node. If you modify CCE node specifications on the ECS console, the original CPU information does not match the new CPU information. As a result, workloads on the node cannot be restarted or created.

For more information, see [Control CPU Management Policies on the Node](#).

Impact

The clusters that have enabled a CPU management policy will be affected.

Solution

Step 1 Log in to the CCE node (ECS) and delete the `cpu_manager_state` file.

Example command for the file deletion:

```
rm -rf /mnt/paas/kubernetes/kubelet/cpu_manager_state
```

Step 2 Restart the node or kubelet. The following is the kubelet restart command:

```
systemctl restart kubelet
```

Step 3 Verify that workloads on the node can be successfully restarted or created.

----End

4.4 OSs

4.4.1 What Should I Do If There Is a Service Access Failure After a Backend Service Upgrade or a 1-Second Latency When a Service Accesses a CCE Cluster?

Symptom

If the kernel version of a node is earlier than 5.9 and a CCE cluster runs in IPVS forwarding mode, there may be a service access failure after a backend service upgrade or a 1-second latency when a service accesses the CCE cluster. This is caused by a bug in reusing Kubernetes IPVS connections.

IPVS Connection Reuse Parameters

The port reuse policy of IPVS is determined by the kernel parameter `net.ipv4.vs.conn_reuse_mode`.

1. If `net.ipv4.vs.conn_reuse_mode` is set to **0**, IPVS does not reschedule a new connection, but forwards the new connection to the original RS (IPVS backend).
2. If `net.ipv4.vs.conn_reuse_mode` is set to **1**, IPVS reschedules a new connection.

Problems Caused by IPVS Connection Reuse

- **Problem 1**

If `net.ipv4.vs.conn_reuse_mode` is set to **0**, IPVS does not proactively schedule new connections with port reuse or trigger any connection termination or drop operations. Data packets of the new connections will be directly forwarded to the previously used backend pod. If the backend pod has

been deleted or recreated, an exception occurs. However, according to the current implementation logic, in a high-concurrency service access scenario, connection requests for port reuse are continuously forwarded, while kube-proxy did not delete the old ones, resulting in a service access failure.

- **Problem 2**

If **net.ipv4.vs.conn_reuse_mode** is set to **1** and the source port is the same as that of a previous connection in a high-concurrency scenario, the connection is not reused but rescheduled. According to the processing logic of `ip_vs_in()`, if **net.ipv4.vs.conntrack** is enabled, the first SYN packet is dropped. As a result, the SYN packet will be retransmitted, leading to a 1-second latency, and the performance deteriorates.

Community Settings and Impact on CCE Clusters

The default value of **net.ipv4.vs.conn_reuse_mode** on a node is **1**. However, the Kubernetes kube-proxy resets this parameter.

Cluster Version	kube-proxy Action	Impact on CCE Cluster
1.17 or earlier	By default, kube-proxy sets net.ipv4.vs.conn_reuse_mode to 0 . For details, see Fix IPVS low throughput issue .	If CCE clusters of 1.17 or earlier versions use the IPVS service forwarding mode, kube-proxy will set the net.ipv4.vs.conn_reuse_mode value of all nodes to 0 by default. This causes Problem 1 : The RS cannot be removed when the port is reused.

Cluster Version	kube-proxy Action	Impact on CCE Cluster
1.19 or later	<p>kube-proxy sets the value of net.ipv4.vs.conn_reuse_mode based on the kernel version. For details, see ipvs: only attempt setting of sysctlconnreuse on supported kernels.</p> <ul style="list-style-type: none"> If the kernel version is later than 4.1, kube-proxy will set net.ipv4.vs.conn_reuse_mode to 0. In other cases, the default value 1 will be retained. <p>NOTE This issue has been resolved in Linux kernel 5.9. Since Kubernetes 1.22, kube-proxy does not modify the net.ipv4.vs.conn_reuse_mode parameter of nodes that use the kernel 5.9 or later. For details, see Don't set sysctl net.ipv4.vs.conn_reuse_mode for kernels >=5.9.</p>	<p>If the IPVS service forwarding mode is used in CCE clusters of 1.19.16-r0 or later, the value of net.ipv4.vs.conn_reuse_mode varies with the kernel versions of node OSs.</p> <ul style="list-style-type: none"> For a node running EulerOS 2.5 or CentOS 7.6, if the kernel version is earlier than 4.1, kube-proxy will keep net.ipv4.vs.conn_reuse_mode at 1. This results in Problem 2, which is, there is a 1-second latency in the high-concurrency scenarios. For a node running Ubuntu 18.04, if the kernel version is later than 4.1, kube-proxy will set net.ipv4.vs.conn_reuse_mode to 0. This causes Problem 1: The RS cannot be removed when the port is reused. For a node running EulerOS 2.9, if the kernel version is too early, kube-proxy will set net.ipv4.vs.conn_reuse_mode to 0. This results in Problem 1. To resolve this problem, upgrade the kernel version. For details, see Rectification Plan. For a node running Huawei Cloud EulerOS 2.0 or Ubuntu 22.04, if the kernel version is later than 5.9, the problem has been resolved.

Suggestions

Evaluate the impact of these problems. If they affect your services, take the following measures:

1. Use an OS that is not affected by the preceding issues, for example, Huawei Cloud EulerOS 2.0 or Ubuntu 22.04. The newly created nodes which run EulerOS 2.9 are not affected by the preceding issues. Upgrade the earlier kernel versions used by existing nodes to the fixed version. For details, see [Rectification Plan](#).
2. Use a cluster whose forwarding mode is iptables.

Rectification Plan

If you use a node running EulerOS 2.9, check whether the kernel version meets the requirements. If the kernel version of the node is too early, reset the node or create a new one.

The following kernel versions are recommended:

- x86: 4.18.0-147.5.1.6.h686.eulerosv2r9.x86_64
- Arm: 4.19.90-vhulk2103.1.0.h584.eulerosv2r9.aarch64

Kubernetes community issue: <https://github.com/kubernetes/kubernetes/issues/81775>

5 Node Pool

5.1 What Should I Do If a Node Pool Is Abnormal?

Fault Locating

Locate the fault based on the status of the abnormal node pool, as shown in [Table 5-1](#).

Table 5-1 Node pool exceptions

Abnormal Node Pool Status	Description	Solution
Error	The node pool cannot be deleted.	Delete the node pool again. If the node pool still cannot be deleted, submit a service ticket and delete the node pool.
QuotaInsufficient	The node pool cannot be scaled out due to insufficient quota.	Submit a service ticket and increase the quota.
SoldOut	The underlying resources are insufficient.	Update the node pool configuration and select other available resources.

Abnormal Node Pool Status	Description	Solution
Configuration invalid	<p>The ECS group does not exist (ServerGroupNot Exists).</p> <p>The ECS group to which the node pool belongs is not present. This may be because you manually deleted the ECS group.</p>	<ol style="list-style-type: none"> 1. Log in to the CCE console. In the navigation pane, choose Nodes, click the Node Pools tab, and click the name of the target node pool. Click the Overview tab, click Expand, and check the ECS group to which the node pool belongs. 2. Log in to the ECS console. In the navigation pane, choose Elastic Cloud Server > ECS Group and see if the target ECS group is present. 3. If the ECS group is not present, log in to the CCE console. In the navigation pane, choose Nodes, click the Node Pools tab, locate the row containing the target node pool, and click Update. In the Advanced Settings area, unbind or change the ECS group.

5.2 What Should I Do If No Node Creation Record Is Displayed When the Node Pool Is Being Expanding?

Symptom

The node pool keeps being in the expanding state, but no node creation record is displayed in the operation record.

Troubleshooting

Check and rectify the following faults:

- Whether the specifications configured for the node pool are insufficient.
- Whether the ECS or memory quota of the tenant is insufficient.
- The ECS capacity verification of the tenant may fail if too many nodes are created at a time.

Solution

- If the resources of the ECS flavor cannot meet service requirements, use ECSs of another flavor.
- If the ECS or memory quota is insufficient, increase the quota.
- If the ECS capacity verification fails, perform the verification again.

5.3 What Should I Do If a Node Pool Scale-Out Fails?

Fault Locating

Locate the fault based on the events of the failure to scale out a node pool, as shown in [Table 5-2](#).

Table 5-2 Node pool scale-out failure

Event	Possible Cause	Reference
...call fsp to query keypair fail, error code : Ecs.0314, reason is : the keypair *** does not match the user_id ***...	<p>The possible causes are as follows:</p> <ul style="list-style-type: none"> The key pair selected for logging in to the node pool has been deleted. The key pair selected for logging in to the node pool is a private one which cannot be used by the current user to log in to the node pool and create nodes in the node pool. 	Failed to Obtain the Key Pair Used for Logging In to a Node Pool
{"error": {"message": "encrypted key id [***] is invalid.", "code": "Ecs.0912"}}	<p>The possible causes are as follows:</p> <ul style="list-style-type: none"> The KMS key ID entered during node pool creation does not exist. The KMS key ID entered during node pool creation is the key of another user, but the user has not authorized you. 	Invalid KMS Key ID
Security group [****] not found	<p>This issue can arise in the following scenarios:</p> <p>A custom security group is set up for the node pool but gets deleted, so the node pool scale-out fails.</p> <p>No custom security group is configured for the node pool and the default security group is deleted, so the node pool scale-out fails.</p>	The Security Group Specified by the Node Pool Deleted

Failed to Obtain the Key Pair Used for Logging In to a Node Pool

If a node pool scale-out fails, the event contains **Ecs.0314**. This error code indicates that the key pair used for logging in to the node pool cannot be obtained, which results in the creation failure of a new ECS.

```
...call fsp to query keypair fail, error code : Ecs.0314, reason is : the keypair *** does not match the user_id  
*** ...
```

The possible causes are as follows:

- The key pair selected for logging in to the node pool has been deleted.
- The key pair selected for logging in to the node pool is a private one which cannot be used by the current user to log in to the node pool and create nodes in the node pool.

Solution:

- If the scale-out fails due to the first cause, you can create a key pair and then create a node pool which can be logged in to using this key pair.
- If the scale-out fails due to the second cause, only the user who created the private key pair can scale out the node pool. You can use another key pair when creating a new node pool.

Invalid KMS Key ID

When a node pool fails to be expanded, the reported event contains **Ecs.0912**.

```
{"error":{"message":"encrypted key id [***] is invalid.,"code":"Ecs.0912"}}
```

The possible causes are as follows:

- The KMS key ID entered during node pool creation does not exist.
- The KMS key ID entered during node pool creation is the key of another user, but the user has not authorized you.

Solution:

- If the scale-out fails due to the first cause, ensure that the entered key ID exists.
- If the scale-out fails due to second cause, use the ID of the shared key that has been authorized to you.

The Security Group Specified by the Node Pool Deleted

When a node pool fails to be expanded, the event contains the following information:

```
Security group [****] not found
```

This issue can arise in the following scenarios:

- Scenarios 1: A custom security group is set up for the node pool but gets deleted, so the node pool scale-out fails.
- Scenarios 2: No custom security group is configured for the node pool and the default security group is deleted, so the node pool scale-out fails.

Solution:

- Scenario 1: Update the security group specified by the **customSecurityGroups** field by calling the API for updating a node pool. For details, see [Updating a Specified Node Pool](#).
- Scenario 2: Log in to the CCE console and change the **default node security group** on the **Settings** page of the cluster. The new node security group must meet the communication rules of the cluster ports. For details, see [How Can I Configure a Security Group Rule in a Cluster?](#)

5.4 How Do I Modify ECS Configurations When an ECS Cannot Be Managed by a Node Pool?

If an ECS cannot be managed by a node pool due to the reasons listed in this section, you can modify the configuration to manage the ECS.

Cause	Solution	Reference
Inconsistent flavors	Change the ECS flavor to that contained in the node pool.	Modifying the Flavor of an ECS
Inconsistent VPC and subnet	Change the VPC and subnet where the ECS resides to be the same VPC and subnet as the node pool.	Changing the VPC and Subnet of an ECS
Different billing modes	Change the billing mode of the ECS to be the same as that of the node pool.	Changing the Billing Mode of an ECS
Different data disk configuration	Change the data disk configuration of the ECS to be the same as that of the node pool.	Changing Data Disk Configuration of an ECS
Different enterprise projects	Change the enterprise project of the ECS to be the same as that of the node pool.	Changing the Enterprise Project of an ECS
Different ECS groups	Change the ECS group of the ECS to be the same as that of the node pool.	Changing the ECS Group of an ECS

Modifying the Flavor of an ECS

 **NOTE**

The flavor of the ECS to be managed must be changed to that contained in the target node pool.

For more operation guides, see [General Operations](#).

Step 1 Log in to the ECS console.

Step 2 Click the name of the target ECS. On the page displayed, click **Stop**. After the ECS is stopped, choose **More > Modify Specifications** in the **Operation** column.

Step 3 On the **Modify ECS Specifications** page, select the needed flavor and submit the application.

Step 4 Go back to the ECS list page and start the ECS.

----End

Changing the VPC and Subnet of an ECS

NOTE

You need to change the VPC and subnet of the ECS that you want to manage to match those of the target node pool.

For details, see [Changing a VPC](#).

Step 1 Log in to the ECS console.

Step 2 Locate the row containing the target ECS and choose **More > Manage Network > Change VPC** in the **Operation** column.

Step 3 Configure the parameters for changing the VPC.

- **VPC:** Select the target VPC.
- **Subnet:** Select the target subnet.
- **Private IP Address:** Select **Assign new** or **Use existing** as required.

Step 4 Click **OK**.

----End

Changing the Billing Mode of an ECS

NOTE

The billing mode of the ECS to be managed must be the same as that of the target node pool.

From Pay-per-Use to Yearly/Monthly

For details, see [Pay-per-Use to Yearly/Monthly](#).

Step 1 Log in to the ECS console.

Step 2 Locate the row containing the target ECS and choose **More > Change Billing Mode** in the **Operation** column.

Step 3 Click **OK**. Then you are switched to Billing Center.

Step 4 Select the usage duration, determine whether to enable auto-renewal, confirm the expected expiration date and price, and click **Pay**.

Step 5 Select a payment method and make your payment. Once the order is paid, yearly/monthly billing is applied.

----End

From Yearly/Monthly to Pay-per-Use

For details, see [Yearly/Monthly to Pay-per-Use](#) .

- Step 1** Log in to the ECS console.
- Step 2** Locate the row containing the target ECS and choose **More > Change to Pay-per-Use > Change to Pay-per-Use Immediately** in the **Operation** column.
- Step 3** Click **OK**. Then you are switched to Billing Center.
- Step 4** Select the resources to be changed to pay-per-use resources following instructions.
- Step 5** Confirm the refund information and click **Change to Pay-Per-Use**.
- Step 6** Confirm the resources to be changed to pay-per-use resources again and click **OK**.

----End

Changing Data Disk Configuration of an ECS

NOTE

The number, space, and type of data disks of the ECS to be managed must be the same as those of data disks in the node pool.

Data Disk Number

For more operation guides, see [Adding a Disk to an ECS](#) or [Detaching an EVS Disk from a Running ECS](#).

- Step 1** Log in to the ECS console.
- Step 2** Click the name of the target ECS to access the ECS details page.
- Step 3** Click the **Disks** tab.
 - If there are fewer data disks on the node to be managed than the number of data disks configured for the target node pool, you need to add more disks. Click **Add Disk** and configure parameters for the new disk.

NOTICE

The specifications and space of the new disk must be the same as those configured for the target node pool. You need also select **SCSI** for **Advanced Settings**.

- If there are more data disks on the node to be managed than the number of data disks configured for the target node pool, you need to remove some disks. Click **Detach** on the right of the EVS disk to be removed.

----End

Data Disk Space

For more operation guides, see [Expanding the Capacity of an EVS Disk](#).

- Step 1** Log in to the ECS console.

- Step 2** Click the name of the target ECS to access the ECS details page.
- Step 3** Click the **Disks** tab and click **Expand Capacity** on the right of the EVS disk to be expanded.
- Step 4** Configure **New Capacity** following instructions.
- Step 5** Click **Next** and submit the order following instructions.

----End

Data Disk Type

For more operation guides, see [Changing the EVS Disk Type \(OBT\)](#).

- Step 1** Log in to the ECS console.
- Step 2** Click the name of the target ECS to access the ECS details page.
- Step 3** Click the **Disks** tab and click **Modify Specifications** on the right of the EVS disk to be expanded.
- Step 4** Configure **Disk Type** following instructions.
- Step 5** Click **Submit**.

----End

Changing the Enterprise Project of an ECS

NOTE

The enterprise project of the ECS to be managed must be the same as that of the target node pool.

For more operation guides, see [Removing Resources from an Enterprise Project](#).

- Step 1** Log in to the Huawei Cloud management console.
- Step 2** Choose **Enterprise > Project Management** in the upper right corner of the page.
- Step 3** On the page displayed, select an enterprise project and click **View Resource** in the **Operation** column.
- Step 4** Select the resources to be removed and click **Remove**.
- Step 5** Select **ECSs and ECS associated resources**. Resources associated with the ECS will be automatically removed simultaneously.
- Step 6** Select the target enterprise project and click **OK**.

----End

Changing the ECS Group of an ECS

NOTE

The ECS group of the ECS to be managed must be the same as that of the target node pool.

For more operation guides, see [Managing ECS Groups](#).

- Step 1** Log in to the ECS console.
 - Step 2** In the navigation pane, choose **Elastic Cloud Server > ECS Group**.
 - Step 3** Locate the row containing the target ECS group and click **Add ECS** in the **Operation** column.
 - Step 4** In the dialog box displayed, select the ECS to be added.
 - Step 5** Click **OK** to add the ECS to the ECS group.
- End

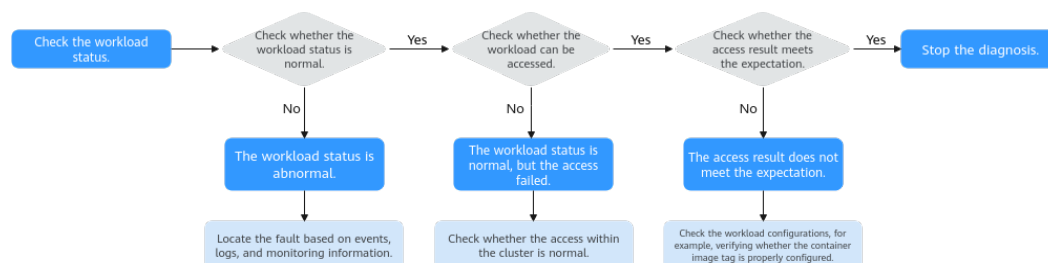
6 Workload

6.1 Workload Exception Troubleshooting

6.1.1 How Can I Find the Fault for an Abnormal Workload?

If a workload is abnormal, you can check the pod events first to locate the fault and then rectify the fault.

Fault Locating



To locate the fault of an abnormal workload, take the following steps:

Step 1 Check whether the pod is running properly.

1. Log in to the CCE console.
2. Click the cluster name to access the cluster console. In the navigation pane, choose **Workloads**.
3. In the upper left corner of the page, select a namespace, locate the target workload, and view its status.
 - If the workload is not ready, you can view pod events and determine the cause. For details, see [Viewing Pod Events](#). You can find the solution to the exception based on the events by referring to [Common Pod Issues](#).
 - If the workload is processing, wait patiently.
 - If the workload is running, no action is required. If the workload status is normal but it cannot be accessed, check whether intra-cluster access is normal.

Step 2 Check whether access within the cluster is normal.

Log in to the CCE console or use `kubectl` to obtain the pod IP address. Then, log in to the node or the pod and run `curl` or use other methods to manually call the APIs and check whether the expected result is returned.

If `{Container IP address}:{Port number}` cannot be accessed, log in to the service container, access `127.0.0.1:{Port number}`, and locate the fault.

Step 3 Check whether the access result meets the expectation.

If the workload is accessible within the cluster but the access result is not as expected, check the workload configurations, such as verifying if the image tag and environment variables are correctly configured.

----End

Common Pod Issues

Status	Description	Solution
Pending	The pod scheduling failed.	For details, see What Should I Do If Pod Scheduling Fails?
Pending	A storage volume fails to be mounted to a pod.	For details, see What Should I Do If a Storage Volume Cannot Be Mounted or the Mounting Times Out?
FailedPullImage ImagePullBackOff	The image pull failed. The image failed to be pulled again.	For details, see What Should I Do If a Pod Fails to Pull the Image?
CreateContainerError CrashLoopBackOff	The container startup failed. The container failed to restart.	For details, see What Should I Do If Container Startup Fails?
Evicted	A pod is in the Evicted state, and the pod keeps being evicted.	For details, see What Should I Do If a Pod Fails to Be Evicted?
Creating	A pod is in the Creating state.	For details, see What Should I Do If a Workload Remains in the Creating State?
Terminating	A pod is in the Terminating state.	For details, see What Should I Do If a Pod Remains in the Terminating State?

Status	Description	Solution
Stopped	A pod is in the Stopped state.	For details, see What Should I Do If a Workload Is Stopped Caused by Pod Deletion?

Viewing Pod Events

Method 1

On the CCE console, click the workload name to go to the workload details page, locate the row containing the abnormal pod, and choose **More > View Events** in the **Operation** column.

Method 2

Run **kubectl describe pod {Pod name}** to view pod events. The following shows an example:

```
$ kubectl describe pod prepare-58bd7bdf9-fthrp
...
Events:
  Type      Reason          Age   From          Message
  ----      -
Warning    FailedScheduling 49s   default-scheduler  0/2 nodes are available: 2 Insufficient cpu.
Warning    FailedScheduling 49s   default-scheduler  0/2 nodes are available: 2 Insufficient cpu.
```

6.1.2 What Should I Do If Pod Scheduling Fails?

Fault Locating

If the pod is in the **Pending** state and the event contains pod scheduling failure information, locate the cause based on the event information. For details about how to view events, see [How Can I Find the Fault for an Abnormal Workload?](#)

Troubleshooting Process

Determine the cause based on the event information, as listed in [Table 6-1](#).

Table 6-1 Pod scheduling failure

Event Information	Cause and Solution
no nodes available to schedule pods.	No node is available in the cluster. Check Item 1: Whether a Node Is Available in the Cluster

Event Information	Cause and Solution
<p>0/2 nodes are available: 2 Insufficient cpu.</p> <p>0/2 nodes are available: 2 Insufficient memory.</p>	<p>Node resources (CPU and memory) are insufficient.</p> <p>Check Item 2: Whether Node Resources (CPU and Memory) Are Sufficient</p>
<p>0/2 nodes are available: 1 node(s) didn't match node selector, 1 node(s) didn't match pod affinity rules, 1 node(s) didn't match pod affinity/anti-affinity.</p>	<p>The node and pod affinity configurations are mutually exclusive. No node meets the pod requirements.</p> <p>Check Item 3: Affinity and Anti-Affinity Configuration of the Workload</p>
<p>0/2 nodes are available: 2 node(s) had volume node affinity conflict.</p>	<p>The EVS volume mounted to the pod and the node are not in the same AZ.</p> <p>Check Item 4: Whether the Workload's Volume and Node Reside in the Same AZ</p>
<p>0/1 nodes are available: 1 node(s) had taints that the pod didn't tolerate.</p>	<p>Taints exist on the node, but the pod cannot tolerate these taints.</p> <p>Check Item 5: Taint Toleration of Pods</p>
<p>0/7 nodes are available: 7 Insufficient ephemeral-storage.</p>	<p>The ephemeral storage space of the node is insufficient.</p> <p>Check Item 6: Ephemeral Volume Usage</p>
<p>0/1 nodes are available: 1 everest driver not found at node</p>	<p>The everest-csi-driver on the node is not in the running state.</p> <p>Check Item 7: Whether everest Works Properly</p>
<p>Failed to create pod sandbox: ...</p> <p>Create more free space in thin pool or use dm.min_free_space option to change behavior</p>	<p>The node thin pool space is insufficient.</p> <p>Check Item 8: Thin Pool Space</p>
<p>0/1 nodes are available: 1 Too many pods.</p>	<p>The number of pods scheduled to the node exceeded the maximum number allowed by the node.</p> <p>Check Item 9: Number of Pods Scheduled onto the Node</p>

Check Item 1: Whether a Node Is Available in the Cluster

Log in to the CCE console and check whether the node status is **Available**. Alternatively, run the following command to check whether the node status is **Ready**:

```
$ kubectl get node
NAME          STATUS    ROLES    AGE   VERSION
192.168.0.37  Ready    <none>   21d   v1.19.10-r1.0.0-source-121-gb9675686c54267
192.168.0.71  Ready    <none>   21d   v1.19.10-r1.0.0-source-121-gb9675686c54267
```

If the status of all nodes is **Not Ready**, no node is available in the cluster.

Solution

- Add a node. If an affinity policy is not configured for the workload, the pod will be automatically migrated to the new node to ensure that services are running properly.
- Locate the unavailable node and rectify the fault. For details, see [What Should I Do If a Cluster Is Available But Some Nodes Are Unavailable?](#)
- Reset the unavailable node.

Check Item 2: Whether Node Resources (CPU and Memory) Are Sufficient

0/2 nodes are available: 2 Insufficient cpu.

0/2 nodes are available: 2 Insufficient memory.

If the resources requested by the pod exceed the allocatable resources of the node where the pod runs, the node cannot provide the resources required to run new pods and pod scheduling onto the node will definitely fail.

If the number of resources that can be allocated to a node is less than the number of resources that a pod requests, the node does not meet the resource requirements of the pod. As a result, the scheduling fails.

Solution

Add nodes to the cluster. Scale-out is the common solution to insufficient resources.

Check Item 3: Affinity and Anti-Affinity Configuration of the Workload

Inappropriate affinity policies will cause pod scheduling to fail.

Example:

An anti-affinity relationship is established between workload 1 and workload 2. Workload 1 is deployed on node 1 while workload 2 is deployed on node 2.

When you try to deploy workload 3 on node 1 and establish an affinity relationship with workload 2, a conflict occurs, resulting in a workload deployment failure.

0/2 nodes are available: 1 node(s) didn't match **node selector, 1 node(s) didn't match **pod affinity rules**, 1 node(s) didn't match **pod affinity/anti-affinity**.**

- **node selector** indicates that the node affinity is not met.

- **pod affinity rules** indicate that the pod affinity is not met.
- **pod affinity/anti-affinity** indicates that the pod affinity/anti-affinity is not met.

Solution

- When adding workload-workload affinity and workload-node affinity policies, ensure that the two types of policies do not with conflict each other. Otherwise, workload deployment will fail.
- If the workload has a node affinity policy, make sure that **supportContainer** in the label of the affinity node is set to **true**. Otherwise, pods cannot be scheduled onto the affinity node and the following event is generated:
No nodes are available that match all of the following predicates: MatchNode Selector, NodeNotSupportsContainer
If the value is **false**, the scheduling fails.

Check Item 4: Whether the Workload's Volume and Node Reside in the Same AZ

0/2 nodes are available: 2 node(s) had volume node affinity conflict. An affinity conflict occurs between volumes and nodes. As a result, the scheduling fails.

This is because EVS disks cannot be attached to nodes across AZs. For example, if the EVS volume is located in AZ 1 and the node is located in AZ 2, scheduling fails.

The EVS volume created on CCE has affinity settings by default, as shown below.

```
kind: PersistentVolume
apiVersion: v1
metadata:
  name: pvc-c29bfac7-efa3-40e6-b8d6-229d8a5372ac
spec:
  ...
  nodeAffinity:
    required:
      nodeSelectorTerms:
        - matchExpressions:
            - key: failure-domain.beta.kubernetes.io/zone
              operator: In
              values:
                -
```

Solution

In the AZ where the workload's node resides, create a volume. Alternatively, create an identical workload and select an automatically assigned cloud storage volume.

Check Item 5: Taint Toleration of Pods

0/1 nodes are available: 1 node(s) had taints that the pod didn't tolerate. This means the node is tainted and the pod cannot be scheduled to the node.

Check the taints on the node. If the following information is displayed, taints exist on the node:

```
$ kubectrl describe node 192.168.0.37
Name:          192.168.0.37
...
Taints:       key1=value1:NoSchedule
...
```

In some cases, the system automatically adds a taint to a node. The current built-in taints include:

- `node.kubernetes.io/not-ready`: The node is not ready.
- `node.kubernetes.io/unreachable`: The node controller cannot access the node.
- `node.kubernetes.io/memory-pressure`: The node has memory pressure.
- `node.kubernetes.io/disk-pressure`: The node has disk pressure. Follow the instructions described in [Check Item 4: Whether the Node Disk Space Is Insufficient](#) to handle it.
- `node.kubernetes.io/pid-pressure`: The node is under PID pressure.
- `node.kubernetes.io/network-unavailable`: The node network is unavailable.
- `node.kubernetes.io/unschedulable`: The node cannot be scheduled.
- `node.cloudprovider.kubernetes.io/uninitialized`: If an external cloud platform driver is specified when kubelet is started, kubelet adds a taint to the current node and marks it as unavailable. After **cloud-controller-manager** initializes the node, kubelet deletes the taint.

Solution

To schedule the pod to the node, use either of the following methods:

- If the taint is added by a user, you can delete the taint on the node. If the taint is **automatically added by the system**, the taint will be automatically deleted after the fault is rectified.
- Specify a toleration for the pod containing the taint. For details, see [Taints and Tolerations](#).

```
apiVersion: v1
kind: Pod
metadata:
  name: nginx
spec:
  containers:
  - name: nginx
    image: nginx:alpine
  tolerations:
  - key: "key1"
    operator: "Equal"
    value: "value1"
    effect: "NoSchedule"
```

Check Item 6: Ephemeral Volume Usage

0/7 nodes are available: 7 Insufficient ephemeral-storage. This means insufficient ephemeral storage of the node.

Check whether the size of the ephemeral volume in the pod is limited. If the size of the ephemeral volume required by the application exceeds the existing capacity of the node, the application cannot be scheduled. To solve this problem, change the size of the ephemeral volume or expand the disk capacity of the node.

```
apiVersion: v1
kind: Pod
metadata:
  name: frontend
spec:
  containers:
  - name: app
    image: images.my-company.example/app:v4
```

```
resources:
  requests:
    ephemeral-storage: "2Gi"
  limits:
    ephemeral-storage: "4Gi"
  volumeMounts:
  - name: ephemeral
    mountPath: "/tmp"
  volumes:
  - name: ephemeral
    emptyDir: {}
```

To obtain the total capacity (**Capacity**) and available capacity (**Allocatable**) of the temporary volume mounted to the node, run the **kubectl describe node** command, and view the application value and limit value of the temporary volume mounted to the node.

The following is an example of the output:

```
...
Capacity:
cpu:          4
ephemeral-storage: 61607776Ki
hugepages-1Gi: 0
hugepages-2Mi: 0
localssd:     0
localvolume:  0
memory:       7614352Ki
pods:         40
Allocatable:
cpu:          3920m
ephemeral-storage: 56777726268
hugepages-1Gi: 0
hugepages-2Mi: 0
localssd:     0
localvolume:  0
memory:       6180752Ki
pods:         40
...
Allocated resources:
(Total limits may be over 100 percent, i.e., overcommitted.)
Resource           Requests          Limits
-----
cpu                 1605m (40%)      6530m (166%)
memory              2625Mi (43%)     5612Mi (92%)
ephemeral-storage  0 (0%)           0 (0%)
hugepages-1Gi      0 (0%)           0 (0%)
hugepages-2Mi      0 (0%)           0 (0%)
localssd            0                0
localvolume         0                0
Events:             <none>
```

Check Item 7: Whether everest Works Properly

0/1 nodes are available: 1 everest driver not found at node. This means the everest-csi-driver of everest is not started properly on the node.

Check the daemon named **everest-csi-driver** in the kube-system namespace and check whether the pod is started properly. If not, delete the pod. The daemon will restart the pod.

Check Item 8: Thin Pool Space

A data disk dedicated for kubelet and the container engine will be attached to a new node. If the data disk space is insufficient, the pod cannot be created.

Solution 1: Clearing images

Perform the following operations to clear unused images:

- Nodes that use containerd
 - a. Obtain local images on the node.
`crictl images -v`
 - b. Delete the images that are not required by image ID.
`crictl rmi Image ID`
- Nodes that use Docker
 - a. Obtain local images on the node.
`docker images`
 - b. Delete the images that are not required by image ID.
`docker rmi Image ID`

NOTE

Do not delete system images such as the cce-pause image. Otherwise, pods may fail to be created.

Solution 2: Expanding the disk capacity

To expand a disk capacity, perform the following steps:

Step 1 Expand the capacity of a data disk on the EVS console.

Only the storage capacity of the EVS disk is expanded. You also need to perform the following steps to expand the capacity of the logical volume and file system.

Step 2 Log in to the CCE console and click the cluster. In the navigation pane, choose **Nodes**. Click **More > Sync Server Data** in the row containing the target node.

Step 3 Log in to the target node.

Step 4 Run the **lsblk** command to check the block device information of the node.

A data disk is divided depending on the container storage **Rootfs**:

Overlayfs: No independent thin pool is allocated. Image data is stored in **dockersys**.

1. Check the disk and partition sizes of the device.

```
# lsblk
NAME        MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda          8:0  0  50G  0 disk
├─sda1       8:1  0  50G  0 part /
└─sdb        8:16  0 150G  0 disk # The data disk has been expanded to 150 GiB, but 50 GiB
space is not allocated.
├─vgpaas-dockersys 253:0  0  90G  0 lvm  /var/lib/containerd
└─vgpaas-kubernetes 253:1  0  10G  0 lvm  /mnt/paas/kubernetes/kubelet
```

2. Expand the disk capacity.

Add the new disk capacity to the **dockersys** logical volume used by the container engine.

- a. Expand the PV capacity so that LVM can identify the new EVS capacity. `/dev/sdb` specifies the physical volume where dockersys is located.

```
pvresize /dev/sdb
```

Information similar to the following is displayed:

```
Physical volume "/dev/sdb" changed
1 physical volume(s) resized or updated / 0 physical volume(s) not resized
```

- b. Expand 100% of the free capacity to the logical volume. *vgpaas/dockersys* specifies the logical volume used by the container engine.
`lvextend -l+100%FREE -n vgpaas/dockersys`

Information similar to the following is displayed:

```
Size of logical volume vgpaas/dockersys changed from <90.00 GiB (23039 extents) to 140.00 GiB (35840 extents).
Logical volume vgpaas/dockersys successfully resized.
```

- c. Adjust the size of the file system. */dev/vgpaas/dockersys* specifies the file system path of the container engine.
`resize2fs /dev/vgpaas/dockersys`

Information similar to the following is displayed:

```
Filesystem at /dev/vgpaas/dockersys is mounted on /var/lib/containerd; on-line resizing required
old_desc_blocks = 12, new_desc_blocks = 18
The filesystem on /dev/vgpaas/dockersys is now 36700160 blocks long.
```

3. Check whether the capacity is expanded.

```
# lsblk
NAME                MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda                  8:0  0  50G  0 disk
└─sda1                8:1  0  50G  0 part /
sdb                  8:16  0 150G  0 disk
└─vgpaas-dockersys 253:0  0 140G  0 lvm  /var/lib/containerd
   └─vgpaas-kubernetes 253:1  0  10G  0 lvm  /mnt/paas/kubernetes/kubelet
```

Devicemapper: A thin pool is allocated to store image data.

1. Check the disk and partition sizes of the device.

```
# lsblk
NAME                MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
vda                  8:0  0  50G  0 disk
└─vda1                8:1  0  50G  0 part /
vdb                  8:16  0 200G  0 disk
└─vgpaas-dockersys 253:0  0  18G  0 lvm  /var/lib/docker
   └─vgpaas-thinpool_tmeta 253:1  0   3G  0 lvm
      └─vgpaas-thinpool 253:3  0  67G  0 lvm          # Space used by thinpool
         ...
      └─vgpaas-thinpool_tdata 253:2  0  67G  0 lvm
         └─vgpaas-thinpool 253:3  0  67G  0 lvm
            ...
└─vgpaas-kubernetes 253:4  0  10G  0 lvm  /mnt/paas/kubernetes/kubelet
```

2. Expand the disk capacity.

Option 1: Add the new disk capacity to the thin pool disk.

- a. Expand the PV capacity so that LVM can identify the new EVS capacity. */dev/vdb* specifies the physical volume where thinpool is located.
`pvresize /dev/vdb`

Information similar to the following is displayed:

```
Physical volume "/dev/vdb" changed
1 physical volume(s) resized or updated / 0 physical volume(s) not resized
```

- b. Expand 100% of the free capacity to the logical volume. *vgpaas/thinpool* specifies the logical volume used by the container engine.
`lvextend -l+100%FREE -n vgpaas/thinpool`

Information similar to the following is displayed:

```
Size of logical volume vgpaas/thinpool changed from <67.00 GiB (23039 extents) to <167.00 GiB (48639 extents).
Logical volume vgpaas/thinpool successfully resized.
```

- c. Do not need to adjust the size of the file system, because the thin pool is not mounted to any devices.

- d. Check whether the capacity is expanded. Run the **lsblk** command to check the disk and partition sizes of the device. If the new disk capacity has been added to the thin pool, the capacity is expanded.

```
# lsblk
NAME                MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
vda                  8:0  0  50G  0 disk
└─vda1               8:1  0  50G  0 part /
vdb                  8:16  0 200G  0 disk
├─vgpaas-dockersys  253:0  0  18G  0 lvm  /var/lib/docker
├─vgpaas-thinpool_tmeta 253:1  0   3G  0 lvm
└─vgpaas-thinpool    253:3  0 167G  0 lvm      # Thin pool space after
capacity expansion
...
├─vgpaas-thinpool_tdata 253:2  0  67G  0 lvm
├─vgpaas-thinpool    253:3  0  67G  0 lvm
...
└─vgpaas-kubernetes  253:4  0  10G  0 lvm  /mnt/paas/kubernetes/kubelet
```

Option 2: Add the new disk capacity to the **dockersys** disk.

- a. Expand the PV capacity so that LVM can identify the new EVS capacity. `/dev/vdb` specifies the physical volume where dockersys is located.

```
pvresize /dev/vdb
```

Information similar to the following is displayed:

```
Physical volume "/dev/vdb" changed
1 physical volume(s) resized or updated / 0 physical volume(s) not resized
```

- b. Expand 100% of the free capacity to the logical volume. `vgpaas/dockersys` specifies the logical volume used by the container engine.

```
lvextend -l+100%FREE -n vgpaas/dockersys
```

Information similar to the following is displayed:

```
Size of logical volume vgpaas/dockersys changed from <18.00 GiB (4607 extents) to <118.00 GiB (30208 extents).
Logical volume vgpaas/dockersys successfully resized.
```

- c. Adjust the size of the file system. `/dev/vgpaas/dockersys` specifies the file system path of the container engine.

```
resize2fs /dev/vgpaas/dockersys
```

Information similar to the following is displayed:

```
Filesystem at /dev/vgpaas/dockersys is mounted on /var/lib/docker; on-line resizing required
old_desc_blocks = 3, new_desc_blocks = 15
The filesystem on /dev/vgpaas/dockersys is now 30932992 blocks long.
```

- d. Check whether the capacity is expanded. Run the **lsblk** command to check the disk and partition sizes of the device. If the new disk capacity has been added to the dockersys, the capacity is expanded.

```
# lsblk
NAME                MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
vda                  8:0  0  50G  0 disk
└─vda1               8:1  0  50G  0 part /
vdb                  8:16  0 200G  0 disk
├─vgpaas-dockersys  253:0  0 118G  0 lvm  /var/lib/docker  # dockersys after
capacity expansion
├─vgpaas-thinpool_tmeta 253:1  0   3G  0 lvm
├─vgpaas-thinpool    253:3  0  67G  0 lvm
...
├─vgpaas-thinpool_tdata 253:2  0  67G  0 lvm
├─vgpaas-thinpool    253:3  0  67G  0 lvm
...
└─vgpaas-kubernetes  253:4  0  10G  0 lvm  /mnt/paas/kubernetes/kubelet
```

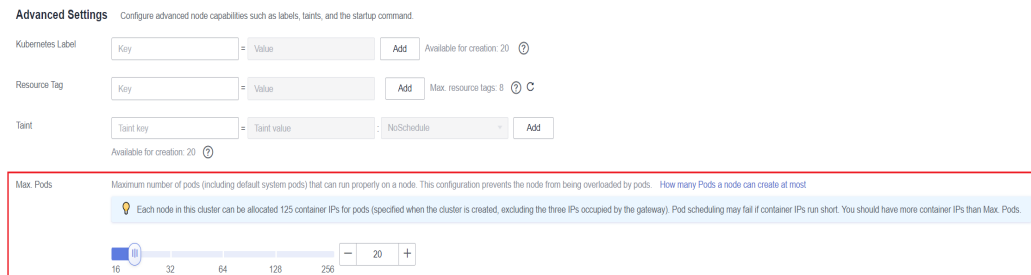
----End

Check Item 9: Number of Pods Scheduled onto the Node

0/1 nodes are available: 1 Too many pods. indicates excessive number of pods have been scheduled to the node.

When creating a node, configure **Max. Pods** in **Advanced Settings** to specify the maximum number of pods that can run properly on the node. The default value varies with the node flavor. You can change the value as needed.

Figure 6-1 Maximum number of pods

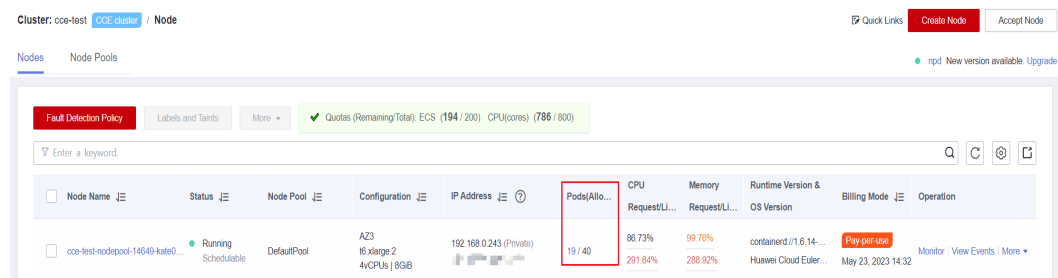


On the **Nodes** page, obtain the **Pods (Allocated/Total)** value of the node, and check whether the number of pods scheduled onto the node has reached the upper limit. If so, add nodes or change the maximum number of pods.

To change the maximum number of pods that can run on a node, do as follows:

- For nodes in the default node pool: Change the **Max. Pods** value when resetting the node.
- For nodes in a customized node pool: Change the value of the node pool parameter **max-pods**. For details, see [Configuring a Node Pool](#).

Figure 6-2 Checking the number of pods



6.1.3 What Should I Do If a Pod Fails to Pull the Image?

Fault Locating

When a workload enters the state of "Pod not ready: Back-off pulling image "xxxxx", a Kubernetes event of **PodsFailed to pull image** or **Failed to re-pull image** will be reported. For details about how to view Kubernetes events, see [Viewing Pod Events](#).

Troubleshooting Process

Determine the cause based on the event information, as listed in [Table 6-2](#).

Table 6-2 FailedPullImage

Event Information	Cause and Solution
Failed to pull image "xxx": rpc error: code = Unknown desc = Error response from daemon: Get xxx: denied: You may not login yet	<p>You have not logged in to the image repository.</p> <p>Check Item 1: Whether imagePullSecret Is Specified When You Use kubectl to Create a Workload</p>
Failed to pull image "nginx:v1.1": rpc error: code = Unknown desc = Error response from daemon: Get https://registry-1.docker.io/v2/: dial tcp: lookup registry-1.docker.io: no such host	<p>The image address is incorrectly configured.</p> <p>Check Item 2: Whether the Image Address Is Correct When a Third-Party Image Is Used</p> <p>Check Item 3: Whether an Incorrect Secret Is Used When a Third-Party Image Is Used</p>
Failed create pod sandbox: rpc error: code = Unknown desc = failed to create a sandbox for pod "nginx-6dc48bf8b6-l8xrw": Error response from daemon: mkdir xxxxx: no space left on device	<p>The disk space is insufficient.</p> <p>Check Item 4: Whether the Node Disk Space Is Insufficient</p>
Failed to pull image "xxx": rpc error: code = Unknown desc = error pulling image configuration: xxx x509: certificate signed by unknown authority	<p>An unknown or insecure certificate is used by the third-party image repository from which the image is pulled.</p> <p>Check Item 5: Whether the Remote Image Repository Uses an Unknown or Insecure Certificate</p>
Failed to pull image "xxx": rpc error: code = Unknown desc = context canceled	<p>The image size is too large.</p> <p>Check Item 6: Whether the Image Size Is Too Large</p>
Failed to pull image "docker.io/bitnami/nginx:1.22.0-debian-11-r3": rpc error: code = Unknown desc = Error response from daemon: Get https://registry-1.docker.io/v2/: net/http: request canceled while waiting for connection (Client.Timeout exceeded while awaiting headers)	<p>Check Item 7: Connection to the Image Repository</p>

Event Information	Cause and Solution
<p>ERROR: toomanyrequests: Too Many Requests.</p> <p>Or</p> <p>you have reached your pull rate limit, you may increase the limit by authenticating an upgrading</p>	<p>The rate is limited because the number of image pull times reaches the upper limit.</p> <p>Check Item 8: Whether the Number of Public Image Pull Times Reaches the Upper Limit</p>

Check Item 1: Whether imagePullSecret Is Specified When You Use kubectl to Create a Workload

If the workload status is abnormal and a Kubernetes event is displayed indicating that the pod fails to pull the image, check whether the **imagePullSecrets** field exists in the YAML file.

Items to Check

- If an image needs to be pulled from SWR, the **name** parameter must be set to **default-secret**.

```
apiVersion: extensions/v1beta1
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
```

- If an image needs to be pulled from a third-party image repository, the **imagePullSecrets** parameter must be set to the created secret name. When you use kubectl to create a workload from a third-party image, specify the **imagePullSecret** field, in which **name** indicates the name of the secret used to pull the image.

Check Item 2: Whether the Image Address Is Correct When a Third-Party Image Is Used

CCE allows you to create workloads using images pulled from third-party image repositories.

Enter the third-party image address according to requirements. The format must be **ip:port/path/name:version** or **name:version**. If no tag is specified, **latest** is used by default.

- For a private repository, enter an image address in the format of **ip:port/path/name:version**.
- For an open-source Docker repository, enter an image address in the format of **name:version**, for example, **nginx:latest**.

The following information is displayed when you fail to pull an image due to incorrect image address provided.

```
Failed to pull image "nginx:v1.1": rpc error: code = Unknown desc = Error response from daemon: Get https://registry-1.docker.io/v2/: dial tcp: lookup registry-1.docker.io: no such host
```

Solution

You can either edit your YAML file to change the image address or log in to the CCE console to replace the image on the **Upgrade** tab on the workload details page.

Check Item 3: Whether an Incorrect Secret Is Used When a Third-Party Image Is Used

Generally, a third-party image repository can be accessed only after authentication (using your account and password). CCE uses the secret authentication mode to pull images. Therefore, you need to create a secret for an image repository before pulling images from the repository.

Solution

If your secret is incorrect, images will fail to be pulled. In this case, create a new secret.

Check Item 4: Whether the Node Disk Space Is Insufficient

If the Kubernetes event contains information "no space left on device", there is no disk space left for storing the image. As a result, the image will fail to be pulled. In this case, clear the image or expand the disk space to resolve this issue.

```
Failed create pod sandbox: rpc error: code = Unknown desc = failed to create a sandbox for pod "nginx-6dc48bf8b6-l8xrw": Error response from daemon: mkdir xxxxx: no space left on device
```

Run the following command to obtain the disk space for storing images on a node:

```
ls
```

```
[root@zhouxu-20650 ~]# ls
LV          VG      Attr      LSize  Pool Origin  Data%  Meta%  Move Log Cpy%Sync  Convert
kubernetes  vgpaas  -wi-ao--- <10.00g
thinpool   vgpaas  twi-aot--- 84.00g
5.05      0.07
```

Solution 1: Clearing images

Perform the following operations to clear unused images:

- Nodes that use containerd
 - a. Obtain local images on the node.

```
crictl images -v
```

- b. Delete the images that are not required by image ID.

```
crictl rmi Image ID
```

- Nodes that use Docker

- a. Obtain local images on the node.

```
docker images
```

- b. Delete the images that are not required by image ID.

```
docker rmi Image ID
```

 **NOTE**

Do not delete system images such as the cce-pause image. Otherwise, pods may fail to be created.

Solution 2: Expanding the disk capacity

To expand a disk capacity, perform the following steps:

- Step 1** Expand the capacity of a data disk on the EVS console.

Only the storage capacity of the EVS disk is expanded. You also need to perform the following steps to expand the capacity of the logical volume and file system.

- Step 2** Log in to the CCE console and click the cluster. In the navigation pane, choose **Nodes**. Click **More > Sync Server Data** in the row containing the target node.

- Step 3** Log in to the target node.

- Step 4** Run the **lsblk** command to check the block device information of the node.

A data disk is divided depending on the container storage **Rootfs**:

Overlayfs: No independent thin pool is allocated. Image data is stored in **dockersys**.

1. Check the disk and partition sizes of the device.

```
# lsblk
NAME                MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda                  8:0    0  50G  0 disk
└─sda1                8:1    0  50G  0 part /
sdb                  8:16    0 150G  0 disk # The data disk has been expanded to 150 GiB, but 50 GiB
space is not allocated.
└─vgpaas-dockersys 253:0    0  90G  0 lvm  /var/lib/containerd
└─vgpaas-kubernetes 253:1    0  10G  0 lvm  /mnt/paas/kubernetes/kubelet
```

2. Expand the disk capacity.

Add the new disk capacity to the **dockersys** logical volume used by the container engine.

- a. Expand the PV capacity so that LVM can identify the new EVS capacity. */dev/sdb* specifies the physical volume where dockersys is located.

```
pvresize /dev/sdb
```

Information similar to the following is displayed:

```
Physical volume "/dev/sdb" changed
1 physical volume(s) resized or updated / 0 physical volume(s) not resized
```

- b. Expand 100% of the free capacity to the logical volume. *vgpaas/dockersys* specifies the logical volume used by the container engine.

```
lvextend -l+100%FREE -n vgpaas/dockersys
```

Information similar to the following is displayed:

Size of logical volume `vgpaas/dockersys` changed from <90.00 GiB (23039 extents) to 140.00 GiB (35840 extents).
Logical volume `vgpaas/dockersys` successfully resized.

- c. Adjust the size of the file system. `/dev/vgpaas/dockersys` specifies the file system path of the container engine.

```
resize2fs /dev/vgpaas/dockersys
```

Information similar to the following is displayed:

Filesystem at `/dev/vgpaas/dockersys` is mounted on `/var/lib/containerd`; on-line resizing required
old_desc_blocks = 12, new_desc_blocks = 18
The filesystem on `/dev/vgpaas/dockersys` is now 36700160 blocks long.

3. Check whether the capacity is expanded.

```
# lsblk
NAME                MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda                  8:0  0  50G  0 disk
└─sda1                8:1  0  50G  0 part /
sdb                  8:16  0 150G  0 disk
├─vgpaas-dockersys    253:0  0 140G  0 lvm  /var/lib/containerd
└─vgpaas-kubernetes  253:1  0  10G  0 lvm  /mnt/paas/kubernetes/kubelet
```

Devicemapper: A thin pool is allocated to store image data.

1. Check the disk and partition sizes of the device.

```
# lsblk
NAME                MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
vda                  8:0  0  50G  0 disk
└─vda1                8:1  0  50G  0 part /
vdb                  8:16  0 200G  0 disk
├─vgpaas-dockersys    253:0  0  18G  0 lvm  /var/lib/docker
├─vgpaas-thinpool_tmeta 253:1  0   3G  0 lvm
├─vgpaas-thinpool      253:3  0  67G  0 lvm          # Space used by thinpool
├─...
├─vgpaas-thinpool_tdata 253:2  0  67G  0 lvm
├─vgpaas-thinpool      253:3  0  67G  0 lvm
├─...
└─vgpaas-kubernetes  253:4  0  10G  0 lvm  /mnt/paas/kubernetes/kubelet
```

2. Expand the disk capacity.

Option 1: Add the new disk capacity to the thin pool disk.

- a. Expand the PV capacity so that LVM can identify the new EVS capacity. `/dev/vdb` specifies the physical volume where thinpool is located.
`pvresize /dev/vdb`

Information similar to the following is displayed:

Physical volume `"/dev/vdb"` changed
1 physical volume(s) resized or updated / 0 physical volume(s) not resized

- b. Expand 100% of the free capacity to the logical volume. `vgpaas/thinpool` specifies the logical volume used by the container engine.
`lvextend -l+100%FREE -n vgpaas/thinpool`

Information similar to the following is displayed:

Size of logical volume `vgpaas/thinpool` changed from <67.00 GiB (23039 extents) to <167.00 GiB (48639 extents).
Logical volume `vgpaas/thinpool` successfully resized.

- c. Do not need to adjust the size of the file system, because the thin pool is not mounted to any devices.
- d. Check whether the capacity is expanded. Run the `lsblk` command to check the disk and partition sizes of the device. If the new disk capacity has been added to the thin pool, the capacity is expanded.

```
# lsblk
NAME                MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
vda                  8:0  0  50G  0 disk
```

```

└─vda1          8:1  0  50G  0 part /
vdb            8:16  0  200G  0 disk
├─vgpaas-dockersys 253:0  0  18G  0 lvm  /var/lib/docker
├─vgpaas-thinpool_tmeta 253:1  0   3G  0 lvm
├─vgpaas-thinpool 253:3  0  167G  0 lvm      # Thin pool space after
capacity expansion
...
├─vgpaas-thinpool_tdata 253:2  0   67G  0 lvm
├─vgpaas-thinpool 253:3  0   67G  0 lvm
...
└─vgpaas-kubernetes 253:4  0   10G  0 lvm  /mnt/paas/kubernetes/kubelet
    
```

Option 2: Add the new disk capacity to the **dockersys** disk.

- a. Expand the PV capacity so that LVM can identify the new EVS capacity. `/dev/vdb` specifies the physical volume where dockersys is located.

```
pvresize /dev/vdb
```

Information similar to the following is displayed:

```
Physical volume "/dev/vdb" changed
1 physical volume(s) resized or updated / 0 physical volume(s) not resized
```

- b. Expand 100% of the free capacity to the logical volume. `vgpaas/dockersys` specifies the logical volume used by the container engine.

```
lvextend -l+100%FREE -n vgpaas/dockersys
```

Information similar to the following is displayed:

```
Size of logical volume vgpaas/dockersys changed from <18.00 GiB (4607 extents) to <118.00 GiB (30208 extents).
Logical volume vgpaas/dockersys successfully resized.
```

- c. Adjust the size of the file system. `/dev/vgpaas/dockersys` specifies the file system path of the container engine.

```
resize2fs /dev/vgpaas/dockersys
```

Information similar to the following is displayed:

```
Filesystem at /dev/vgpaas/dockersys is mounted on /var/lib/docker; on-line resizing required
old_desc_blocks = 3, new_desc_blocks = 15
The filesystem on /dev/vgpaas/dockersys is now 30932992 blocks long.
```

- d. Check whether the capacity is expanded. Run the **lsblk** command to check the disk and partition sizes of the device. If the new disk capacity has been added to the dockersys, the capacity is expanded.

```

# lsblk
NAME                                MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
vda                                  8:0   0  50G  0 disk
└─vda1                               8:1   0  50G  0 part /
vdb                                  8:16   0 200G  0 disk
├─vgpaas-dockersys                   253:0   0 118G  0 lvm  /var/lib/docker # dockersys after
capacity expansion
├─vgpaas-thinpool_tmeta               253:1   0   3G  0 lvm
├─vgpaas-thinpool                    253:3   0  67G  0 lvm
...
├─vgpaas-thinpool_tdata               253:2   0  67G  0 lvm
├─vgpaas-thinpool                    253:3   0  67G  0 lvm
...
└─vgpaas-kubernetes                  253:4   0  10G  0 lvm  /mnt/paas/kubernetes/kubelet
    
```

----End

Check Item 5: Whether the Remote Image Repository Uses an Unknown or Insecure Certificate

When a pod pulls an image from a third-party image repository that uses an unknown or insecure certificate, the image fails to be pulled from the node. The

pod event list contains the event "Failed to pull the image" with the cause "x509: certificate signed by unknown authority".

NOTE

The security of EulerOS 2.9 images has been improved by removing insecure or expired certificates from the system. While some third-party images on certain nodes may not report any errors, it is common for this type of error to occur in EulerOS 2.9. To fix the issue, you can carry out the following operations.

Solution

- Step 1** Check the IP address and port number of the third-party image server for which the error message "unknown authority" is displayed.

You can see the IP address and port number of the third-party image server for which the error is reported in the event information "Failed to pull image".

```
Failed to pull image "bitnami/redis-cluster:latest": rpc error: code = Unknown desc = error pulling image configuration: Get https://production.cloudflare.docker.com/registry-v2/docker/registry/v2/blobs/sha256/e8/e83853f03a2e792614e7c1e6de75d63e2d6d633b4e7c39b9d700792ee50f7b56/data?verify=1636972064-AQbl5RActnuddZV%2F3EShZwnqOe8%3D: x509: certificate signed by unknown authority
```

The IP address of the third-party image server is *production.cloudflare.docker.com*, and the default HTTPS port number is *443*.

- Step 2** Load the root certificate of the third-party image server to the node where the third-party image is to be downloaded.

Run the following command on the EulerOS and CentOS nodes with *{server_url}*: *{server_port}* replaced with the IP address and port number obtained in Step 1, for example, **production.cloudflare.docker.com:443**.

If the container engine of the node is containerd, replace **systemctl restart docker** with **systemctl restart containerd**.

```
openssl s_client -showcerts -connect {server_url}:{server_port} < /dev/null | sed -ne '/-BEGIN CERTIFICATE-/,/-END CERTIFICATE-/p' > /etc/pki/ca-trust/source/anchors/tmp_ca.crt
update-ca-trust
systemctl restart docker
```

Run the following command on Ubuntu nodes:

```
openssl s_client -showcerts -connect {server_url}:{server_port} < /dev/null | sed -ne '/-BEGIN CERTIFICATE-/,/-END CERTIFICATE-/p' > /usr/local/share/ca-certificates/tmp_ca.crt
update-ca-trust
systemctl restart docker
```

----End

Check Item 6: Whether the Image Size Is Too Large

The pod event list contains the event "Failed to pull image". This may be caused by a large image size.

```
Failed to pull image "XXX": rpc error: code = Unknown desc = context canceled
```

However, the image can be manually pulled by running the **docker pull** command.

Possible Causes

In Kubernetes clusters, there is a default timeout period for pulling images. If the image pulling progress is not updated within a certain period of time, the download will be canceled. If the node performance is poor or the image size is

too large, the image may fail to be pulled and the workload may fail to be started.

Solution

- Solution 1 (recommended):
 - a. Log in to the node and manually pull the image.
 - containerd nodes:
`crictrl pull <image-address>`
 - Docker nodes:
`docker pull <image-address>`
 - b. When creating a workload, ensure that **imagePullPolicy** is set to **IfNotPresent** (the default configuration). In this case, the workload uses the image that has been pulled to the local host.
- Solution 2 (applies to clusters of v1.25 or later): Modify the configuration parameters of the node pools. The configuration parameters for nodes in the **DefaultPool** node pool cannot be modified.
 - a. Log in to the CCE console.
 - b. Click the cluster name to access the cluster console. Choose **Nodes** in the navigation pane and click the **Node Pools** tab.
 - c. Locate the row that contains the target node pool and click **Manage**.
 - d. In the window that slides out from the right, modify the **image-pull-progress-timeout** parameter under **Docker/containerd**. This parameter specifies the timeout interval for pulling an image.
 - e. Click **OK**.

Check Item 7: Connection to the Image Repository

Symptom

The following error message is displayed during workload creation:

```
Failed to pull image "docker.io/bitnami/nginx:1.22.0-debian-11-r3": rpc error: code = Unknown desc = Error response from daemon: Get https://registry-1.docker.io/v2/: net/http: request canceled while waiting for connection (Client.Timeout exceeded while awaiting headers)
```

Possible Causes

Failed to connect to the image repository due to the disconnected network. SWR allows you to pull images only from the official Docker repository. For image pulls from other repositories, you need to access the Internet.

Solution

- Bind a public IP address to the node which needs to pull the images.
- Upload the image to SWR and then pull the image from SWR.

Check Item 8: Whether the Number of Public Image Pull Times Reaches the Upper Limit

Symptom

The following error message is displayed during workload creation:

ERROR: toomanyrequests: Too Many Requests.

Or

you have reached your pull rate limit, you may increase the limit by authenticating an upgrading: <https://www.docker.com/increase-rate-limits>.

Possible Causes

Docker Hub sets the maximum number of container image pull requests. For details, see [Understanding Your Docker Hub Rate Limit](#).

Solution

Push the frequently used image to SWR and then pull the image from SWR.

6.1.4 What Should I Do If Container Startup Fails?

Fault Locating

On the details page of a workload, if an event is displayed indicating that the container fails to be started, perform the following steps to locate the fault:

Step 1 Log in to the node where the abnormal workload is located.

Step 2 Check the ID of the container where the workload pod exits abnormally.

```
docker ps -a | grep $podName
```

Step 3 View the logs of the corresponding container.

```
docker logs $containerID
```

Rectify the fault of the workload based on logs.

Step 4 Check the error logs.

```
cat /var/log/messages | grep $containerID | grep oom
```

Check whether the system OOM is triggered based on the logs.

----End

Troubleshooting Process

Determine the cause based on the event information, as listed in [Table 6-3](#).

Table 6-3 Container startup failure

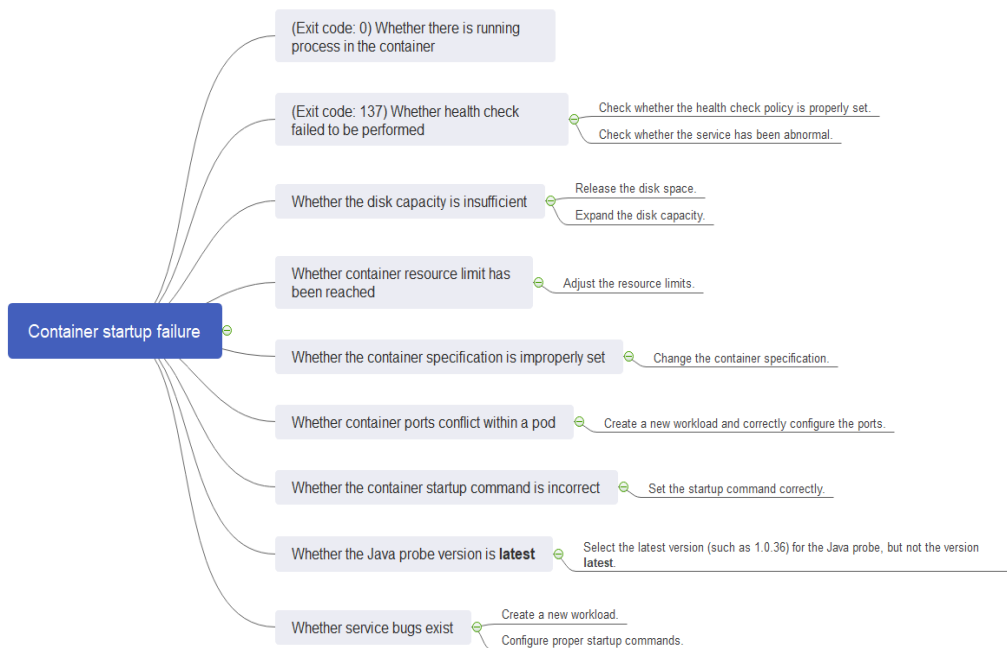
Log or Event	Cause and Solution
The log contains exit(0) .	No process exists in the container. Check whether the container is running properly. Check Item 1: Whether There Are Processes that Keep Running in the Container (Exit Code: 0)

Log or Event	Cause and Solution
<p>Event information: Liveness probe failed: Get http... The log contains exit(137).</p>	<p>Health check fails. Check Item 2: Whether Health Check Fails to Be Performed (Exit Code: 137)</p>
<p>Event information: Thin Pool has 15991 free data blocks which are less than minimum required 16383 free data blocks. Create more free space in thin pool or use dm.min_free_space option to change behavior</p>	<p>The disk space is insufficient. Clear the disk space. Check Item 3: Whether the Container Disk Space Is Insufficient</p>
<p>The keyword OOM exists in the log.</p>	<p>The memory is insufficient. Check Item 4: Whether the Upper Limit of Container Resources Has Been Reached Check Item 5: Whether the Resource Limits Are Improperly Configured for the Container</p>
<p>Address already in use</p>	<p>A conflict occurs between container ports in the pod. Check Item 6: Whether the Container Ports in the Same Pod Conflict with Each Other</p>
<p>Error: failed to start container "filebeat": Error response from daemon: OCI runtime create failed: container_linux.go:330: starting container process caused "process_linux.go:381: container init caused \"setenv: invalid argument\": unknown</p>	<p>A secret is mounted to the workload, and the value of the secret is not encrypted using Base64. Check Item 7: Whether the Value of the Secret Mounted to the Workload Meets Requirements</p>

In addition to the preceding possible causes, there are some other possible causes:

- **Check Item 8: Whether the Container Startup Command Is Correctly Configured**
- **Check Item 9: Whether the User Service Has a Bug**
- Use the correct image when you create a workload on an Arm node.

Figure 6-3 Troubleshooting process of the container restart failure



Check Item 1: Whether There Are Processes that Keep Running in the Container (Exit Code: 0)

Step 1 Log in to the node where the abnormal workload is located.

Step 2 View the container status.

```
docker ps -a | grep $podName
```

Example:

```

[root@xxx ~]# docker ps -a | grep test
1f59a7f4c777        613855f01959          "/bin/bash"         10 seconds ago    Exited (0) 10 seconds ago
k8s_container-0_test-66b79cbdb7-htcjf_default_5c388617-ac32-11e9-9168-fa163ec28742_1  2c73ac8717cc        cce-pause:2.0      12 seconds ago    Up 12 seconds
k8s_POD_test-66b79cbdb7-htcjf_default_5c388617-ac32-11e9-9168-fa163ec28742_0
  
```

If no running process exists in the container, the status code **Exited (0)** is displayed.

----End

Check Item 2: Whether Health Check Fails to Be Performed (Exit Code: 137)

The health check configured for a workload is performed on services periodically. If an exception occurs, the pod reports an event and the pod fails to be restarted.

If the liveness-type (workload liveness probe) health check is configured for the workload and the number of health check failures exceeds the threshold, the containers in the pod will be restarted. On the workload details page, if Kubernetes events contain **Liveness probe failed: Get http...**, the health check fails.

Solution

Click the workload name to go to the workload details page, click the **Containers** tab. Then select **Health Check** to check whether the policy is proper or whether services are running properly.

Check Item 3: Whether the Container Disk Space Is Insufficient

The following message refers to the thin pool disk that is allocated from the Docker disk selected during node creation. You can run the **lvs** command as user **root** to view the current disk usage.

Thin Pool has 15991 free data blocks which are less than minimum required 16383 free data blocks. Create more free space in thin pool or use dm.min_free_space option to change behavior

```
# lvs
LV          VG      Attr       LSize   Pool Origin Data%  Meta%  Move Log Cpy%Sync Convert
dockersys  vgpas   -wi-ao---- <18.00g
kubernetes vgpas   -wi-ao---- <18.00g
thinpool   vgpas   twi-aot--- 67.00g   98.04  1.32
```

Solution

Solution 1: Clearing images

Perform the following operations to clear unused images:

- Nodes that use containerd
 - a. Obtain local images on the node.
`crictl images -v`
 - b. Delete the images that are not required by image ID.
`crictl rmi Image ID`
- Nodes that use Docker
 - a. Obtain local images on the node.
`docker images`
 - b. Delete the images that are not required by image ID.
`docker rmi Image ID`

NOTE

Do not delete system images such as the cce-pause image. Otherwise, pods may fail to be created.

Solution 2: Expanding the disk capacity

To expand a disk capacity, perform the following steps:

Step 1 Expand the capacity of a data disk on the EVS console.

Only the storage capacity of the EVS disk is expanded. You also need to perform the following steps to expand the capacity of the logical volume and file system.

Step 2 Log in to the CCE console and click the cluster. In the navigation pane, choose **Nodes**. Click **More > Sync Server Data** in the row containing the target node.

Step 3 Log in to the target node.

Step 4 Run the **lsblk** command to check the block device information of the node.

A data disk is divided depending on the container storage **Rootfs**:

Overlayfs: No independent thin pool is allocated. Image data is stored in **dockersys**.

1. Check the disk and partition sizes of the device.

```
# lsblk
NAME          MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda            8:0  0  50G  0 disk
└─sda1         8:1  0  50G  0 part /
sdb            8:16 0 150G  0 disk # The data disk has been expanded to 150 GiB, but 50 GiB
space is not allocated.
├─vgpaas-dockersys 253:0  0  90G  0 lvm  /var/lib/containerd
└─vgpaas-kubernetes 253:1  0  10G  0 lvm  /mnt/paas/kubernetes/kubelet
```

2. Expand the disk capacity.

Add the new disk capacity to the **dockersys** logical volume used by the container engine.

- a. Expand the PV capacity so that LVM can identify the new EVS capacity. `/dev/sdb` specifies the physical volume where dockersys is located.

```
pvresize /dev/sdb
```

Information similar to the following is displayed:

```
Physical volume "/dev/sdb" changed
1 physical volume(s) resized or updated / 0 physical volume(s) not resized
```

- b. Expand 100% of the free capacity to the logical volume. `vgpaas/dockersys` specifies the logical volume used by the container engine.

```
lvextend -l+100%FREE -n vgpaas/dockersys
```

Information similar to the following is displayed:

```
Size of logical volume vgpaas/dockersys changed from <90.00 GiB (23039 extents) to 140.00
GiB (35840 extents).
Logical volume vgpaas/dockersys successfully resized.
```

- c. Adjust the size of the file system. `/dev/vgpaas/dockersys` specifies the file system path of the container engine.

```
resize2fs /dev/vgpaas/dockersys
```

Information similar to the following is displayed:

```
Filesystem at /dev/vgpaas/dockersys is mounted on /var/lib/containerd; on-line resizing required
old_desc_blocks = 12, new_desc_blocks = 18
The filesystem on /dev/vgpaas/dockersys is now 36700160 blocks long.
```

3. Check whether the capacity is expanded.

```
# lsblk
NAME          MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda            8:0  0  50G  0 disk
└─sda1         8:1  0  50G  0 part /
sdb            8:16 0 150G  0 disk
├─vgpaas-dockersys 253:0  0 140G  0 lvm  /var/lib/containerd
└─vgpaas-kubernetes 253:1  0  10G  0 lvm  /mnt/paas/kubernetes/kubelet
```

Devicemapper: A thin pool is allocated to store image data.

1. Check the disk and partition sizes of the device.

```
# lsblk
NAME          MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
vda            8:0  0  50G  0 disk
└─vda1         8:1  0  50G  0 part /
vdb            8:16 0 200G  0 disk
├─vgpaas-dockersys 253:0  0  18G  0 lvm  /var/lib/docker
├─vgpaas-thinpool_tmeta 253:1  0   3G  0 lvm
├─vgpaas-thinpool 253:3  0  67G  0 lvm # Space used by thinpool
├─...
├─vgpaas-thinpool_tdata 253:2  0  67G  0 lvm
├─vgpaas-thinpool 253:3  0  67G  0 lvm
├─...
└─vgpaas-kubernetes 253:4  0  10G  0 lvm  /mnt/paas/kubernetes/kubelet
```

2. Expand the disk capacity.

Option 1: Add the new disk capacity to the thin pool disk.

- a. Expand the PV capacity so that LVM can identify the new EVS capacity. `/dev/vdb` specifies the physical volume where thinpool is located.
`pvresize /dev/vdb`

Information similar to the following is displayed:

```
Physical volume "/dev/vdb" changed
1 physical volume(s) resized or updated / 0 physical volume(s) not resized
```

- b. Expand 100% of the free capacity to the logical volume. `vgpaas/thinpool` specifies the logical volume used by the container engine.
`lvextend -l+100%FREE -n vgpaas/thinpool`

Information similar to the following is displayed:

```
Size of logical volume vgpaas/thinpool changed from <67.00 GiB (23039 extents) to <167.00 GiB (48639 extents).
Logical volume vgpaas/thinpool successfully resized.
```

- c. Do not need to adjust the size of the file system, because the thin pool is not mounted to any devices.
- d. Check whether the capacity is expanded. Run the `lsblk` command to check the disk and partition sizes of the device. If the new disk capacity has been added to the thin pool, the capacity is expanded.

```
# lsblk
NAME                                MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
vda                                  8:0   0  50G  0 disk
├─vda1                                8:1   0  50G  0 part /
└─vdb                                  8:16  0 200G  0 disk
   ├─vgpaas-dockersys                 253:0   0  18G  0 lvm  /var/lib/docker
   ├─vgpaas-thinpool_tmeta            253:1   0   3G  0 lvm
   └─vgpaas-thinpool                  253:3   0 167G  0 lvm          # Thin pool space after
capacity expansion
   ...
   ├─vgpaas-thinpool_tdata            253:2   0   67G  0 lvm
   └─vgpaas-thinpool                  253:3   0   67G  0 lvm
   ...
vgpaas-kubernetes                   253:4   0  10G  0 lvm  /mnt/paas/kubernetes/kubelet
```

Option 2: Add the new disk capacity to the **dockersys** disk.

- a. Expand the PV capacity so that LVM can identify the new EVS capacity. `/dev/vdb` specifies the physical volume where dockersys is located.
`pvresize /dev/vdb`

Information similar to the following is displayed:

```
Physical volume "/dev/vdb" changed
1 physical volume(s) resized or updated / 0 physical volume(s) not resized
```

- b. Expand 100% of the free capacity to the logical volume. `vgpaas/dockersys` specifies the logical volume used by the container engine.
`lvextend -l+100%FREE -n vgpaas/dockersys`

Information similar to the following is displayed:

```
Size of logical volume vgpaas/dockersys changed from <18.00 GiB (4607 extents) to <118.00 GiB (30208 extents).
Logical volume vgpaas/dockersys successfully resized.
```

- c. Adjust the size of the file system. `/dev/vgpaas/dockersys` specifies the file system path of the container engine.
`resize2fs /dev/vgpaas/dockersys`

Information similar to the following is displayed:

```
Filesystem at /dev/vgpaas/dockersys is mounted on /var/lib/docker; on-line resizing required
old_desc_blocks = 3, new_desc_blocks = 15
The filesystem on /dev/vgpaas/dockersys is now 30932992 blocks long.
```

- d. Check whether the capacity is expanded. Run the **lsblk** command to check the disk and partition sizes of the device. If the new disk capacity has been added to the dockersys, the capacity is expanded.

```
# lsblk
NAME                                MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
vda                                  8:0   0  50G  0 disk
├─vda1                               8:1   0  50G  0 part /
└─vgdb                                8:16  0 200G  0 disk
   └─vgpaas-dockersys                 253:0  0 118G  0 lvm  /var/lib/docker # dockersys after
      capacity expansion
         └─vgpaas-thinpool_tmeta       253:1  0   3G  0 lvm
            └─vgpaas-thinpool         253:3  0  67G  0 lvm
               ...
         └─vgpaas-thinpool_tdata       253:2  0  67G  0 lvm
            └─vgpaas-thinpool         253:3  0  67G  0 lvm
               ...
         └─vgpaas-kubernetes           253:4  0  10G  0 lvm  /mnt/paas/kubernetes/kubelet
```

----End

Check Item 4: Whether the Upper Limit of Container Resources Has Been Reached

If the upper limit of container resources has been reached, OOM will be displayed in the event details as well as in the log:

```
cat /var/log/messages | grep 96feb0a425d6 | grep oom
```

```
[root@xxx ~]#
[root@xxx ~]# cat /var/log/messages | grep 96feb0a425d6 | grep oom
2019-07-22T11:57:49.441756+08:00 xxx dockerd: time="2019-07-22T11:57:49.440755329+08:00" level=info msg=event OOMKilled=true containerID=96feb0a425d6669f8f062cf3a6096868617a10711334fd5bce4a6ee6eadc82d module=libcontainerd namespace=moby topic=/tasks/oom
2019-07-22T11:59:55.828162+08:00 xxx [/bin/bash]: [2019-07-22T11:57:49.441756+08:00 xxx dockerd: time="2019-07-22T11:57:49.440755329+08:00" level=info msg=event OOMKilled=true containerID=96feb0a425d6669f8f062cf3a6096868617a10711334fd5bce4a6ee6eadc82d module=libcontainerd namespace=moby topic=/tasks/oom] return code=[127], execute failed by [root(uid=0)] from [pts/0 (192.168.0.7)]
2019-07-22T12:01:47.621029+08:00 xxx [/bin/bash]: [cat /var/log/messages | grep 96feb0a425d6 | grep oom] return code=[0], execute success by [root(uid=0)] from [pts/0 (192.168.0.7)]
[root@xxx ~]#
```

When a workload is created, if the requested resources exceed the configured upper limit, the system OOM is triggered and the container exits unexpectedly.

Check Item 5: Whether the Resource Limits Are Improperly Configured for the Container

If the resource limits set for the container during workload creation are less than required, the container fails to be restarted.

Check Item 6: Whether the Container Ports in the Same Pod Conflict with Each Other

Step 1 Log in to the node where the abnormal workload is located.

Step 2 Check the ID of the container where the workload pod exits abnormally.

```
docker ps -a | grep $podName
```

Step 3 View the logs of the corresponding container.

```
docker logs $containerID
```

Rectify the fault of the workload based on logs. As shown in the following figure, container ports in the same pod conflict. As a result, the container fails to be started.

Figure 6-4 Container restart failure due to a container port conflict

```
[root@k8s-POD_test2-65dbb945d6-xh9n2_default ~]# docker ps -a|grep test2
aebc17c4d66c          94818572c4ef          "nginx -g 'daemon ..." 8 se
conds ago            Exited (1) 5 seconds ago
k8s_container-1_test2-65dbb945d6-xh9n2_defau
lt_38892324-94b7-11e9-aa5f-fa163e07fc60_3
0c43d629292e          nginx                  "nginx -g 'daemon ..." Abou
t a minute ago      Up About a minute
k8s_container-0_test2-65dbb945d6-xh9n2_defau
lt_38892324-94b7-11e9-aa5f-fa163e07fc60_0
3484b34393ce          cfe-pause:11.23.1     "/pause"              Abou
t a minute ago      Up About a minute
k8s_POD_test2-65dbb945d6-xh9n2_default_38892
324-94b7-11e9-aa5f-fa163e07fc60_0
[root@k8s-POD_test2-65dbb945d6-xh9n2_default ~]# docker logs aebc17c4d66c
2019/06/22 06:31:29 [emerg] 1#1: bind() to 0.0.0.0:80 failed (98: Address already in use)
nginx: [emerg] bind() to 0.0.0.0:80 failed (98: Address already in use)
2019/06/22 06:31:29 [emerg] 1#1: bind() to 0.0.0.0:80 failed (98: Address already in use)
nginx: [emerg] bind() to 0.0.0.0:80 failed (98: Address already in use)
2019/06/22 06:31:29 [emerg] 1#1: bind() to 0.0.0.0:80 failed (98: Address already in use)
nginx: [emerg] bind() to 0.0.0.0:80 failed (98: Address already in use)
2019/06/22 06:31:29 [emerg] 1#1: bind() to 0.0.0.0:80 failed (98: Address already in use)
nginx: [emerg] bind() to 0.0.0.0:80 failed (98: Address already in use)
2019/06/22 06:31:29 [emerg] 1#1: bind() to 0.0.0.0:80 failed (98: Address already in use)
nginx: [emerg] bind() to 0.0.0.0:80 failed (98: Address already in use)
2019/06/22 06:31:29 [emerg] 1#1: still could not bind()
nginx: [emerg] still could not bind()
```

----End

Solution

Re-create the workload and set a port number that is not used by any other pod.

Check Item 7: Whether the Value of the Secret Mounted to the Workload Meets Requirements

Information similar to the following is displayed in the event:

```
Error: failed to start container "filebeat": Error response from daemon: OCI runtime create failed:
container_linux.go:330: starting container process caused "process_linux.go:381: container init caused
\"setenv: invalid argument\\": unknown
```

The root cause is that a secret is mounted to the workload, but the value of the secret is not encrypted using Base64.

Solution:

Create a secret on the console. The value of the secret is automatically encrypted using Base64.

If you use YAML to create a secret, you need to manually encrypt its value using Base64.

```
# echo -n "Content to be encoded" | base64
```

Check Item 8: Whether the Container Startup Command Is Correctly Configured

The error messages are as follows:


```
[root@k8s-master-001 ~]# docker ps -a | grep test1
2ae258d570c2      94818572c4ef      "/bin/sh -c 'sleep..." 14 s
econds ago      Up 12 seconds      k8s_container-0_test1-dbc59fc55-8gr9f_defau
1t_19f0d2a0-94ba-11e9-aa5f-fa163e07fc60_1
492b258c1e89      94818572c4ef      "/bin/sh -c 'sleep..." Abou
t a minute ago  Exited (1) 14 seconds ago  k8s_container-0_test1-dbc59fc55-8gr9f_defau
1t_19f0d2a0-94ba-11e9-aa5f-fa163e07fc60_0
2fcd00990111      cfe-pause:11.23.1  "/pause"           Abou
t a minute ago  Up About a minute      k8s_POD_test1-dbc59fc55-8gr9f_default_19f0d
2a0-94ba-11e9-aa5f-fa163e07fc60_0
[root@k8s-master-001 ~]# docker logs 492b258c1e89
cat: /tmp/test: No such file or directory
```

Solution

Click the workload name to go to the workload details page, click the **Containers** tab. Choose **Lifecycle**, click **Startup Command**, and ensure that the command is correct.

Check Item 9: Whether the User Service Has a Bug

Check whether the workload startup command is correctly executed or whether the workload has a bug.

- Step 1** Log in to the node where the abnormal workload is located.
- Step 2** Check the ID of the container where the workload pod exits abnormally.
- Step 3** View the logs of the corresponding container.

```
docker ps -a | grep $podName
```

```
docker logs $containerID
```

Note: In the preceding command, *containerID* indicates the ID of the container that has exited.

Figure 6-5 Incorrect startup command of the container

```
[root@dcb-ha-11638 ~]# docker ps -a | grep nginx
cf0357f617f9      3f8a4339aadd      "/bin/bash /tmp/test." 2 minutes ago
      Exited (127) 2 minutes ago      k8s_container-0_nginx-267
0177225-kt929_test_d6402ef7-4e0f-11e8-b4f7-fa163e74044e_5
c2176ce394a1      cfe-pause:3.7.6   "/pause"           5 minutes ago
      Up 5 minutes      k8s_POD_nginx-2670177225-
kt929_test_d6402ef7-4e0f-11e8-b4f7-fa163e74044e_0
[root@dcb-ha-11638 ~]# docker logs cf035
/bin/bash: /tmp/test.sh: No such file or directory
[root@dcb-ha-11638 ~]#
```

As shown in the figure above, the container fails to be started due to an incorrect startup command. For other errors, rectify the bugs based on the logs.

----End

Solution

Create a new workload and configure a correct startup command.

6.1.5 What Should I Do If a Pod Fails to Be Evicted?

Principle of Eviction

When a node is abnormal, Kubernetes will evict pods on the node to ensure workload availability.

In Kubernetes, both kube-controller-manager and kubelet can evict pods.

- **Eviction implemented by kube-controller-manager**

kube-controller-manager consists of multiple controllers, and eviction is implemented by node controller. node controller periodically checks the status of all nodes. If a node is in the **NotReady** state for a period of time, all pods on the node will be evicted.

kube-controller-manager supports the following startup parameters:

- **pod-eviction-timeout**: indicates an interval when a node is down, after which pods on that node are evicted. The default interval is 5 minutes.
- **node-eviction-rate**: indicates the number of nodes to be evicted per second. The default value is **0.1**, indicating that pods are evicted from one node every 10 seconds.
- **secondary-node-eviction-rate**: specifies a rate at which nodes are evicted in the second grade. If a large number of nodes are down in the cluster, the eviction rate will be reduced to **secondary-node-eviction-rate**. The default value is **0.01**.
- **unhealthy-zone-threshold**: specifies a threshold for an AZ to be considered unhealthy. The default value is **0.55**, meaning that if the percentage of faulty nodes in an AZ exceeds 55%, the AZ will be considered unhealthy.
- **large-cluster-size-threshold**: specifies a threshold for a cluster to be considered large. The parameter defaults to **50**. If there are more nodes than this threshold, the cluster is considered as a large one. If there are more than 55% faulty nodes in a cluster, the eviction rate is reduced to 0.01. If the cluster is a small one, the eviction rate is reduced to 0, which means, pods running on the nodes in the cluster will not be evicted.

- **Eviction implemented by kubelet**

If resources of a node are to be used up, kubelet executes the eviction policy based on the pod priority, resource usage, and resource request. If pods have the same priority, the pod that uses the most resources or requests for the most resources will be evicted first.

kube-controller-manager evicts all pods on a faulty node, while kubelet evicts some pods on a faulty node. kubelet periodically checks the memory and disk resources of nodes. If the resources are insufficient, it will evict some pods based on the priority. For details about the pod eviction priority, see [Pod selection for kubelet eviction](#).

There are soft eviction thresholds and hard eviction thresholds.

- **Soft eviction thresholds**: A grace period is configured for node resources. kubelet will reclaim node resources associated with these thresholds if that grace period elapses. If the node resource usage reaches these thresholds but falls below them before the grace period elapses, kubelet will not evict pods on the node.

You can configure soft eviction thresholds using the following parameters:

- **eviction-soft**: indicates a soft eviction threshold. If a node's **eviction signal** reaches a certain threshold, for example, **memory.available<1.5Gi**, kubelet will not immediately evict some pods on the node but wait for a grace period configured by **eviction-**

soft-grace-period. If the threshold is reached after the grace period elapses, kubelet will evict some pods on the node.

- **eviction-soft-grace-period:** indicates an eviction grace period. If a pod reaches the soft eviction threshold, it will be terminated after the configured grace period elapses. This parameter indicates the time difference for a terminating pod to respond to the threshold being met. The default grace period is 90 seconds.
- **eviction-max-pod-grace-period:** indicates the maximum allowed grace period to use when terminating pods in response to a soft eviction threshold being met.
- **Hard eviction thresholds:** Pods are immediately evicted once these thresholds are reached.

You can configure hard eviction thresholds using the following parameters:

eviction-hard: indicates a hard eviction threshold. When the **eviction signal** of a node reaches a certain threshold, for example, **memory.available<1Gi**, which means, when the available memory of the node is less than 1 GiB, a pod eviction will be triggered immediately.

kubelet supports the following default hard eviction thresholds:

- **memory.available<100Mi**
- **nodefs.available<10%**
- **imagefs.available<15%**
- **nodefs.inodesFree<5%** (for Linux nodes)

kubelet also supports other parameters:

- **eviction-pressure-transition-period:** indicates a period for which the kubelet has to wait before transitioning out of an eviction pressure condition. The default value is 5 minutes. If the time exceeds the threshold, the node is set to **DiskPressure** or **MemoryPressure**. Then some pods running on the node will be evicted. This parameter can prevent mistaken eviction decisions when a node is oscillating above and below a soft eviction threshold in some cases.
- **eviction-minimum-reclaim:** indicates the minimum number of resources that must be reclaimed in each eviction. This parameter can prevent kubelet from repeatedly evicting pods because only a small number of resources are reclaimed during pod evictions in some cases.

Fault Locating

If the pods are not evicted when the node is faulty, perform the following steps to locate the fault:

After the following command is executed, the command output shows that many pods are in the **Evicted** state.

```
kubectl get pods
```

Check results will be recorded in kubelet logs of the node. You can run the following command to search for the information:

```
cat /var/log/cce/kubernetes/kubelet.log | grep -i Evicted -C3
```

Troubleshooting Process

The issues here are described in order of how likely they are to occur.

Check these causes one by one until you find the cause of the fault.

- [Check Item 1: Whether the Node Is Under Resource Pressure](#)
- [Check Item 2: Whether Tolerations Have Been Configured for the Workload](#)
- [Check Item 3: Whether the Conditions for Stopping Pod Eviction Are Met](#)
- [Check Item 4: Whether the Allocated Resources of the Pod Are the Same as Those of the Node](#)
- [Check Item 5: Whether the Workload Pod Fails Continuously and Is Redeployed](#)

Check Item 1: Whether the Node Is Under Resource Pressure

If a node suffers resource pressure, kubelet will change the **node status** and add taints to the node. Perform the following steps to check whether the corresponding taint exists on the node:

```
$ kubectl describe node 192.168.0.37
Name:          192.168.0.37
...
Taints:        key1=value1:NoSchedule
...
```

Table 6-4 Statuses of nodes with resource pressure and solutions

Node Status	Taint	Eviction Signal	Description
MemoryPressure	node.kubernetes.io/memory-pressure	memory.available	The available memory on the node reaches the eviction thresholds.
DiskPressure	node.kubernetes.io/disk-pressure	nodefs.available, nodefs.inodesFree, imagefs.available or imagefs.inodesFree	The available disk space and inode on the root file system or image file system of the node reach the eviction thresholds.
PIDPressure	node.kubernetes.io/pid-pressure	pid.available	The available process identifier on the node is below the eviction thresholds.

Check Item 2: Whether Tolerations Have Been Configured for the Workload

Use `kubectl` or locate the row containing the target workload and choose **More > Edit YAML** in the **Operation** column to check whether tolerance is configured for the workload. For details, see [Taints and Tolerations](#).

Check Item 3: Whether the Conditions for Stopping Pod Eviction Are Met

In a cluster that runs fewer than 50 worker nodes, if the number of faulty nodes accounts for over 55% of the total nodes, the pod eviction will be suspended. In this case, Kubernetes will not attempt to evict the workload on the faulty node. For details, see [Rate limits on eviction](#).

Check Item 4: Whether the Allocated Resources of the Pod Are the Same as Those of the Node

An evicted pod will be frequently scheduled to the original node.

Possible Causes

Pods on a node are evicted based on the node resource usage. The evicted pods are scheduled based on the allocated node resources. Eviction and scheduling are based on different rules. Therefore, an evicted container may be scheduled to the original node again.

Solution

Properly allocate resources to each container.

Check Item 5: Whether the Workload Pod Fails Continuously and Is Redeployed

A workload pod fails and is being redeployed constantly.

Analysis

After a pod is evicted and scheduled to a new node, if pods in that node are also being evicted, the pod will be evicted again. Pods may be evicted repeatedly.

If a pod is evicted by kube-controller-manager, it would be in the **Terminating** state. This pod will be automatically deleted only after the node where the container is located is restored. If the node has been deleted or cannot be restored due to other reasons, you can forcibly delete the pod.

If a pod is evicted by kubelet, it would be in the **Evicted** state. This pod is only used for subsequent fault locating and can be directly deleted.

Solution

Run the following command to delete the evicted pods:

```
kubectl get pods <namespace> | grep Evicted | awk '{print $1}' | xargs kubectl delete pod <namespace>
```

In the preceding command, `<namespace>` indicates the namespace name. Configure it based on your requirements.

References

[Kubelet does not delete evicted pods](#)

6.1.6 What Should I Do If a Storage Volume Cannot Be Mounted or the Mounting Times Out?

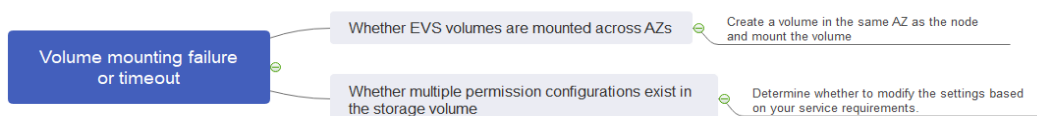
Troubleshooting Process

The issues here are described in order of how likely they are to occur.

Check these causes one by one until you find the cause of the fault.

- [Check Item 1: Whether EVS Volumes Are Mounted Across AZs](#)
- [Check Item 2: Whether Multiple Permission Configurations Exist in the Storage Volume](#)
- [Check Item 3: Whether There Is More Than One Replica for a Deployment with EVS Volumes](#)
- [Check Item 4: Whether the EVS Disk File System Is Damaged](#)

Figure 6-6 Troubleshooting for storage volume mounting failure or mounting timeout



Check Item 1: Whether EVS Volumes Are Mounted Across AZs

Symptom

Mounting an EVS volume to a StatefulSet times out.

Fault Locating

If your node is in **AZ 1** but the volume to be mounted is in **AZ 2**, the mounting times out and the volume cannot be mounted.

Solution

Create a volume in the same AZ as the node and mount the volume.

Check Item 2: Whether Multiple Permission Configurations Exist in the Storage Volume

If the volume to be mounted stores too much data and involves permission-related configurations, the file permissions need to be modified one by one, which results in mounting timeout.

Fault Locating

- Check whether the **securityContext** field contains **runAsuser** and **fsGroup**. **securityContext** is a Kubernetes field that defines the permission and access control settings of pods or containers.

- Check whether the startup commands contain commands used to obtain or modify file permissions, such as **ls**, **chmod**, and **chown**.

Solution

Determine whether to modify the settings based on your service requirements.

Check Item 3: Whether There Is More Than One Replica for a Deployment with EVS Volumes

Symptom

The pod fails to be created, and an event indicating that the storage fails to be added is reported.

```
Multi-Attach error for volume "pvc-62a7a7d9-9dc8-42a2-8366-0f5ef9db5b60" Volume is already used by pod(s) testttt-7b774658cb-lc98h
```

Fault Locating

Check whether the number of replicas of the Deployment is greater than 1.

If the Deployment uses an EVS volume, the number of replicas can only be 1. If you specify more than two pods for the Deployment on the backend, CCE does not restrict the creation of the Deployment. However, if these pods are scheduled to different nodes, some pods cannot be started because the EVS volumes used by the pods cannot be mounted to the nodes.

Solution

Set the number of replicas of the Deployment that uses an EVS volume to 1 or use other volume types.

Check Item 4: Whether the EVS Disk File System Is Damaged

Symptom

The pod fails to be created, and information similar to the following is displayed, indicating that the disk file system is damaged.

```
MountVolume.MountDevice failed for volume "pvc-08178474-c58c-4820-a828-14437d46ba6f" : rpc error: code = Internal desc = [09060def-afd0-11ec-9664-fa163eef47d0] /dev/sda has file system, but it is detected to be damaged
```

Solution

Back up the disk in EVS and run the following command to restore the file system:

```
fsck -y {Drive letter}
```

6.1.7 What Should I Do If a Workload Remains in the Creating State?

Symptom

The workload remains in the creating state.

Troubleshooting Process

The issues here are described in order of how likely they are to occur.

Check these causes one by one until you find the cause of the fault.

- [Check Item 1: Whether the cce-pause Image Is Deleted by Mistake](#)
- [Check Item 2: Modifying Node Specifications After the CPU Management Policy Is Enabled in the Cluster](#)

Check Item 1: Whether the cce-pause Image Is Deleted by Mistake

Symptom

When creating a workload, an error message indicating that the sandbox cannot be created is displayed. This is because the **cce-pause:3.1** image fails to be pulled.

```
Failed to create pod sandbox: rpc error: code = Unknown desc = failed to get sandbox image "cce-pause:3.1": failed to pull image "cce-pause:3.1": failed to pull and unpack image "docker.io/library/cce-pause:3.1": failed to resolve reference "docker.io/library/cce-pause:3.1": pulling from host **** failed with status code [manifests 3.1]: 400 Bad Request
```

Possible Causes

The image is a system image added during node creation. If the image is deleted by mistake, the workload cannot be created.

Solution

Step 1 Log in to the faulty node.

Step 2 Decompress the cce-pause image installation package.

```
tar -xzf /opt/cloud/cce/package/node-package/pause-*.tgz
```

Step 3 Import the image.

- Docker nodes:

```
docker load -i ./pause/package/image/cce-pause-*.tar
```
- containerd nodes:

```
ctr -n k8s.io images import --all-platforms ./pause/package/image/cce-pause-*.tar
```

Step 4 Create a workload.

----End

Check Item 2: Modifying Node Specifications After the CPU Management Policy Is Enabled in the Cluster

The kubelet option **cpu-manager-policy** defaults to **static**. This allows granting enhanced CPU affinity and exclusivity to pods with certain resource characteristics on the node. If you modify CCE node specifications on the ECS console, the original CPU information does not match the new CPU information. As a result, workloads on the node cannot be restarted or created.

Step 1 Log in to the CCE node (ECS) and delete the **cpu_manager_state** file.

Example command for deleting the file:

```
rm -rf /mnt/paas/kubernetes/kubelet/cpu_manager_state
```


Step 2 Restart the node or kubelet. The following is the kubelet restart command:

```
systemctl restart kubelet
```

Verify that workloads on the node can be successfully restarted or created.

For details, see [What Should I Do If I Fail to Restart or Create Workloads on a Node After Modifying the Node Specifications?](#)

----End

6.1.8 What Should I Do If a Pod Remains in the Terminating State?

Symptom

When obtaining workloads in a namespace, you may come across pods that are in the **Terminating** state.

For example, if you use the command below to obtain pods in the **aos** namespace, you may notice that some pods are in the **Terminating** state:

```
#kubectl get pod -n aos
NAME                                READY   STATUS    RESTARTS   AGE
aos-apiserver-5f8f5b5585-s9l92     1/1    Terminating    0         3d1h
aos-cmdbserver-789bf5b497-6rwrg    1/1    Running         0         3d1h
aos-controller-545d78bs8d-vm6j9    1/1    Running         3         3d1h
```

Running **kubectl delete pods <podname> -n <namespace>** cannot delete the pods.

```
kubectl delete pods aos-apiserver-5f8f5b5585-s9l92 -n aos
```

Possible Causes

The following lists some possible causes:

- **The node that runs a terminating pod is abnormal.** When a node is unavailable, CCE migrates pods on the node and sets the pods running on the node to the **Terminating** state.
After the node is restored, the pods in the **Terminating** state will be automatically deleted.
- **A container is unresponsive.** If a container within a pod fails to respond to the SIGTERM signal while being terminated, the pod can become stuck in the **Terminating** state.
- **There are unfinished requests or resource occupation within a pod.** If a process within a pod continues to run for an extended period, it may prevent the pod from being terminated and cause it to enter the **Terminating** state.
- **A pod has finalizers specified.** Finalizers are used to clean up resources before they are deleted. If a pod has finalizers specified and the cleanup process is paused or unresponsive, the pod will remain in the **Terminating** state.
- **A pod has terminationGracePeriodSeconds configured.** Once a graceful exit time is set for a pod, it will enter the **Terminating** state upon termination and be automatically deleted after exiting gracefully.

Solution

NOTE

Before forcibly deleting a pod, it is important to consider the potential risks to your services. This is especially true for StatefulSet pods, because there is a higher chance of data inconsistency and abnormal container exits. Take the time to evaluate these risks before proceeding with the operation. For details, see [Force Delete StatefulSet Pods](#).

Run the following command to forcibly delete the pods created in any ways:

```
kubectl delete pod <pod> -n <namespace> --grace-period=0 --force
```

Run the following command to delete the terminating pod described at the very beginning of this section:

```
kubectl delete pod aos-apiserver-5f8f5b5585-s9l92 -n aos --grace-period=0 --force
```

6.1.9 What Should I Do If a Workload Is Stopped Caused by Pod Deletion?

Problem

A workload is in **Stopped** state.

Cause:

The **metadata.enable** field in the YAML file of the workload is **false**. As a result, the pod of the workload is deleted and the workload is in the stopped status.

```
kind: Deployment
apiVersion: apps/v1
metadata:
  name: test
  namespace: default
  selfLink: /apis/apps/v1/namespaces/default/deployments/test
  uid: b130db9f-9306-11e9-a2a9-fa163eaff9f7
  resourceVersion: '7314771'
  generation: 1
  creationTimestamp: '2019-06-20T02:54:16Z'
  labels:
    appgroup: ''
  annotations:
    deployment.kubernetes.io/revision: '1'
    description: ''
  enable: false
spec:
```

Solution

Delete the **enable** field or set it to **true**.

6.1.10 What Should I Do if an Error Occurs When I Deploy a Service on the GPU Node?

Symptom

The following exceptions occur when services are deployed on the GPU nodes in a CCE cluster:

1. The GPU memory of containers cannot be queried.
2. Seven GPU services are deployed, but only two of them can be accessed properly. Errors are reported during the startup of the remaining five services.
 - The CUDA versions of the two services that can be accessed properly are 10.1 and 10.0, respectively.
 - The CUDA versions of the failing services are also 10.0 and 10.1.
3. Files named **core.*** are found in the GPU service containers. No such files existed in any of the previous deployments.

Fault Locating

1. The driver version of the gpu add-on is too old. After a new driver is downloaded and installed, the fault is rectified.
2. The workloads do not declare that GPU resources are required.

Suggested Solution

After you install `gpu-beta` (`gpu-device-plugin`) on a node, `nvidia-smi` will be automatically installed. If an error is reported during GPU deployment, this issue is typically caused by an NVIDIA driver installation failure. Check whether the NVIDIA driver has been downloaded.

- GPU node:
If the add-on version is earlier than 2.0.0, run the following command:

```
cd /opt/cloud/cce/nvidia/bin && ./nvidia-smi
```


If the add-on version is 2.0.0 or later and the driver installation path is changed, run the following command:

```
cd /usr/local/nvidia/bin && ./nvidia-smi
```
- Container:

```
cd /usr/local/nvidia/bin && ./nvidia-smi
```

If GPU information is returned, the device is available and the add-on has been installed.

If the driver address is incorrect, uninstall the add-on, reinstall it, and configure the correct address.

NOTE

You are advised to store the NVIDIA driver in the OBS bucket and set the bucket policy to public read.

Helpful Links

- [How Do I Rectify Failures When the NVIDIA Driver Is Used to Start Containers on GPU Nodes?](#)

6.1.11 How Can I Locate Faults Using an Exit Code?

When a container fails to be started or terminated, the exit code is recorded by Kubernetes events to report the cause. This section describes how to locate faults using an exit code.

Viewing an Exit Code

You can use `kubectl` to connect to the cluster and run the following command to check the pod:

```
kubectl describe pod {pod name}
```

In the command output, the **Exit Code** field indicates the status code of the last program exit. If the value is not **0**, the program exits abnormally. You can further analyze the cause through this code.

```
Containers:
  container-1:
    Container ID: ...
    Image: ...
    Image ID: ...
    Ports: ...
    Host Ports: ...
    Args: ...
    State: Running
      Started: Sat, 28 Jan 2023 09:06:53 +0000
    Last State: Terminated
      Reason: Error
      Exit Code: 255
      Started: Sat, 28 Jan 2023 09:01:33 +0000
      Finished: Sat, 28 Jan 2023 09:05:11 +0000
    Ready: True
    Restart Count: 1
```

Description

The exit code ranges from 0 to 255.

- If the exit code is 0, the container exits normally.
- Generally, if the abnormal exit is caused by the program, the exit code ranges from 1 to 128. In special scenarios, the exit code ranges from 129 to 255.
- When a program exits due to external interrupts, the exit code ranges from 129 to 255. When the operating system sends **an interrupt signal** to the program, the exist code is the interrupt signal value plus 128. For example, if the interrupt signal value of **SIGKILL** is 9, the exit status code is 137 (9 + 128).
- If the exist code is not in the range of 0 to 255, for example, `exit(-1)`, the exit code is automatically converted to a value that is within this range.

If the exist code is a positive number, the conversion formula is as follows:

```
code % 256
```

If the exit code is a negative number, the conversion formula is as follows:

```
256 - (|code| % 256)
```

For details, see [Exit Codes With Special Meanings](#).

Common Exit Codes

Table 6-5 Common exit codes

Exit Code	Name	Description
0	Normal exit	The container exits normally. This status code does not always indicate that an exception occurs. When there is no process in the container, it may also be displayed.
1	Common program error	There are many causes for this exception, most of which are caused by the program. You need to further locate the cause through container logs. For example, this error occurs when an x86 image is running on an Arm node.
125	The container is not running.	The possible causes are as follows: <ul style="list-style-type: none"> An undefined flag is used in the command, for example, docker run --abcd. The user-defined command in the image has insufficient permissions on the local host. The container engine is incompatible with the host OS or hardware.
126	Command calling error	The command called in the image cannot be executed. For example, the file permission is insufficient or the file cannot be executed.
127	The file or directory cannot be found.	The file or directory specified in the image cannot be found.
128	Invalid exit parameter	The container exits but no valid exit code is provided. There are multiple possible causes. You need to further locate the cause. For example, an application running on the containerd node attempts to call the docker command.
137	Immediate termination (SIGKILL)	The program is terminated by the SIGKILL signal. The common causes are as follows: <ul style="list-style-type: none"> The memory usage of the container in the pod reaches the resource limit. For example, OOM causes cgroup to forcibly stop the container. If OOM occurs, the kernel of the node stops some processes to release the memory. As a result, the container may be terminated. If the container health check fails, kubelet stops the container. Other external processes, such as malicious scripts, forcibly stop the container.

Exit Code	Name	Description
139	Segmentation error (SIGSEGV)	The container receives the SIGSEGV signal from the OS because the container attempts to access an unauthorized memory location.
143	Graceful termination (SIGTERM)	The container is correctly closed as instructed by the host. Generally, this exit code 143 does not require troubleshooting.
255	The exit code is out of range.	The container exit code is out of range. For example, <code>exit(-1)</code> may be used for abnormal exit, and -1 is automatically converted to 255. Further troubleshooting is required.

Linux Standard Interrupt Signals

You can run the **kill -l** command to view the signals and corresponding values in the Linux OS.

Table 6-6 Common Linux standard interrupt signals

Signal	Value	Action	Commit
SIGHUP	1	Term	Sent when the user terminal connection (normal or abnormal) ends.
SIGINT	2	Term	Program termination signal, which is sent by the terminal by pressing Ctrl+C .
SIGQUIT	3	Core	Similar to SIGINT , the exit command is sent by the terminal. Generally, the exit command is controlled by pressing Ctrl+\ .
SIGILL	4	Core	Invalid instruction, usually because an error occurs in the executable file.
SIGABRT	6	Core	Signal generated when the abort function is invoked. The process ends abnormally.
SIGFPE	8	Core	A floating-point arithmetic error occurs. Other arithmetic errors such as divisor 0 also occur.
SIGKILL	9	Term	Any process is terminated.
SIGSEGV	11	Core	Attempt to access an unauthorized memory location.
SIGPIPE	13	Term	The pipe is disconnected.
SIGALRM	14	Term	Indicates clock timing.

Signal	Value	Action	Commit
SIGTERM	15	Term	Process end signal, which is usually the normal exit of the program.
SIGUSR1	10	Term	This is a user-defined signal in applications.
SIGUSR2	12	Term	This is a user-defined signal in applications.
SIGCHLD	17	Ign	This signal is generated when a subprocess ends or is interrupted.
SIGCONT	18	Cont	Resume a stopped process.
SIGSTOP	19	Stop	Suspend the execution of a process.
SIGTSTP	20	Stop	Stop a process.
SIGTTIN	21	Stop	The background process reads the input value from the terminal.
SIGTTOU	22	Stop	The background process reads the output value from the terminal.

6.2 Container Configuration

6.2.1 When Is Pre-stop Processing Used?

Question

When is pre-stop processing used?

Answer

Service processing takes a long time. Pre-stop processing makes sure that during an upgrade, a pod is killed only when the service in the pod has been processed.

6.2.2 How Do I Set an FQDN for Accessing a Specified Container in the Same Namespace?

Context

When creating a workload, users can specify a container, pod, and namespace as an FQDN for accessing the container in the same namespace.

FQDN stands for Fully Qualified Domain Name, which contains both the host name and domain name. These two names are combined using a period (.).

For example, if the host name is **bigserver** and the domain name is **mycompany.com**, the FQDN is **bigserver.mycompany.com**.

Solution

Solution 1: Use the domain name for service discovery. The host name and namespace must be pre-configured. The domain name of the registered service is in the format of *service name.namespace name.svc.cluster.local*. The limitation of this solution is that the registration center must be deployed using containers.

Solution 2: Use the host network to deploy containers and then configure affinity between the containers and a node in the cluster. In this way, the service address (that is, the node address) of the containers can be determined. The registered address is the IP address of the node where the service is located. This solution allows you to deploy the registration center using VMs, whereas the disadvantage is that the host network is not as efficient as the container network.

6.2.3 What Should I Do If Health Check Probes Occasionally Fail?

When the liveness and readiness probes fail to perform the health check, locate the service fault first.

Common causes are as follows:

- The service processing takes a long time. As a result, the response times out.
- The Tomcat connection setup and waiting time are too long (for example, too many connections or threads). As a result, the response times out.
- The performance of the node where the container is located, such as the disk I/O, reaches the bottleneck. As a result, the service processing times out.

6.2.4 How Do I Set the umask Value for a Container?

Symptom

A container is started in **tailf /dev/null** mode and the directory permission is **700** after the startup script is manually executed. If the container is started by Kubernetes itself without **tailf**, the obtained directory permission is **751**.

Solution

The reason is that the umask values set in the preceding two startup modes are different. Therefore, the permissions on the created directories are different.

The umask value is used to set the default permission for a newly created file or directory. If the umask value is too small, group users or other users will have excessive permissions, posing security threats to the system. Therefore, the default umask value for all users is set to **0077**. That is, the default permission on directories created by users is **700**, and the default permission on files is **600**.

You can add the following content to the startup script to set the permission on the created directory to **700**:

1. Add **umask 0077** to the **/etc/bashrc** file and all files in **/etc/profile.d/**.
2. Run the following command:

```
echo "umask 0077" >> $FILE
```


 NOTE

FILE indicates the file name, for example, `echo "umask 0077" >> /etc/bashrc`.

3. Set the owner and group of the `/etc/bashrc` file and all files in `/etc/profile.d/` to `root`.
4. Run the following command:

```
chown root.root $FILE
```

6.2.5 What Is the Retry Mechanism When CCE Fails to Start a Pod?

CCE is a fully managed Kubernetes service and is fully compatible with Kubernetes APIs and kubectl.

In Kubernetes, the spec of a pod contains a `restartPolicy` field. The value of `restartPolicy` can be **Always**, **OnFailure**, or **Never**. The default value is **Always**.

- **Always**: When a container fails, kubelet automatically restarts the container.
- **OnFailure**: When a container stops running and the exit code is not `0` (indicating normal exit), kubelet automatically restarts the container.
- **Never**: kubelet does not restart the container regardless of the container running status.

`restartPolicy` applies to all containers in a pod.

`restartPolicy` only refers to restarts of the containers by kubelet on the same node. When containers in a pod exit, kubelet restarts them with an exponential back-off delay (10s, 20s, 40s, ...), which is capped at five minutes. Once a container has been running for 10 minutes without any problems, kubelet resets the restart backoff timer for the container.

The settings of `restartPolicy` vary depending on the controller:

- **Replication Controller (RC)** and **DaemonSet**: `restartPolicy` must be set to **Always** to ensure continuous running of the containers.
- **Job**: `restartPolicy` must be set to **OnFailure** or **Never** to ensure that containers are not restarted after being executed.

6.3 Scheduling Policies

6.3.1 How Do I Evenly Distribute Multiple Pods to Each Node?

The kube-scheduler component in Kubernetes is responsible for pod scheduling. For each newly created pod or other unscheduled pods, kube-scheduler selects an optimal node from them to run on. kube-scheduler selects a node for a pod in a 2-step operation: filtering and scoring. In the filtering step, all nodes where it is feasible to schedule the pod are filtered out. In the scoring step, kube-scheduler ranks the remaining nodes to choose the most suitable pod placement. Finally, kube-scheduler schedules the pod to the node with the highest score. If there is more than one node with the equal scores, kube-scheduler selects one of them at random.

BalancedResourceAllocation is only one of the scoring priorities. Other scoring items may also cause uneven distribution. For details about scheduling, see [Kubernetes Scheduler](#) and [Scheduling Policies](#).

You can configure pod anti-affinity policies to evenly distribute pods onto different nodes.

Example:

```
kind: Deployment
apiVersion: apps/v1
metadata:
  name: nginx
  namespace: default
spec:
  replicas: 2
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
        - name: container-0
          image: nginx:alpine
          resources:
            limits:
              cpu: 250m
              memory: 512Mi
            requests:
              cpu: 250m
              memory: 512Mi
      affinity:
        podAntiAffinity:
          # Workload anti-affinity
          preferredDuringSchedulingIgnoredDuringExecution:
            # Ensure that the following conditions are met:
            - weight: 100 # Priority that can be configured when the best-effort policy is used. The value
              ranges from 1 to 100. A larger value indicates a higher priority.
              podAffinityTerm:
                labelSelector:
                  # Select the label of the pod, which is anti-affinity with the
                  workload.
                matchExpressions:
                  - key: app
                    operator: In
                    values:
                      - nginx
                namespaces:
                  - default
                topologyKey: kubernetes.io/hostname # It takes effect on the node.
      imagePullSecrets:
        - name: default-secret
```

6.3.2 How Do I Prevent a Container on a Node from Being Evicted?

Context

During workload scheduling, two containers on a node may compete for resources. As a result, kubelet evicts both containers. This section describes how to set a policy to retain one of the containers.

Solution

kubelet uses the following criteria to evict a pod:

- Quality of Service (QoS) class: **BestEffort**, **Burstable**, and **Guaranteed**
- Consumed resources based on the pod scheduling request

Pods of different QoS classes are evicted in the following sequence:

BestEffort -> Burstable -> Guaranteed

- BestEffort pods: These pods have the lowest priority. They will be the first to be killed if the system runs out of memory.
- Burstable pods: These pods will be killed if the system runs out of memory and no BestEffort pods exist.
- Guaranteed pods: These pods will be killed if the system runs out of memory and no Burstable or BestEffort pods exist.

NOTE

- If a pod is killed because of excessive resource usage (while the node resources are still sufficient), the system tends to restart the pod on the same node.
- If resources are sufficient, you can assign the QoS class of Guaranteed to all pods. In this way, more compute resources are used to improve service performance and stability, reducing troubleshooting time and costs.
- To improve resource utilization, assign the QoS class of Guaranteed to service pods and Burstable or BestEffort to other pods (for example, filebeat).

6.3.3 Why Are Pods Not Evenly Distributed on Nodes?

Pod Scheduling Principles in Kubernetes

The kube-scheduler component in Kubernetes is responsible for pod scheduling. For each newly created pod or other unscheduled pods, kube-scheduler selects an optimal node from them to run on. kube-scheduler selects a node for a pod in a 2-step operation: filtering and scoring. In the filtering step, all nodes where it is feasible to schedule the pod are filtered out. In the scoring step, kube-scheduler ranks the remaining nodes to choose the most suitable pod placement. Finally, kube-scheduler schedules the pod to the node with the highest score. If there is more than one node with the equal scores, kube-scheduler selects one of them at random.

BalancedResourceAllocation is only one of the scoring priorities. Other scoring items may also cause uneven distribution. For details about scheduling, see [Kubernetes Scheduler](#) and [Scheduling Policies](#).

Possible Causes of Why Pods Are Not Evenly Distributed on Nodes

- Resource configurations may vary between nodes, including differences in CPU and memory size. This can result in the pods' defined requests not being met, even if a node's actual load is low. As a result, the pod cannot be scheduled to the node.
- Custom scheduling policies can be used to schedule pods based on affinity and anti-affinity rules, resulting in uneven distribution of pods across nodes.

- Nodes can have taints that prevent unscheduled pods from being assigned to them if tolerations are not set.
- Certain workloads may have unique distribution constraints. For instance, if an EVS disk is attached to a workload, the workload pods can only be scheduled to nodes within the same AZ as the disk.
- Certain pods may require specific resources, such as GPUs. In such cases, the scheduler can only assign those pods to GPU nodes.
- The health and status of a node can impact scheduling decisions. If a node is unhealthy, it may not be able to accept new pods.

Possible Causes of Why Pod Loads Are Unevenly Distributed on Nodes

When allocating pods, kube-scheduler does not take into account the actual load of applications. This can result in some nodes being heavily loaded while others are lightly loaded, especially if the application load is uneven.

Volcano Scheduler offers CPU and memory load-aware scheduling for pods and preferentially schedules pods to the node with the lightest load to balance node loads. This prevents an application or node failure due to heavy loads on a single node. For details, see [Load-aware Scheduling](#).

6.3.4 How Do I Evict All Pods on a Node?

You can run the **kubectl drain** command to safely evict all pods from a node.

NOTE

By default, the **kubectl drain** command retains some system pods, for example, everest-csi-driver.

Step 1 Use kubectl to connect to the cluster.

Step 2 Check the nodes in the cluster.

```
kubectl get node
```

Step 3 Select a node and view all pods on the node.

```
kubectl get pod --all-namespaces -owide --field-selector spec.nodeName=192.168.0.160
```

The pods on the node before eviction are as follows:

NAMESPACE	NAME	READY	STATUS	RESTARTS	AGE	IP
default	nginx-5bcc57c74b-lgcvh	1/1	Running	0	7m25s	10.0.0.140
192.168.0.160	<none>	<none>				
kube-system	coredns-6fcd88c4c-97p6s	1/1	Running	0	3h16m	10.0.0.138
192.168.0.160	<none>	<none>				
kube-system	everest-csi-controller-56796f47cc-99dtm	1/1	Running	0	3h16m	10.0.0.139
192.168.0.160	<none>	<none>				
kube-system	everest-csi-driver-dpfzl	2/2	Running	2	12d	192.168.0.160
192.168.0.160	<none>	<none>				
kube-system	icagent-tpfpv	1/1	Running	1	12d	192.168.0.160
192.168.0.160	<none>	<none>				

Step 4 Evict all pods on the node.

```
kubectl drain 192.168.0.160
```

If a pod mounted with local storage or controlled by a DaemonSet exists on the node, the message "error: unable to drain node "192.168.0.160" due to error: cannot delete DaemonSet-managed Pods..." will be displayed. The eviction

command does not take effect. You can add the following parameters to the end of the preceding command to forcibly evict the pod:

- **--delete-emptydir-data**: forcibly evicts pods mounted with local storage, for example, coredns.
- **--ignore-daemonsets**: forcibly evicts the DaemonSet pods, for example, everest-csi-driver.

In the example, both types of pods exist on the node. Therefore, the eviction command is as follows:

```
kubectl drain 192.168.0.160 --delete-emptydir-data --ignore-daemonsets
```

Step 5 After the eviction, the node is automatically marked as unschedulable. That is, the node is tainted **node.kubernetes.io/unschedulable = : NoSchedule**.

After the eviction, only system pods are retained on the node.

NAMESPACE	NAME	READY	STATUS	RESTARTS	AGE	IP	NODE
NOMINATED	NODE	READINESS GATES					
kube-system	everest-csi-driver-dpfzl	2/2	Running	2	12d	192.168.0.160	192.168.0.160
<none>	<none>						
kube-system	icagent-tpfpv	1/1	Running	1	12d	192.168.0.160	192.168.0.160
<none>	<none>						

----End

Related Operations

Drain, cordon, and uncordon operations of kubectl:

- **drain**: Safely evicts all pods from a node and marks the node as unschedulable.
- **cordon**: Marks the node as unschedulable. That is, the node is tainted **node.kubernetes.io/unschedulable = : NoSchedule**.
- **uncordon**: Marks the node as schedulable.

For more information, see the [kubectl documentation](#).

6.3.5 Why Cannot a Pod Be Scheduled to a Node?

Step 1 Check whether the node and Docker are normal. For details, see [Check Item 7: Whether Internal Components Are Normal](#).

Step 2 If the node and Docker are normal, check whether an affinity policy is configured for the pod. For details, see [Check Item 3: Affinity and Anti-Affinity Configuration of the Workload](#).

Step 3 Check whether the resources on the node are sufficient. If the resources are insufficient, expand the capacity or add nodes.

----End

6.4 Others

6.4.1 What Should I Do If a Cron Job Cannot Be Restarted After Being Stopped for a Period of Time?

When a cron job is paused mid-execution and later resumed, the controller checks the number of missed scheduling times between the last scheduled time and the current time. If this number exceeds 100, the controller will not start the job and logs the error. For details, see [CronJob limitations](#).

Cannot determine if job needs to be started. Too many missed start time (> 100). Set or decrease `.spec.startingDeadlineSeconds` or check clock skew.

For example, assume that a cron job is set to create a job every minute from 08:30:00 and the `startingDeadlineSeconds` field is not set. If the cron job controller stops running from 08:29:00 to 10:21:00, the job will not be started because the time difference between 08:29:00 and 10:21:00.00 exceeds 100 minutes, that is, the number of missed scheduling times exceeds 100 (in the example, a scheduling period is 1 minute).

If the `startingDeadlineSeconds` field is set, the controller calculates the number of missed jobs in the last x seconds (x indicates the value of `startingDeadlineSeconds`). For example, if `startingDeadlineSeconds` is set to **200**, the controller counts the number of jobs missed in the last 200 seconds. In this case, if the cron job controller stops running from 08:29:00 to 10:21:00, the job will start again at 10:22:00, because only three scheduling requests are missed in the last 200 seconds (in the example, one scheduling period is 1 minute).

Solution

Configure the `startingDeadlineSeconds` parameter in a cron job. This parameter can be created or modified only by using `kubectl` or APIs.

Example YAML:

```
apiVersion: batch/v1
kind: CronJob
metadata:
  name: hello
spec:
  startingDeadlineSeconds: 200
  schedule: "* * * * *"
  jobTemplate:
    spec:
      template:
        spec:
          containers:
            - name: hello
              image: busybox:1.28
              imagePullPolicy: IfNotPresent
              command:
                - /bin/sh
                - -c
                - date; echo Hello
          restartPolicy: OnFailure
```

If you create a cron job again, you can temporarily avoid this issue.

6.4.2 What Is a Headless Service When I Create a StatefulSet?

The inter-pod discovery service of CCE corresponds to the headless Service of Kubernetes. Headless Services specify **None** for the cluster IP (`spec.clusterIP`) in YAML, which means no cluster IP is allocated.

Differences Between Headless Services and Common Services

- **Common Services:**
One Service may be backed by multiple endpoints (pods). A client accesses the cluster IP address and the request is forwarded to the real server based on the iptables or IPVS rules to implement load balancing. For example, a Service has two endpoints, but only the Service address is returned during DNS query. The iptables or IPVS rules determine the real server that the client accesses. The client cannot access the specified endpoint.
- **Headless Services:**
When a headless Service is accessed, the actual endpoint (pod IP addresses) is returned. The headless Service points directly to each endpoint, that is, each pod has a DNS domain name. In this way, pods can access each other, achieving inter-pod discovery and access.

Headless Service Application Scenarios

If there is no difference between multiple pods of a workload, you can use a common Service and use the cluster kube-proxy to implement load balancing, for example, an Nginx Deployment.

However, in some application scenarios, pods of a workload have different roles. For example, in a Redis cluster, each Redis pod is different. They have a master/slave relationship and need to communicate with each other. In this case, a common Service cannot access a specified pod through the cluster IP address. Therefore, you need to allow the headless Service to directly access the real IP address of the pod to implement mutual access among pods.

Headless Services work with [StatefulSet](#) to deploy stateful applications, such as Redis and MySQL.

6.4.3 What Should I Do If Error Message "Auth is empty" Is Displayed When a Private Image Is Pulled?

Problem Description

When you replace the image of a container in a created workload and use an uploaded image on the CCE console, an error message "Auth is empty, only accept X-Auth-Token or Authorization" is displayed when the uploaded image is pulled.

```
Failed to pull image "IP address:Port number /magicdoom/tidb-operator:latest": rpc error: code = Unknown desc = Error response from daemon: Get https://IP address:Port number /v2/magicdoom/tidb-operator/manifests/latest: error parsing HTTP 400 response body: json: cannot unmarshal number into Go struct field Error.code of type errcode.ErrorCode: "{\"errors\": [{\"code\": 400, \"message\": \"Auth is empty, only accept X-Auth-Token or Authorization.\"}]}"
```

Solution

You can select a private image to create an application on the CCE console. In this case, CCE automatically carries the secret. This problem will not occur during the upgrade.

When you create a workload using an API, you can include the secret in Deployments to avoid this problem during the upgrade.

```
imagePullSecrets:
- name: default-secret
```

6.4.4 What Is the Image Pull Policy for Containers in a CCE Cluster?

A container image is required to create a container. Images may be stored locally or in a remote image repository.

The **imagePullPolicy** field in the Kubernetes configuration file is used to describe the image pull policy. This field has the following value options:

- **Always:** Always force a pull.
imagePullPolicy: Always
- **IfNotPresent:** The image is pulled only if it is not already present locally.
imagePullPolicy: IfNotPresent
- **Never:** The image is assumed to exist locally. No attempt is made to pull the image.
imagePullPolicy: Never

Description

1. If this field is set to **Always**, the image is pulled from the remote repository each time a container is started or restarted.
If **imagePullPolicy** is left blank, the policy defaults to **Always**.
2. If the policy is set to **IfNotPresent**:
 - a. When the required image does not exist locally, it will be pulled from the remote repository.
 - b. When the content, except the tag, of the required image is the same as that of the local image, and the image with that tag exists only in the remote repository, Kubernetes will not pull the image from the remote repository.

6.4.5 What Can I Do If a Layer Is Missing During Image Pull?

Symptom

When containerd is used as the container engine, there is a possibility that the image layer is missing when an image is pulled to a node. As a result, the workload container fails to be created.

```
Events:
Type      Reason      Age         From          Message
----      -
Normal    Scheduled   54s        default-scheduler      Successfully assigned cattle-prometheus/prometheus-server-6c69469c-f4-nfs7f to 10.14.11.139
Normal    SuccessfulMountVolume   55s        kubelet        Successfully mounted volumes for pod "prometheus-server-6c69469c-f4-nfs7f_cattle-prometheus(48ac202a-649a-429c-91ca-573d8aabc772)"
Normal    SuccessfulUpdateSecurityGroup   52s        yangtse-controller    Successfully updated security group to "e0a07f89-6fd8-431a-b901-e0075805198c"
Normal    Pulled      8s (x6 over 51s)  kubelet        Container image "100.125.0.29:20202/redis-stack/redis-stack:7/busybox:1.29.2" already present on machine
Warning   FailedCreate  7s (x6 over 50s)  kubelet        Error: failed to create containerd container: error unpacking image: failed to extract layer sha256:19d9e4e6210689cd752390e14de48b0ec61f488a05af5ab2f9cca54c299d: failed to get reader from content store: content digest sha256:8c5a7d1afbc602695fcb2cd6443743cec5ff32033ea389ea9d8773b7068185: not found
```

Possible Causes

Docker earlier than v1.10 supports the layer whose **mediaType** is **application/octet-stream**. However, containerd does not support **application/octet-stream**. As a result, the image is not pulled.

Solution

You can use either of the following methods to solve this problem:

- Use Docker v1.11 or later to repackage the image.
- Manually pull the image.
 - a. Log in to the node.
 - b. Run the following command to pull the image:
ctr -n k8s.io images pull --user u:p images
 - c. Use the newly pulled image to create a workload.

7 Networking

7.1 Network Exception Troubleshooting

7.1.1 How Do I Locate a Workload Networking Fault?

Troubleshooting Process

The issues here are described in order of how likely they are to occur.

Check these causes one by one until you find the cause of the fault.

- [Check Item 1: Container and Container Port](#)
- [Check Item 2: Node IP Address and Node Port](#)
- [Check Item 3: ELB IP Address and Port](#)
- [Check Item 4: NAT Gateway + Port](#)
- [Check Item 5: Whether the Security Group of the Node Where the Container Is Located Allows Access](#)

Check Item 1: Container and Container Port

Log in to the CCE console or use `kubectl` to query the IP address of the pod. Then, log in to the node or container in the cluster and run the `curl` command to manually call the API. Check whether the expected result is returned.

If `<container IP address>:<port>` cannot be accessed, you are advised to log in to the application container and access `<127.0.0.1>:<port>` to locate the fault.

Common issues:

1. The container port is incorrectly configured (the container does not listen to the access port).
2. The URL does not exist (no related path exists in the container).
3. A Service exception (a Service bug in the container) occurs.

4. Check whether the cluster network kernel component is abnormal (container tunnel network model: openswitch kernel component; VPC network model: ipvlan kernel component).

Check Item 2: Node IP Address and Node Port

Only NodePort or LoadBalancer Services can be accessed using the node IP address and node port.

- **NodePort Services:**

The access port of a node is the port exposed externally by the node.

- **LoadBalancer Service:**

You can view the node port of a LoadBalancer Service by editing the YAML file.

Example:

nodePort: 30637 indicates the exposed node port. **targetPort: 80** indicates the exposed pod port. **port: 123** is the exposed Service port. LoadBalancer Services also use this port to configure the ELB listener.

```
spec:
  ports:
    - name: cce-service-0
      protocol: TCP
      port: 123
      targetPort: 80
      nodePort: 30637
```

After finding the node port (nodePort), access <IP address>:<port> of the node where the container is located and check whether the expected result is returned.

Common issues:

1. The service port is not allowed in the inbound rules of the node.
2. A custom route is incorrectly configured for the node.
3. The label of the pod does not match that of the Service (created using kubectl or API).

Check Item 3: ELB IP Address and Port

There are several possible causes if <IP address>:<port> of the ELB cannot be accessed, but <IP address>:<port> of the node can be accessed.

Possible causes:

- The backend server group of the port or URL does not meet the expectation.
- The security group on the node has not exposed the related protocol or port to the ELB.
- The health check of the layer-4 load balancing is not enabled.
- The certificate used for Services of layer-7 load balancing has expired.

Common issues:

1. When exposing a layer-4 ELB load balancer, if you have not enabled health check on the console, the load balancer may route requests to abnormal nodes.
2. For UDP access, the ICMP port of the node has not been allowed in the inbound rules.
3. The label of the pod does not match that of the Service (created using kubectl or API).

Check Item 4: NAT Gateway + Port

Generally, no EIP is configured for the backend server of NAT. Otherwise, exceptions such as network packet loss may occur.

Check Item 5: Whether the Security Group of the Node Where the Container Is Located Allows Access

Log in to the management console and choose **Service List > Networking > Virtual Private Cloud**. On the Network console, choose **Access Control > Security Groups**, locate the security group rule of the CCE cluster, and modify and harden the security group rule.

- CCE cluster:

The security group name of the node is **{Cluster name}-cce-node-{Random characters}**.

Check the following:

- IP address, port, and protocol of an external request to access the workloads in the cluster. They must be allowed in the inbound rule of the cluster security group.
- IP address, port, and protocol of a request sent by a workload to visit external applications outside the cluster. They must be allowed in the outbound rule of the cluster security group.

For details about security group configuration, see [How Can I Configure a Security Group Rule in a Cluster?](#)

7.1.2 Why Does the Browser Return Error Code 404 When I Access a Deployed Application?

CCE does not return any error code when you fail to access your applications using a browser. Check your services first.

404 Not Found

If the error code shown in the following figure is returned, it indicates that the ELB cannot find the corresponding forwarding policy. Check the forwarding policies.

Figure 7-1 404:ALB

404 Not Found

ALB

If the error code shown in the following figure is returned, it indicates that errors occur on Nginx (your services). In this case, check your services.

Figure 7-2 404:nginx/1.**.*

404 Not Found

nginx/1.14.0

7.1.3 What Should I Do If a Container Fails to Access the Internet?

If a container cannot access the Internet, check whether the node where the container is located can access the Internet. Then check whether the network configuration of the container is correct. For example, check whether the DNS configuration can resolve the domain name.

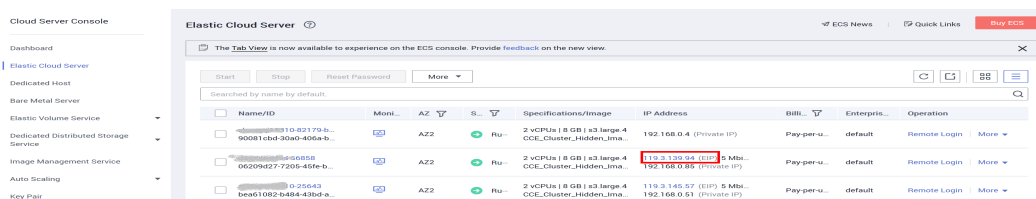
Check Item 1: Whether the Node Can Access the Internet

Step 1 Log in to the ECS console.

Step 2 Check whether an EIP has been bound to the ECS (node) or whether the ECS has a NAT gateway configured.

Figure 7-3 shows that an EIP has been bound. If no EIP is displayed, bind an EIP to the ECS.

Figure 7-3 Node with an EIP bound



----End

Check Item 2: Whether a Network ACL Has Been Configured for the Node

Step 1 Log in to the VPC console.

Step 2 In the navigation pane on the left, choose **Access Control > Network ACLs**.

Step 3 Check whether a network ACL has been configured for the subnet where the node is located and whether external access is restricted.

----End

Check Item 3: Whether the DNS Configuration of the Container Is Correct

Run `cat /etc/resolv.conf` in the container to check the DNS configuration. An example is as follows:

```
nameserver 10.247.x.x
search default.svc.cluster.local svc.cluster.local cluster.local
options ndots:5
```

If **nameserver** is set to **10.247.x.x**, DNS is connected to the CoreDNS of the cluster. Ensure that the CoreDNS of the cluster is running properly. If another IP address is displayed, an in-cloud or on-premises DNS server is used. Ensure that the domain name resolution is correctly configured.

7.1.4 What Should I Do If a Node Fails to Connect to the Internet (Public Network)?

If a node fails to be connected to the Internet, perform the following operations:

Check Item 1: Whether an EIP Has Been Bound to the Node

Log in to the ECS console and check whether an EIP has been bound to the ECS corresponding to the node.

If there is an IP address in the EIP column, an EIP has been bound. If there is no IP address in that column, bind one.

Check Item 2: Whether a Network ACL Has Been Configured for the Node

Log in to the VPC console. In the navigation pane, choose **Access Control > Network ACLs**. Check whether a network ACL has been configured for the subnet where the node is located and whether external access is restricted.

7.1.5 What Should I Do If Nginx Ingress Access in the Cluster Is Abnormal After the NGINX Ingress Controller Add-on Is Upgraded?

Symptom

An Nginx ingress whose type is not specified (`kubernetes.io/ingress.class: nginx` is not added to annotations) exists in the cluster. After the NGINX Ingress Controller add-on is upgraded from 1.x to 2.x, services are interrupted.

Fault Locating

For an Nginx ingress, check the YAML. If the ingress type is not specified in the YAML file and the ingress is managed by the NGINX Ingress Controller, the ingress is at risk.

Step 1 Check the ingress type.

Run the following command:

```
kubectl get ingress <ingress-name> -oyaml | grep -E 'kubernetes.io/ingress.class: | ingressClassName:'
```

- Fault scenario: If the command output is empty, the ingress type is not specified.
- Normal scenario: The command output is not empty, indicating that the ingress type has been specified by **annotations** or **ingressClassName**.

```
[root@192-168-0-31 paas]# kubectl get ingress test -oyaml | grep -E 'kubernetes.io/ingress.class: | ingressClassName:' -B 1
Warning: extensions/v1beta1 Ingress is deprecated in v1.14+, unavailable in v1.22+; use networking.k8s.io/v1 Ingress
annotations:
  kubernetes.io/ingress.class: nginx
spec:
  ingressClassName: nginx
```

Step 2 Ensure that the ingress is managed by the Nginx ingress Controller. The LoadBalancer Ingresses are not affected by this issue.

- For clusters of v1.19, confirm this issue using **managedFields**.

```
kubectl get ingress <ingress-name> -oyaml | grep 'manager: nginx-ingress-controller'
```

```
[root@192-168-0-31 paas]# kubectl get ingress test -oyaml | grep 'manager: nginx-ingress-controller'
Warning: extensions/v1beta1 Ingress is deprecated in v1.14+, unavailable in v1.22+; use networking.k8s.io/v1 Ingress
manager: nginx-ingress-controller
```

- For clusters of other versions, check the logs of the NGINX Ingress Controller pod.

```
kubectl logs -nkube-system cceaddon-nginx-ingress-controller-545db6b4f7-bv74t | grep 'updating Ingress status'
```

```
[root@192-168-0-31 paas]# kubectl logs -nkube-system cceaddon-nginx-ingress-controller-545db6b4f7-bv74t | grep 'updating Ingress status'
8 status.go:281] "updating Ingress status" namespace="default" ingress="test" currentValue=[] newV
alue={{IP: Hostname: Ports:[]}} {IP: Hostname: Ports:[]}}
```

If the fault persists, contact technical support.

----End

Solution

Add an annotation to the Nginx ingress as follows:

```
kubectl annotate ingress <ingress-name> kubernetes.io/ingress.class=nginx
```

NOTICE

There is no need to add this annotation to LoadBalancer Ingresses. **Verify** that these Ingresses are managed by NGINX Ingress Controller.

Possible Causes

The nginx-ingress add-on is developed based on the NGINX Ingress Controller template and image of the open source community.

For the NGINX Ingress Controller of an earlier version (community version v0.49 or earlier, corresponding to CCE nginx-ingress version v1.x.x), the ingress type is not

specified as `nginx` during Ingress creation, which is, **`kubernetes.io/ingress.class: nginx`** is not added to annotations. This Ingress can also be managed by Nginx Ingress Controller. For details, see the [GitHub code](#).

For the NGINX Ingress Controller of a later version (community version v1.0.0 or later, corresponding to CCE nginx-ingress version 2.x.x), if the ingress type is not specified as `nginx` during Ingress creation, this Ingress will be ignored by the NGINX Ingress Controller and the Ingress rules become invalid. The services will be interrupted. For details, see the [GitHub code](#).

Related link: <https://github.com/kubernetes/ingress-nginx/pull/7341>

You can specify the ingress type in either of the following ways:

- Add the **`kubernetes.io/ingress.class: nginx`** annotation to the Ingress.
- Use `spec`. Set the **`.spec.ingressClassName`** field to **`nginx`**. IngressClass resources are required.

An example is as follows:

```
apiVersion: networking.k8s.io/v1beta1
kind: Ingress
metadata:
  name: test
  namespace: default
  annotations:
    kubernetes.io/ingress.class: nginx
spec:
  ingressClassName: nginx
  rules:
...
status:
  loadBalancer: {}
```

7.1.6 What Could Cause Access Exceptions After Configuring an HTTPS Certificate for a LoadBalancer Ingress?

If you configure an HTTPS certificate for a LoadBalancer ingress, access may become abnormal if any of the following issues arise. To fix the problem, refer to the causes listed in the table.

Cause	Symptom	Solution
The certificate has expired.	The error similar to the following is displayed when the <code>curl</code> command is executed: SSL certificate problem: certificate has expired	Replace the certificate in a timely manner.
An unmatched HTTPS certificate chain is used by a client to verify the HTTPS certificate configured for the LoadBalancer ingress.	The error similar to the following is displayed when the <code>curl</code> command is executed: SSL certificate problem: unable to get local issuer certificate	Ensure that the HTTPS certificate chain on the client matches the certificate configured for the LoadBalancer ingress.

Cause	Symptom	Solution
No domain name is specified when a certificate is created.	The error similar to the following is displayed when the curl command is executed: SSL: unable to obtain common name from peer certificate	Specify a domain name when creating a certificate.
The domain name to be accessed is different from the domain name of the HTTPS certificate.	The error similar to the following is displayed when the curl command is executed: SSL: certificate subject name 'example.com' does not match target host name 'test.com'	Configure a certificate that matches the domain name for the ingress.

 **NOTE**

You can run the following command to check the certificate information, such as expiration time and domain name. **ca.crt** specifies the certificate path.

```
openssl x509 -in ca.crt -subject -noout -text
```

Updating a Certificate

- To update a TLS certificate, modify the secret where the certificate is imported to on CCE. The TLS certificate is imported to a secret first. CCE then automatically handles the certificate configurations on the ELB console and gives a name to the certificate (started with **k8s_plb_default**). This certificate, which is generated by CCE, **cannot be modified or deleted from the ELB console**.
- To update a certificate created on the ELB console, modify the certificate on the ELB console. There is no need to manually set up the cluster secret.

7.2 Network Planning

7.2.1 What Is the Relationship Between Clusters, VPCs, and Subnets?

A VPC is similar to a private local area network (LAN) managed by a home gateway whose IP address is 192.168.0.0/16. A VPC is a private network built on the cloud and provides basic network environment for running ECSs, ELBs, and middleware. Networks of different scales can be configured based on service requirements. Generally, you can set the CIDR block to 10.0.0.0/8–24, 172.16.0.0/12–24, or 192.168.0.0/16–24. The largest CIDR block is 10.0.0.0/8, which corresponds to a class A network.

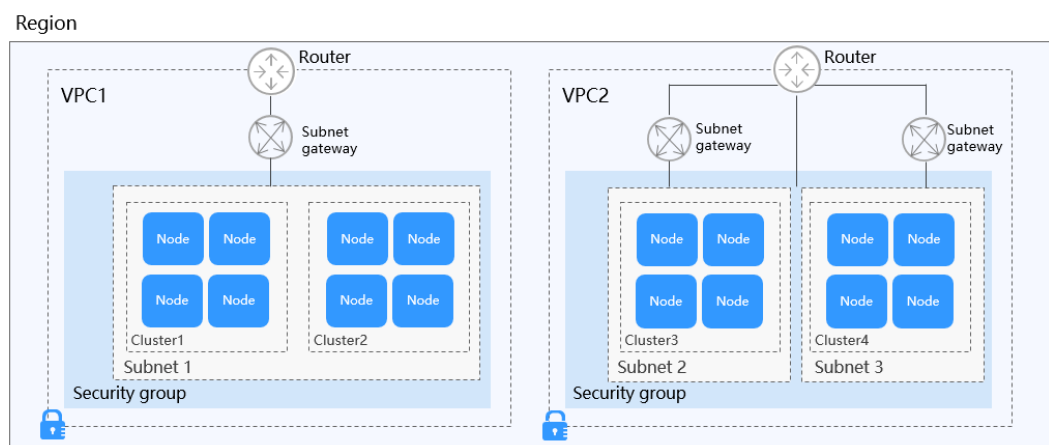
A VPC can be divided into multiple subnets. Security groups are configured to determine whether these subnets can communicate with each other. This ensures that subnets can be isolated from each other, so that you can deploy different services on different subnets.

A cluster is one or a group of cloud servers (also known as nodes) in the same VPC. It provides computing resource pools for running containers.

As shown in [Figure 7-4](#), a region may consist of multiple VPCs. A VPC consists of one or more subnets. The subnets communicate with each other through a subnet gateway. A cluster is created in a subnet. There are three scenarios:

- Different clusters are created in different VPCs.
- Different clusters are created in the same subnet.
- Different clusters are created in different subnets.

Figure 7-4 Relationship between clusters, VPCs, and subnets



7.2.2 How Can I Configure a Security Group Rule in a Cluster?

CCE is a universal container platform. Its default security group rules apply to common scenarios. When a cluster is created, a security group is automatically created for the master node and worker node, separately. The security group name of the master node is $\{Cluster\ name\}\text{-cce-control-}\{Random\ ID\}$, and the security group name of the worker node is $\{Cluster\ name\}\text{-cce-node-}\{Random\ ID\}$.

You can modify the security group rules on the VPC console as required. (Log in to the management console, choose **Service List** > **Networking** > **Virtual Private Cloud**. On the page displayed, choose **Access Control** > **Security Groups** in the navigation pane, locate the corresponding security groups, and modify their rules.)

The default security group rules of the clusters using different networks are as follows:

- [Security Group Rules of a Cluster Using a VPC Network](#)
- [Security Group Rules of a Cluster Using a Tunnel Network](#)
- [Security Group Rules of a CCE Turbo Cluster Using the Cloud Native 2.0 Network](#)

NOTICE

- Modifying or deleting security group rules may affect cluster running. Exercise caution when performing this operation. If you need to modify security group rules, do not modify the rules of the port on which CCE running depends.
- When adding a new security group rule to a cluster, ensure that the new rule does not conflict with the original rules. Otherwise, the original rules may become invalid, affecting the cluster running.

Security Group Rules of a Cluster Using a VPC Network

Security group of worker nodes

A security group named *{Cluster name}-cce-node-{Random ID}* is automatically created for worker nodes in a cluster. For details about the default ports, see [Table 7-1](#).

Table 7-1 Default ports in the security group for worker nodes that use a VPC network

Direction	Port	Default Source Address	Description	Modifiable	Modification Suggestion
Inbound rules	All UDP ports	VPC CIDR block	Allow access between worker nodes and between the worker nodes and master nodes.	No	N/A
	All TCP ports				
	All ICMP ports	Security group of master nodes	Allow master nodes to access worker nodes.	No	N/A
	TCP port range: 30000 to 32767	All IP addresses : 0.0.0.0/0	Allow access from NodePort.	Yes	These ports must permit requests from VPC, container, and load balancer CIDR blocks.
	UDP port range: 30000 to 32767				
All	Container CIDR block	Allow containers in a cluster to access nodes.	No	N/A	

Direction	Port	Default Source Address	Description	Modifiable	Modification Suggestion
	All	Security group of worker nodes	Access outside the security group of the worker nodes is restricted, but mutual access between instances in the security group of the worker nodes is not restricted.	No	N/A
	TCP port 22	All IP addresses : 0.0.0.0/0	Allow SSH access to Linux ECSs.	Recommended	N/A
Outbound rule	All	All IP addresses : 0.0.0.0/0	Allow traffic on all ports by default. You are advised to retain this setting.	Yes	If you want to harden security by allowing traffic only on specific ports, remember to allow such ports. For details, see Hardening Outbound Rules .

Security group of master nodes

A security group named *{Cluster name}-cce-control-{Random ID}* is automatically created for master nodes in a cluster. For details about the default ports, see [Table 7-2](#).

Table 7-2 Default ports in the security group for master nodes that use a VPC network

Direction	Port	Default Source Address	Description	Modifiable	Modification Suggestion
Inbound rules	TCP port 5444	VPC CIDR block	Allow access from kube-apiserver, which provides lifecycle management for Kubernetes resources.	No	N/A
	TCP port 5444	Container CIDR block			

Direction	Port	Default Source Address	Description	Modifiable	Modification Suggestion
	TCP port 9443	VPC CIDR block	Allow the network add-on of the worker nodes to access master nodes.	No	N/A
	TCP port 5443	All IP addresses : 0.0.0.0/0	Allow kube-apiserver of the master nodes to listen to the worker nodes.	Recommended	The port must allow traffic from the CIDR blocks of the VPC, the control plane of the hosted service mesh, and container.
	TCP port 8445	VPC CIDR block	Allow the storage add-on of worker nodes to access master nodes.	No	N/A
	All	Security group of master nodes	Access outside the security group of the master nodes is restricted, but mutual access between instances in the security group of the master nodes is not restricted.	No	N/A
Outbound rule	All	All IP addresses : 0.0.0.0/0	Allow traffic on all ports by default.	No	N/A

Security Group Rules of a Cluster Using a Tunnel Network

Security group of worker nodes

A security group named *{Cluster name}-cce-node-{Random ID}* is automatically created for worker nodes in a cluster. For details about the default ports, see [Table 7-3](#).

Table 7-3 Default ports in the security group for worker nodes that use a tunnel network

Dir ecti on	Port	Default Source Address	Description	Modif iable	Modification Suggestion
Inb oun d rule s	UDP port 4789	All IP addresses : 0.0.0.0/0	Allow access between containers.	No	N/A
	TCP port 10250	CIDR block of master nodes	Allow master nodes to access kubelet on worker nodes, for example, by running kubectl exec {pod} .	No	N/A
	TCP port range: 30000 to 32767	All IP addresses : 0.0.0.0/0	Allow access from NodePort.	Yes	The ports must allow traffic from the CIDR blocks of the VPC, load balancer, and container.
	UDP port range: 30000 to 32767				
	TCP port 22	All IP addresses : 0.0.0.0/0	Allow SSH access to Linux ECSs.	Reco mme nded	N/A
	All	Security group of worker nodes	Access outside the security group of the worker nodes is restricted, but mutual access between instances in the security group of the worker nodes is not restricted.	No	N/A
Out bou nd rule	All	All IP addresses : 0.0.0.0/0	Allow traffic on all ports by default. You are advised to retain this setting.	Yes	If you want to harden security by allowing traffic only on specific ports, remember to allow such ports. For details, see Hardening Outbound Rules .

Security group of master nodes

A security group named *{Cluster name}-cce-control-{Random ID}* is automatically created for master nodes in a cluster. For details about the default ports, see [Table 7-4](#).

Table 7-4 Default ports in the security group for master nodes that use a tunnel network

Direction	Port	Default Source Address	Description	Modifiable	Modification Suggestion
Inbound rules	UDP port 4789	All IP addresses : 0.0.0.0/0	Allow access between containers.	No	N/A
	TCP port 5444	VPC CIDR block	Allow access from kube-apiserver, which provides lifecycle management for Kubernetes resources.	No	N/A
	TCP port 5444	Container CIDR block			
	TCP port 9443	VPC CIDR block	Allow the network add-on of the worker nodes to access master nodes.	No	N/A
	TCP port 5443	All IP addresses : 0.0.0.0/0	Allow kube-apiserver of the master nodes to listen to the worker nodes.	Recommended	The port must allow traffic from the CIDR blocks of the VPC, the control plane of the hosted service mesh, and container.
	TCP port 8445	VPC CIDR block	Allow the storage add-on of worker nodes to access master nodes.	No	N/A

Direction	Port	Default Source Address	Description	Modifiable	Modification Suggestion
	All	Security group of master nodes	Access outside the security group of the master nodes is restricted, but mutual access between instances in the security group of the master nodes is not restricted.	No	N/A
Outbound rule	All	All IP addresses : 0.0.0.0/0	Allow traffic on all ports by default.	No	N/A

Security Group Rules of a CCE Turbo Cluster Using the Cloud Native 2.0 Network

Security group of worker nodes

A security group named *{Cluster name}-cce-node-{Random ID}* is automatically created for worker nodes in a cluster. For details about the default ports, see [Table 7-5](#).

Table 7-5 Default ports in the security group for worker nodes

Direction	Port	Default Source Address	Description	Modifiable	Modification Suggestion
Inbound rules	TCP port 10250	CIDR block of master nodes	Allow master nodes to access kubelet on worker nodes, for example, by running kubectl exec {pod} .	No	N/A
	TCP port range: 30000 to 32767	All IP addresses : 0.0.0.0/0	Allow access from NodePort.	Yes	The ports must allow traffic from the CIDR blocks of the VPC, load balancer, and container.
	UDP port range: 30000 to 32767				

Direction	Port	Default Source Address	Description	Modifiable	Modification Suggestion
	TCP port 22	All IP addresses : 0.0.0.0/0	Allow SSH access to Linux ECSs.	Recommended	N/A
	All	Security group of worker nodes	Access outside the security group of the worker nodes is restricted, but mutual access between instances in the security group of the worker nodes is not restricted.	No	N/A
	All	Container subnet CIDR block	Allow containers in a cluster to access nodes.	No	N/A
Outbound rule	All	All IP addresses : 0.0.0.0/0	Allow traffic on all ports by default. You are advised to retain this setting.	Yes	If you want to harden security by allowing traffic only on specific ports, remember to allow such ports. For details, see Hardening Outbound Rules .

Security group of master nodes

A security group named *{Cluster name}-cce-control-{Random ID}* is automatically created for master nodes in a cluster. For details about the default ports, see [Table 7-6](#).

Table 7-6 Default ports in the security group for master nodes

Direction	Port	Default Source Address	Description	Modifiable	Modification Suggestion
Inbound rules	TCP port 5444	All IP addresses : 0.0.0.0/0	Allow access from kube-apiserver, which provides lifecycle management for Kubernetes resources.	No	N/A
	TCP port 5444	VPC CIDR block		No	N/A
	TCP port 9443	VPC CIDR block	Allow the network add-on of the worker nodes to access master nodes.	No	N/A
	TCP port 5443	All IP addresses : 0.0.0.0/0	Allow kube-apiserver of the master nodes to listen to the worker nodes.	Recommended	The port must allow traffic from the CIDR blocks of the VPC, the control plane of the hosted service mesh, and container.
	TCP port 8445	VPC CIDR block	Allow the storage add-on of worker nodes to access master nodes.	No	N/A
	All	Security group of master nodes	Access outside the security group of the master nodes is restricted, but mutual access between instances in the security group of the master nodes is not restricted.	No	N/A
	All	Container subnet CIDR block	Allow traffic from all source IP addresses in the container subnet CIDR block.	No	N/A
Outbound rule	All	All IP addresses : 0.0.0.0/0	Allow traffic on all ports by default.	No	N/A

Security group of ENI

In a CCE Turbo cluster, an additional security group named *{Cluster name}-cce-eni-{Random ID}* is created. By default, containers in the cluster are bound to this security group. For details about the default ports, see [Table 7-7](#).

Table 7-7 Default ports of the ENI security group

Direction	Port	Default Source Address	Description	Modifiable	Modification Suggestion
Inbound rules	All	ENI security group	Allow containers in a cluster to access each other.	No	N/A
		VPC CIDR block	Allow instances in the cluster VPC to access containers.	No	N/A
Outbound rule	All	All IP addresses : 0.0.0.0/0	Allow traffic on all ports by default.	No	N/A

Hardening Outbound Rules

By default, all security groups created by CCE allow all the **outbound** traffic. You are advised to retain this configuration. To harden outbound rules, ensure that the ports listed in the following table are enabled.

Table 7-8 Minimum configurations of outbound security group rules for a worker node

Port	Allowed CIDR	Description
UDP port 53	DNS server of the subnet	Allow traffic on the port for domain name resolution.
UDP port 4789 (required only by clusters that use the tunnel networks)	All IP addresses	Allow access between containers.
TCP port 5443	CIDR block of master nodes	Allow kube-apiserver of the master nodes to listen to the worker nodes.

Port	Allowed CIDR	Description
TCP port 5444	CIDR blocks of the VPC and container	Allow access from kube-apiserver, which provides lifecycle management for Kubernetes resources.
TCP port 6443	CIDR block of master nodes	None
TCP port 8445	VPC CIDR block	Allow the storage add-on of worker nodes to access master nodes.
TCP port 9443	VPC CIDR block	Allow the network add-on of the worker nodes to access master nodes.
All ports	198.19.128.0/17	Allow worker nodes to access the VPC Endpoint (VPCEP) service.

7.3 Security Hardening

7.3.1 How Do I Prevent Cluster Nodes from Being Exposed to Public Networks?

Question

How do I prevent cluster nodes from being exposed to public networks?

Solution

- If access to port 22 of a cluster node is not required, you can define a security group rule that disables access to port 22.
- Do not bind an EIP to a cluster node unless necessary.

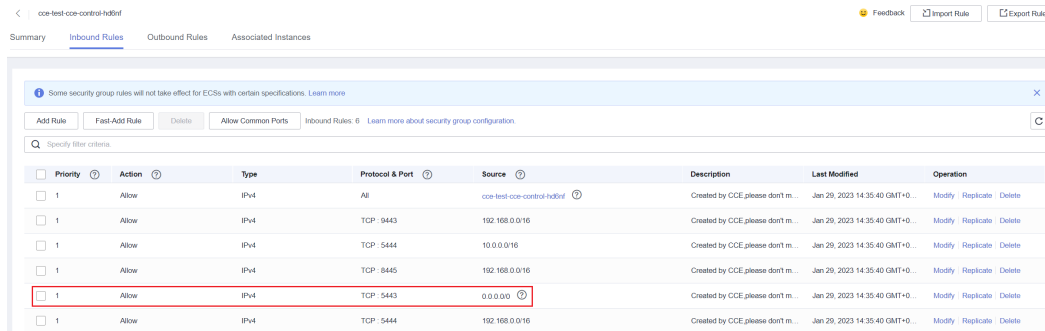
7.3.2 How Do I Configure an Access Policy for a Cluster?

After the public API Server address is bound to the cluster, modify the security group rules of port 5443 on the master node to harden the access control policy of the cluster.

- Step 1** Log in to the CCE console and click the cluster name to access the cluster console. On the **Overview** page, copy the cluster ID in the **Basic Info** area.
- Step 2** Log in to the VPC console. In the navigation pane, choose **Access Control** > **Security Groups**.
- Step 3** Select **Description** as the filter criterion and paste the cluster ID to search for the target security group.

Step 4 Locate the row that contains the security group (starting with *{CCE cluster name}-cce-control*) of the master node and click **Manage Rules** in the **Operation** column.

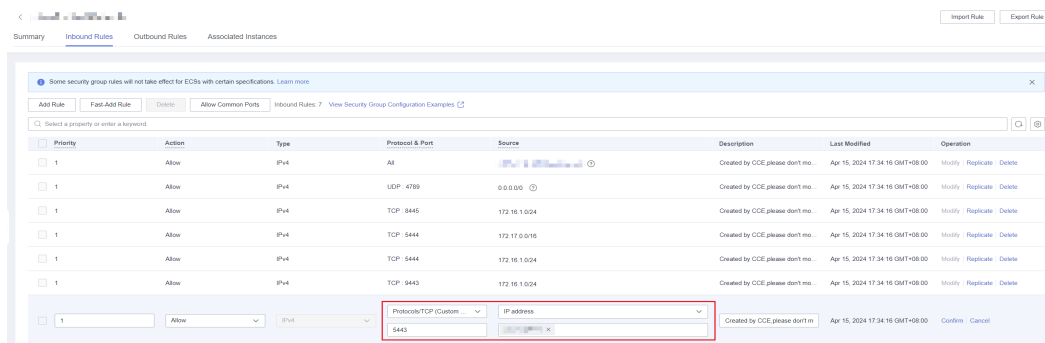
Step 5 On the page displayed, locate the row that contains port 5443 and click **Modify** in the **Operation** column to modify its inbound rules.



Step 6 Change the source IP address that can be accessed as required. For example, if the IP address used by the client to access the API Server is **100.*.***, you can add an inbound rule for port 5443 and set the source IP address to **100.*.***.

NOTE

In addition to the client IP address, the port must allow traffic from the CIDR blocks of the VPC, container, and the control plane of the hosted service mesh to ensure that the API Server can be accessed from within the cluster.



Step 7 Click **Confirm**.

----End

7.3.3 How Do I Change the Security Group of Nodes in a Cluster in Batches?

Notes and Constraints

Do not add more than 1000 instances to the same security group. Otherwise, the security group performance may deteriorate.

Procedure

Step 1 Log in to the VPC console and select the desired region and project in the upper left corner.

Step 2 In the navigation pane on the left, choose **Access Control > Security Groups**.

Step 3 On the **Security Groups** page, click **Manage Instance** in the **Operation** column.

Step 4 On the **Servers** tab, click **Add**.

Step 5 Select the servers to be added to the security group and click **OK**. You can also search for servers by name, ID, private IP address, status, enterprise project, or tag.

You can change the maximum number of servers displayed on a page in the lower left corner to add a maximum of 20 servers to a security group at a time.

 **NOTE**

After the node is added to a new security group, the original security group is retained. To remove the instance, click **Manage Instance** of the original security group and select the node servers to be removed.

----End

7.4 Network Configuration

7.4.1 How Can Container IP Addresses Survive a Container Restart?

If Containers Will Run in a Single-Node Cluster

Add **hostNetwork: true** to the **spec.spec** in the YAML file of the workload to which the containers will belong.

If Containers Will Run in a Multi-Node Cluster

Configure node affinity policies, in addition to perform the operations described in "If the Container Runs in a Single-Node Cluster". However, after the workload is created, the number of running pods cannot exceed the number of affinity nodes.

Expected Result

After the previous settings are complete and the workload is running, the IP addresses of the workload's pods are the same as the node IP addresses. After the workload is restarted, these IP addresses will keep unchanged.

7.4.2 How Can I Check Whether an ENI Is Used by a Cluster?

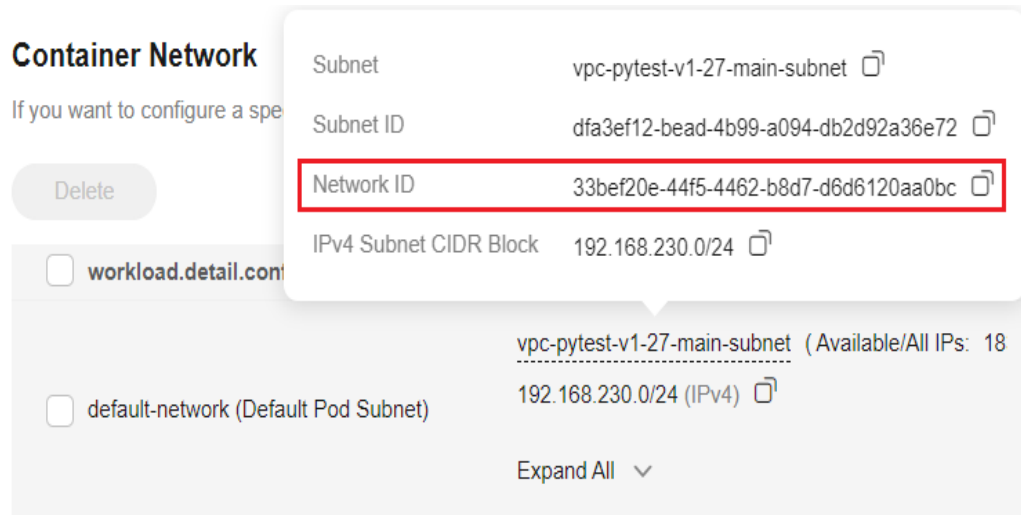
Scenarios

Pod subnets can be deleted from CCE Turbo clusters of v1.23.17-r0, v1.25.12-r0, v1.27.9-r0, v1.28.7-r0, v1.29.3-r0, or later versions.

Deleting a pod subnet from a cluster can be risky. It is important to ensure that none of the ENIs currently in use by the cluster belong to the subnet, including those being used by pods and pre-bound to pods.

Procedure

- Step 1** Log in to the CCE console and click the cluster name to access the cluster console.
- Step 2** In the navigation pane, choose **Settings** and click the **Network** tab.
- Step 3** In the **Container Network** area, copy the network ID of the subnet. (The default-network is used as an example.)



- Step 4** Log in to the VPC console. In the navigation pane, choose **Virtual Private Cloud** > **Subnets**. In the right pane, obtain the target subnet based on the network ID.
- Step 5** In the **Resources** area, locate **Network Interfaces** and click the number next to it. On the page displayed, check the network interfaces and supplementary network interfaces of the subnet.
- Step 6** Check the names or descriptions of the network interfaces. If the name or description of a network interface contains the ID of the cluster, it indicates that the network interface is used by the cluster. You can obtain the cluster ID on the **Overview** page of the CCE console.

To delete the subnet ENIs used in the cluster, submit a service ticket.

----End

7.4.3 How Can I Delete a Security Group Rule Associated with a Deleted Subnet?

Scenarios

Pod subnets can be deleted from CCE Turbo clusters of v1.23.17-r0, v1.25.12-r0, v1.28.7-r0, or later versions.

When you delete a subnet, CCE does not automatically remove the security group rules associated with the subnet in the default node security group created by CCE. You must manually delete these rules.

Procedure

- Step 1** Log in to the CCE console and click the cluster name to access the cluster console.
- Step 2** In the navigation pane, choose **Settings** and click the **Network** tab.
- Step 3** In the **Container Network** area, copy the IPv4 CIDR block of the subnet. (The default-network is used as an example.)
- Step 4** In the navigation pane, choose **Overview**. In the **Networking Configuration** area, click the name of the default node security group.
- Step 5** On the page displayed, click the **Inbound Rules** tab, locate the row containing the subnet CIDR block based on the source IP address, and find the corresponding security group rule.
- Step 6** Click **Delete** in the **Operation** column.

----End

7.4.4 How Can I Synchronize Certificates When Multiple Ingresses in Different Namespaces Share a Listener?

Context

In a cluster, multiple ingresses can share the same listener, allowing them to use the same port on a single load balancer. When two ingresses are set up with HTTPS certificates, the server certificate that is used will be based on the configuration of the earliest ingress.

If ingresses in separate namespaces use the same listener and TLS certificates, due to namespace isolation, the secrets associated with the TLS certificates may not display normally for the ingress that was created later.

The following table shows an example for the configurations of two ingresses.

Ingress Name	ingress1	ingress2
Namespace	namespace1	namespace2
Creation Time	2024-04-01	2024-04-02
Protocol	HTTPS	HTTPS
Load Balancer	elb1	elb1
Port	443	443
Certificate Source	TLS key	TLS key
Secret Corresponding to the TLS Secret	namespace1/secret1	namespace2/secret2
Valid Certificate	namespace1/secret1	namespace1/secret1

Symptom

Within a given cluster, ingress1 and ingress2 are created in namespace1 and namespace2, respectively. Both ingresses connect to the same listener and use TLS certificates.

Ingress1's certificate is used because ingress1 was created first. But, ingress2 cannot read the configuration of secret1 because it is in a different namespace than namespace1. As a result, the configuration page of ingress2 will display the following information.

Update Ingress

Name: ingress2

Namespace: namespace2

Load Balancer: testnotdel [↗](#)
 I have read [Notes on Using Load Balancers.](#)

Listener

External Protocol	HTTPS
External Port	443
Access Control	Not configured v
Certificate Source	TLS secret
Server Certificate	The server certificate has been configured for the listener and cannot be modified. The secret ID of the certificate is 40a0773d-b874-4739-a490-3565f1ffa689. Listener Details ↗

Solution

Each load balancer certificate has a corresponding TLS key, and the key content is identical. The CCE agency permissions enable access to certificate information without namespace restrictions. This means that you can switch the certificate source of ingress1 to the server certificate and assign the load balancer certificate corresponding to the TLS key. The configuration modification page of ingress2 displays the server certificate that works.

- Step 1** Log in to the CCE console and click the cluster name to access the cluster console.
- Step 2** In the navigation pane, choose **Services & Ingresses**, click the **Ingresses** tab, and click the load balancer link of ingress1 to go to the ELB console.
- Step 3** Click the **Listeners** tab, find the listener based on the port configured for ingress1, and click the listener name to go to the details page.
- Step 4** On the page displayed, find and record the server certificate.

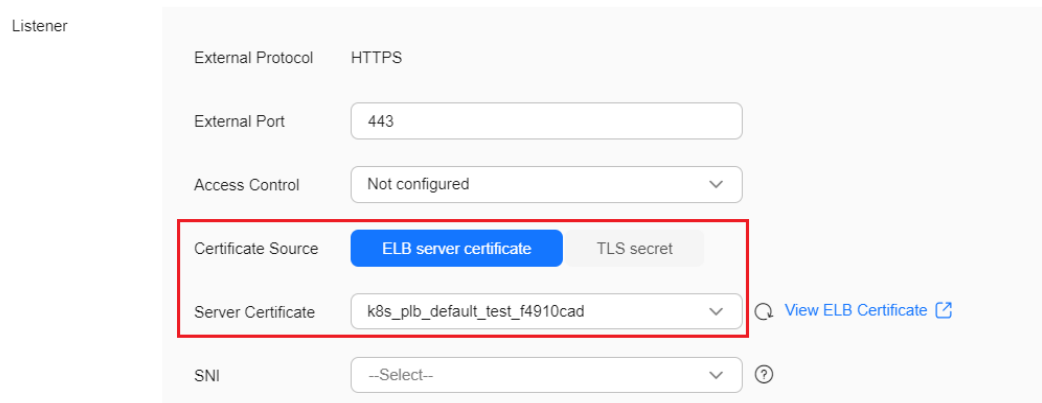
Elastic Load Balancer / Load balancer [redacted] / Listener (k8s_HTTPS_443) [Switch Listen...](#)

Summary Forwarding Policies Monitoring Tags

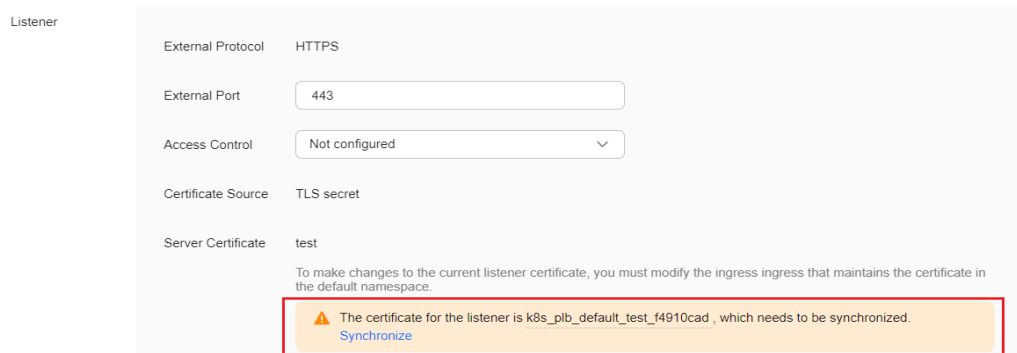
Basic Information		ID	621992b2-2a47-408f-94a7-cb43d85179 ↗
Name	k8s_HTTPS_443 ↗	Server CertificateID	40a0773d-b874-4739-a490-3565f1ffa689 ↗
Frontend Protocol/Port	HTTPS/443	CA CertificateID	--/-- Configure
Access Control	All IP addresses Configure	SNI	-- Configure
Transfer Client IP Address	Enabled ⊙	Advanced Forwarding	Enable ⊙
Created	May 27, 2024 10:41:34 GMT+08:00	Modification Protection	Disabled Configure
Description	{"attention": "Auto-generated by CCE service, do not modify!", "cluster_id": "26200ac9-18dd-11ef-b13c-0255ac100b0e", "secret_id": "40a0773d-b874-4739-a490-3565f1ffa689"} ↗		

Step 5 Go back to the CCE console. On the **Ingresses** tab, locate the row containing ingress1 and choose **More > Update** in the **Operation** column. In the window that slides out from the right, set **Certificate Source** to **ELB server certificate**, select the server certificate obtained in the previous step, and click **OK**.

The certificate source of ingress1 has been changed from the TLS key to the server certificate, but the key content remains the same, as does the configuration that is applied.



Step 6 Switch to namespace2. On the **Ingresses** tab, locate the row containing ingress2 and choose **More > Update** in the **Operation** column. In the window that slides out from the right, locate the **Server Certificate** parameter in the **Listener** area, click **Synchronize**, and click **OK**.



Step 7 Verify that the configuration of ingress2 is displayed properly after the update is complete.

----End

7.4.5 How Can I Determine Which Ingress the Listener Settings Have Been Applied To?

With CCE, you can associate multiple ingresses with a single load balancer listener and establish various forwarding policies. Listener configuration parameters are stored in annotations, which means that a listener can have different configuration parameters on different ingresses. This section describes how to determine which ingress the listener settings are applied to. It covers:

- Obtaining the first ingress
- Configuring and updating a listener certificate
- Impacts of deleting the first ingress on listener settings and configuration synchronization

Obtaining the First Ingress

Listener parameters can be configured for all ingresses associated with the same listener, so CCE uses the listener annotation configurations (excluding SNI certificates) from the earliest created ingress (the first ingress). The first ingress is determined by sorting the **metadata.createTimestamp** fields of the ingresses in ascending order.

- The first ingress information is written into annotations in clusters of v1.21.15-r0, v1.23.14-r0, v1.25.9-r0, v1.27.6-r0, v1.28.4-r0, v1.29.1-r0, and later. You can check **kubernetes.io/elb.listener-master-ingress** in the annotations of the existing ingresses.

```

1  apiVersion: networking.k8s.io/v1
2  kind: Ingress
3  metadata:
4    name: ingress-first
5    namespace: default
6    uid: 43b57afc-7f55-4310-ac1a-ac8afdd3d5fd
7    resourceVersion: '1558102'
8    generation: 1
9    creationTimestamp: '2024-09-09T02:31:07Z'
10  annotations:
11    kubernetes.io/elb.class: performance
12    kubernetes.io/elb.id: be56202a-c2cb-40d5-900e-d7a007a4b054
13    kubernetes.io/elb.listener-master-ingress: default/ingress-first
14    kubernetes.io/elb.port: 443
15    kubernetes.io/elb.tls-certificate-ids: 87e311e965db421ca806c151368c01ca,8f47921346e74aa58ba38660127e5967
16    kubernetes.io/elb.tls-ciphers-policy: tls-1-2-fs
17  managedFields:

```

- To get the first ingress in clusters earlier than v1.21.15-r0, v1.23.14-r0, v1.25.9-r0, v1.27.6-r0, v1.28.4-r0, and v1.29.1-r0, use the kubectl command to obtain the ingresses associated with the same load balancer listener, sort these ingresses in ascending order, based on their creation time, and check the first one.

The query command is as follows: (Replace the load balancer ID and port number as needed.)

```

elb_id=${1}
elb_port=${2}
kubectl get ingress --all-namespaces --sort-by='metadata.creationTimestamp' -o=custom-
columns=Name:'metadata.name',Namespace:'metadata.namespace',elbID:'metadata.annotations.kuber
netes.io/elb.id',elbPort:'metadata.annotations.kubernetes.io/elb
\port',elbPorts:'metadata.annotations.kubernetes.io/elb.listen-ports' | awk 'NR==1 {print; next} {if
($5 != "<none>") $4 = "<none>"; print}' | grep -E "^Name|${elb_id}" | grep -E "^Name|${elb_port}" |
awk '{printf "%-30s %-30s %-38s %-10s %-10s\n", $1, $2, $3, $4, $5}'

```

In the command output, the first column specifies the ingress names, the second specifies the namespaces, the third specifies the load balancer IDs, the fourth specifies the listener ports, and the fifth specifies the ports of multiple listeners. (If multiple listener port numbers are configured, they will replace the listener port numbers and become effective.)

Name	Namespace	elbID	elbPort	elbPorts
ingress-first	default	be56202a-c2cb-40d5-900e-d7a007a4b054	443	<none>
ingress-second	default	be56202a-c2cb-40d5-900e-d7a007a4b054	443	<none>
ingress-third	test	be56202a-c2cb-40d5-900e-d7a007a4b054	443	<none>

Configuring and Updating a Listener Certificate

You can configure an ingress certificate in a cluster using either of the following methods:

- Use a TLS certificate, keep it as a secret, and manage it on CCE. TLS certificates are configured using the `spec.tls` fields in ingresses. They will be automatically created, updated, or deleted via the ELB console.
- Use a certificate created in the ELB service. The certificate content is maintained on the ELB console. Such certificates are configured in the `annotation` fields in ingresses.

ELB server certificates are also maintained on the ELB console, so you do not need to import the certificate content to CCE secrets. This allows for unified cross-namespace configuration. It is recommended that you use ELB server certificates to configure the certificates for ingresses.

ELB server certificates are supported in clusters of versions v1.19.16-r2, v1.21.5-r0, and v1.23.3-r0.

To update the listener server certificate created by an ingress using a TLS certificate, follow these steps:

1. Check the command in [Obtaining the First Ingress](#) and obtain all ingresses associated with the same listener.

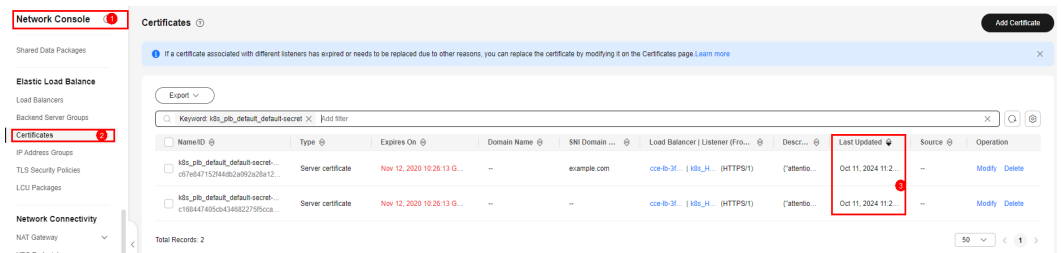
Name	Namespace	elbID	elbPort	elbPorts
ingress-first	default	be56202a-c2cb-40d5-900e-d7a007a4b054	443	<none>
ingress-second	default	be56202a-c2cb-40d5-900e-d7a007a4b054	443	<none>
ingress-third	test	be56202a-c2cb-40d5-900e-d7a007a4b054	443	<none>

2. Obtain the certificate configuration in the first ingress.

```
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: ingress-first
  namespace: default
...
spec:
  tls:
    - secretName: default-ns-secret-1
    - hosts:
      - 'example.com'
      secretName: default-ns-secret-2
...

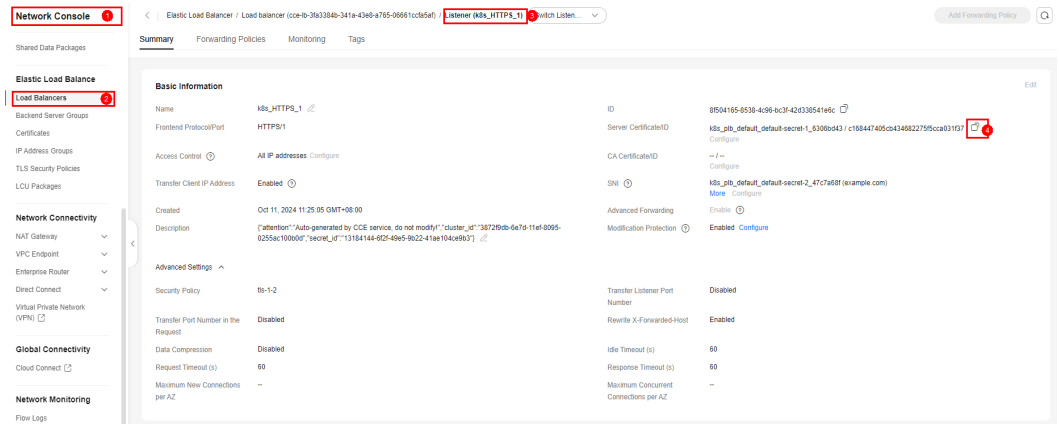
```

3. Update the configurations of `default-ns-secret-1` and `default-ns-secret-2` in the first ingress.
4. After the keys are updated, check whether the server certificate has been updated on the ELB console.

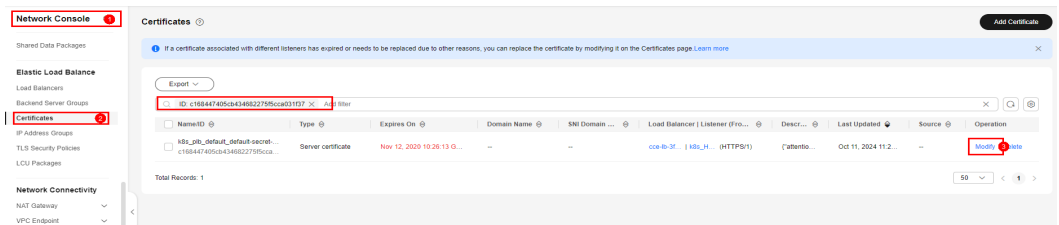


To update the listener server certificate created by an ingress using an ELB certificate, follow these steps:

1. On the ELB console, obtain the ID of the server certificate used by the load balancer listener.



2. Modify the server certificate.



Impacts of Deleting the First Ingress on Listener Settings

Listener settings only apply to the first ingress configuration (excluding SNI certificates). If the first ingress is deleted, the earliest created ingress in use becomes the new first ingress, and the listener settings will be updated accordingly. This means that if the listener settings of the old and new first ingresses are different, there may be unexpected updates on the ELB console. To avoid this, check if the listener configuration of the ingress that will become the new first ingress is the same as the one for the original first ingress, or otherwise meets your expectations.

- You can synchronize the listener settings on the console. The procedure is as follows:
 - a. Log in to the CCE console and click the cluster name to access the cluster console.
 - b. In the navigation pane, choose **Services**, click the **Ingresses** tab, and choose **More > Update** in the **Operation** column.
 - c. Click **Synchronize** to automatically synchronize the server certificate. This option is available when the listener settings of the ingress are inconsistent with those of the load balancer.

NOTE

If you synchronize a server certificate and an SNI certificate, and the current ingress is using a **TLS key**, the certificate will be replaced with an ELB server certificate. If the cluster version is earlier than v1.19.16-r2, v1.21.5-r0, v1.23.3-r0 and does not support ELB server certificates, you need to **manually synchronize the configurations using YAML**.

Update Ingress

Name ingress-second

Namespace default

Load Balancer [cce-lb-3fa3384b-341a-43e8-a765-06661ccfa5af](#)

I have read [Notes on Using Load Balancers](#).

Listener

Frontend Protocol HTTPS

External Port 445

⚠ The listener configuration is maintained by the first route ingress-first in the default namespace. However, the current configuration is different from that in the first route. You need to manually synchronize it with the first route configuration. [Synchronize](#)

Access Control Inherit ELB Configurations

Certificate Source ELB server certificate

Server Certificate [example-certificate-2](#) [View ELB Certificate](#)

SNI --Select--

Security Policy Security Policy TLS-1-2

- d. Click **OK** to apply the modification.
- You can manually synchronize the listener settings using YAML. The procedure is as follows:

- a. Check the command in **Obtaining the First Ingress** and obtain all ingresses associated with the same listener.

Name	Namespace	elbID	elbPort	elbPorts
ingress-first	default	be56202a-c2cb-40d5-900e-d7a007a4b054	443	<none>
ingress-second	default	be56202a-c2cb-40d5-900e-d7a007a4b054	443	<none>
ingress-third	test	be56202a-c2cb-40d5-900e-d7a007a4b054	443	<none>

- b. Before deleting some ingresses such as **ingress-first** and **ingress-second**, synchronize the listener settings of **ingress-first** to the annotations of **ingress-third**.

If the listener server certificate was created using a TLS key, you need to synchronize the configurations saved in the ingress' **spec.tls** to **ingress-third**.

```
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: ingress-first
  namespace: default
  ...
spec:
  tls:
    - secretName: default-ns-secret-1
    - hosts:
      - 'example.com'
      secretName: default-ns-secret-2
  ...
```

8 Storage

8.1 How Do I Expand the Storage Capacity of a Container?

Application Scenarios

The default storage size of a container is 10 GiB. If a large volume of data is generated in the container, expand the capacity using the method described in this topic.

Solution

- Step 1** Log in to the CCE console and click the cluster name to access the cluster console.
- Step 2** Choose **Nodes** from the navigation pane.
- Step 3** Click the **Nodes** tab, locate the row containing the target node, and choose **More** > **Reset Node** in the **Operation** column.

NOTICE

Resetting a node may make the node-specific resources (such as local storage and workloads scheduled to this node) unavailable. Exercise caution when performing this operation to avoid impact on running services.

- Step 4** Reconfigure node parameters.

If you need to adjust the container storage space, pay attention to the following configurations:

Storage Settings: Click **Expand** next to the data disk to set the following parameter:

Space Allocation for Pods: indicates the base size of a pod. It is the maximum size that a workload's pods (including the container images) can grow to in the disk space. Proper settings can prevent pods from taking all the disk space

available and avoid service exceptions. It is recommended that the value is less than or equal to 80% of the container engine space. This parameter is related to the node OS and container storage rootfs and is not supported in some scenarios.

Step 5 After the node is reset, log in to the node and check whether the container capacity has been expanded. The command output varies with the container storage rootfs.

- **Overlayfs:** No independent thin pool is allocated. Image data is stored in **dockersys**. Run the following command to check whether the container capacity has been expanded:

docker exec -it container_id /bin/sh or **kubectrl exec -it container_id /bin/sh**
df -h

If the information similar to the following is displayed, the overlay capacity has been expanded from 10 GiB to 15 GiB.

```
Filesystem      Size  Used Avail Use% Mounted on
overlay         15G  104K  15G   1% /
tmpfs           64M   0   64M   0% /dev
tmpfs           3.6G   0   3.6G   0% /sys/fs/cgroup
/dev/mapper/vgpaas-share 98G  4.0G  89G   5% /etc/hosts
...
```

- **Devicemapper:** A thin pool is allocated to store image data. Run the following command to check whether the container capacity has been expanded:

docker exec -it container_id /bin/sh or **kubectrl exec -it container_id /bin/sh**
df -h

If the information similar to the following is displayed, the thin pool capacity has been expanded from 10 GiB to 15 GiB.

```
Filesystem      Size  Used Avail Use% Mounted on
/dev/mapper/vgpaas-thinpool-snap-84 15G  232M  15G   2% /
tmpfs           64M   0   64M   0% /dev
tmpfs           3.6G   0   3.6G   0% /sys/fs/cgroup
/dev/mapper/vgpaas-kubernetes  11G  41M  11G   1% /etc/hosts
/dev/mapper/vgpaas-dockersys  20G  1.1G  18G   6% /etc/hostname
...
```

----End

8.2 What Are the Differences Among CCE Storage Classes in Terms of Persistent Storage and Multi-Node Mounting?

Container storage provides storage for container workloads. It supports multiple storage classes. A pod can use any amount of storage.

Currently, CCE supports local, EVS, SFS, SFS Turbo, and OBS volumes.

The following table lists the differences among these storage classes.

Table 8-1 Differences among storage classes

Storage Class	Persistent Storage	Automatic Migration with Containers	Multi-Node Mounting
Local disks	Supported	Not supported	Not supported
EVS	Supported	Supported	Not supported
OBS	Supported	Supported	Supported. This type of volumes can be shared among multiple nodes or workloads.
SFS Turbo	Supported	Supported	Supported. This type of volumes can be shared among multiple nodes or workloads.

Selecting a Storage Class

You can use the following types of storage volumes when creating a workload. You are advised to store workload data on EVS volumes. If you store workload data on a local volume, the data cannot be restored when a fault occurs on the node.

- **Local volumes:** Mount the file directory of the host where a container is located to a specified container path (corresponding to `hostPath` in Kubernetes). Alternatively, you can leave the source path empty (corresponding to `emptyDir` in Kubernetes). If the source path is left empty, a temporary directory of the host will be mounted to the mount point of the container. A specified source path is used when data needs to be persistently stored on the host, while `emptyDir` is used when temporary storage is needed. A `ConfigMap` is a type of resource that stores configuration data required by a workload. Its contents are user-defined. A `Secret` is an object that contains sensitive data such as workload authentication information and keys. Information stored in a `Secret` is determined by users.
- **EVS volumes:** Mount an EVS volume to a container path. When the container is migrated, the mounted EVS volume is migrated together. This storage class is applicable when data needs to be stored permanently.
- **OBS volumes:** Create OBS volumes and mount them to a container path. OBS volumes are applicable to scenarios such as cloud workload, data analysis, content analysis, and hotspot objects.
- **SFS Turbo volumes:** Create SFS Turbo volumes and mount them to a container path. SFS Turbo volumes are fast, on-demand, and scalable, which makes them suitable for DevOps, containerized microservices, and enterprise office applications.

8.3 Can I Create a CCE Node Without Adding a Data Disk to the Node?

No. A data disk is mandatory.

A data disk dedicated for kubelet and the container engine will be attached to a new node. By default, CCE uses LVM to manage data disks. With LVM, you can adjust the disk space ratio for different resources on a data disk.

If the data disk is uninstalled or damaged, the container engine will malfunction and the node becomes unavailable.

8.4 What Should I Do If the Host Cannot Be Found When Files Need to Be Uploaded to OBS During the Access to the CCE Service from a Public Network?

When a Service deployed on CCE attempts to upload files to OBS after receiving an access request from an offline machine, an error message is displayed, indicating that the host cannot be found. The following figure shows the error message.

Time	message
February 22nd 2020, 18:50:27.521	com.obs.services.exception.ObsException: OBS service Error Message. Request Error : java.net.UnknownHostException: obs.
February 22nd 2020, 18:50:27.521	18:50:27.520 [XNIO-1 task-16] ERROR c.h.f.c.provider.ExceptionProvider - OBS service Error Message. Request Error : java.net.UnknownHostException: obs.
February 22nd 2020, 18:50:27.298	18:50:27.298 [XNIO-1 task-9] ERROR c.h.f.c.provider.ExceptionProvider - OBS service Error Message. Request Error : java.net.UnknownHostException: obs.
February 22nd 2020, 18:50:27.298	com.obs.services.exception.ObsException: OBS service Error Message. Request Error : java.net.UnknownHostException: obs.
February 22nd 2020, 18:50:27.275	18:50:27.274 [XNIO-1 task-9] WARN c.o.s.internal.RestStorageService - com.obs.services.internal.ServiceException: Request Error : java.net.UnknownHostException: obs. HEAD 'https://obs. /obs-it-problem-management-media-test?apiversion' on Host 'obs.'
February 22nd 2020, 18:50:27.275	com.obs.services.internal.ServiceException: Request Error : java.net.UnknownHostException: obs.
February 22nd 2020, 18:50:27.275	2020-02-22 18:50:27 274 com.obs.services.internal.RestStorageService handleThrowable 205 com.obs.se rvices.internal.ServiceException: Request Error : java.net.UnknownHostException:

Fault Locating

After receiving the HTTP request, the Service transfers files to OBS through the proxy.

If too many files are transferred, a large number of resources are consumed. Currently, the proxy is assigned 128 MiB of memory. According to pressure test results, resource consumption is large, resulting in request failure.

The test results show that all traffic passes through the proxy. Therefore, if the service volume is large, more resources need to be allocated.

Solution

1. File transfer involves a large number of packet copies, which occupies a large amount of memory. In this case, increase the proxy memory based on the actual scenario and then try to access the Service and upload files again.
2. Additionally, remove the Service from the mesh because the proxy only forwards packets and does not perform any other operations. If requests pass through the ingress gateway, the grayscale release function of the Service is not affected.

8.5 How Can I Achieve Compatibility Between ExtendPathMode and Kubernetes client-go?

Application Scenarios

The Kubernetes pod structure does not contain **ExtendPathMode**. Therefore, when a user calls the API for creating a pod or deployment by using client-go, the created pod does not contain **ExtendPathMode**. CCE provides a solution to ensure compatibility with the Kubernetes client-go.

Solution

NOTICE

- When creating a pod, you need to add **kubernetes.io/extend-path-mode** to **annotation** of the pod.
- When creating a Deployment, you need to add **kubernetes.io/extend-path-mode** to **kubernetes.io/extend-path-mode** in the template.

The following is an example YAML of creating a pod. After the **kubernetes.io/extend-path-mode** keyword is added to **annotation**, the **containername**, **name**, and **mountpath** fields are matched, and the corresponding **extendpathmode** is added to **volumeMount**.

```
apiVersion: v1
kind: Pod
metadata:
  name: test-8b59d5884-96vdz
  generateName: test-8b59d5884-
  namespace: default
  selfLink: /api/v1/namespaces/default/pods/test-8b59d5884-96vdz
  labels:
    app: test
    pod-template-hash: 8b59d5884
  annotations:
    kubernetes.io/extend-path-mode:
'["{"containername":"container-0","name":"vol-156738843032165499","mountpath":"/
tmp","extendpathmode":"PodUID"}']
  metrics.alpha.kubernetes.io/custom-endpoints: '[{"api":"","path":"","port":"","names":""}]'
ownerReferences:
```

```

- apiVersion: apps/v1
  kind: ReplicaSet
  name: test-8b59d5884
  uid: 2633020b-cd23-11e9-8f83-fa163e592534
  controller: true
  blockOwnerDeletion: true
spec:
  volumes:
  - name: vol-156738843032165499
    hostPath:
      path: /tmp
      type: ""
  - name: default-token-4s959
    secret:
      secretName: default-token-4s959
      defaultMode: 420
  containers:
  - name: container-0
    image: 'nginx:latest'
    env:
    - name: PAAS_APP_NAME
      value: test
    - name: PAAS_NAMESPACE
      value: default
    - name: PAAS_PROJECT_ID
      value: b6315dd3d0ff4be5b31a963256794989
  resources:
    limits:
      cpu: 250m
      memory: 512Mi
    requests:
      cpu: 250m
      memory: 512Mi
  volumeMounts:
  - name: vol-156738843032165499
    mountPath: /tmp
    extendPathMode: PodUID
  - name: default-token-4s959
    readOnly: true
    mountPath: /var/run/secrets/kubernetes.io/serviceaccount
  terminationMessagePath: /dev/termination-log
  terminationMessagePolicy: File
  imagePullPolicy: Always
  restartPolicy: Always
  terminationGracePeriodSeconds: 30
  dnsPolicy: ClusterFirst
  serviceAccountName: default
  serviceAccount: default
  nodeName: 192.168.0.24
  securityContext: {}
  imagePullSecrets:
  - name: default-secret
  - name: default-secret
  affinity: {}
  schedulerName: default-scheduler
  tolerations:
  - key: node.kubernetes.io/not-ready
    operator: Exists
    effect: NoExecute
    tolerationSeconds: 300
  - key: node.kubernetes.io/unreachable
    operator: Exists
    effect: NoExecute
    tolerationSeconds: 300
  priority: 0
  dnsConfig:
    options:
    - name: timeout
      value: ""

```

```
- name: ndots
  value: '5'
- name: single-request-reopen
  enableServiceLinks: true
```

Table 8-2 Descriptions of key parameters

Parameter	Type	Description
containername	String	Name of a container.
name	String	Name of a volume.
mountpath	String	Mount path.
extendpathmode	String	<p>A third-level directory is added to the created volume directory/subdirectory to facilitate the obtaining of a single pod output file.</p> <p>The following types are supported.</p> <ul style="list-style-type: none"> ● None: The extended path is not configured. ● PodUID: ID of a pod. ● PodName: Name of a pod. ● PodUID/ContainerName: ID of a pod or name of a container. ● PodName/ContainerName: Name of a pod or container.

8.6 Can CCE PVCs Detect Underlying Storage Faults?

CCE PersistentVolumeClaims (PVCs) are implemented as they are in Kubernetes. A PVC is defined as a storage declaration and is decoupled from underlying storage. It is not responsible for detecting underlying storage details. Therefore, CCE PVCs cannot detect underlying storage faults.

Cloud Eye allows users to view cloud service metrics. These metrics are built-in based on cloud service attributes. After users enable a cloud service on the cloud platform, Cloud Eye automatically associates its built-in metrics. Users can track the cloud service status by monitoring these metrics.

It is recommended that users who have storage fault detection requirements use Cloud Eye to monitor underlying storage and send alarm notifications.

8.7 What Should I Do If a Yearly/Monthly EVS Disk Cannot Be Automatically Created?

Symptom

When creating a yearly/monthly EVS disk, the payment permission cannot be added to **cce_cluster_agency**.

NOTE

To dynamically create yearly/monthly EVS disks, your cluster version must be v1.23.14-r0, v1.25.9-r0, v1.27.6-r0, v1.28.4-r0, or later. Additionally, you will need to have the Everest add-on 2.4.16 or later installed in the cluster.

Possible Causes

cce_cluster_agency is the system agency of CCE. It contains the cloud service resource operation permissions required by CCE components, but does not include the payment permission. When creating yearly/monthly EVS disks, **cce_cluster_agency** must have the payment permissions, so you must manually add the **bss:order:pay** permission to **cce_cluster_agency**.

Solution

You can create a custom policy, add the **bss:order:pay** permission to it, and grant the policy to **cce_cluster_agency**.

Step 1 Create a custom policy.

1. Log in to the IAM console. In the navigation pane, choose **Permissions > Policies/Roles**. Then click **Create Custom Policy**.
2. Configure parameters for the policy.
 - **Policy Name:** Set it to **CCE Subscribe Operator**.
 - **Policy View:** Select **JSON**.
 - **Policy Content:** Configure it as follows:

```
{
  "Version": "1.1",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "bss:order:pay"
      ]
    }
  ]
}
```

3. Click **OK**.

Step 2 Grant the custom policy to **cce_cluster_agency**.

1. Log in to the IAM console. In the navigation pane, choose **Agencies**.
2. Locate the agency named **cce_cluster_agency** and click **Authorize**.

3. Search for the **CCE Subscribe Operator** custom policy, select it, and click **Next**.
4. Select an authorization scope as needed.
By default, **All resources** is selected.
5. Click **OK**.

Step 3 Go back to the CCE console, create a yearly/monthly EVS disk again, and verify that this problem has been resolved.

----End

9 Namespace

9.1 What Should I Do If a Namespace Fails to Be Deleted Due to an APIService Object Access Failure?

Symptom

The namespace remains in the **Deleting** state. The error message "DiscoveryFailed" is displayed in **status** in the YAML file.

```
76 status:
77   phase: Terminating
78   conditions:
79     - type: NamespaceDeletionDiscoveryFailure
80       status: 'True'
81       lastTransitionTime: '2022-07-04T13:44:55Z'
82       reason: DiscoveryFailed
83       message: 'Discovery failed for some groups, 1 failing: unable to retrieve the complete list of server
84 APIs: metrics.k8s.io/v1beta1: the server is currently unable to handle the request'
85     - type: NamespaceDeletionGroupVersionParsingFailure
86       status: 'False'
```

In the preceding figure, the full error message is "Discovery failed for some groups, 1 failing: unable to retrieve the complete list of server APIs: metrics.k8s.io/v1beta1: the server is currently unable to handle the request".

This indicates that the namespace deletion is blocked when kube-apiserver accesses the APIService resource object of the metrics.k8s.io/v1beta1 API.

Possible Causes

If an APIService object exists in the cluster, deleting the namespace will first access the APIService object. If the access fails, the namespace deletion will be blocked. In addition to the APIService objects created by users, add-ons like metrics-server and prometheus in the CCE cluster automatically create APIService objects.

NOTE

For details, see <https://kubernetes.io/docs/concepts/extend-kubernetes/api-extension/apiserver-aggregation/>.

Solution

Use either of the following methods:

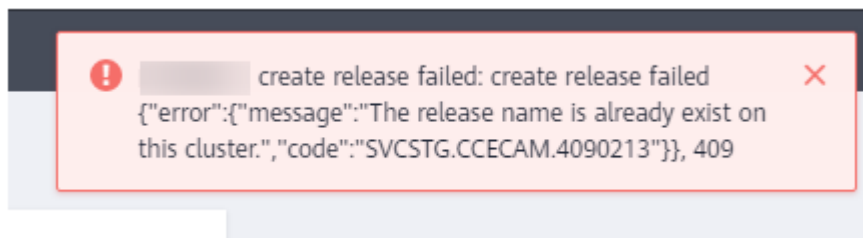
- Rectify the APIService object in the error message. If the object is created by an add-on, ensure that the pod where the add-on locates is running properly.
- Delete the APIService object in the error message. If the object is created by an add-on, uninstall the add-on.

10 Chart and Add-on

10.1 What Should I Do If Installation of an Add-on Fails and "The release name is already exist" Is Displayed?

Symptom

When an add-on fails to be installed, the error message "The release name is already exist" is returned.



Possible Causes

The add-on release record remains in the Kubernetes cluster. Generally, it is because the cluster etcd has backed up and restored the add-on, or the add-on fails to be installed or deleted.

Solution

Use `kubectl` to connect to the cluster and manually clear the Secret and ConfigMap corresponding to the add-on release. The following uses autoscaler add-on release as an example.

- Step 1** Connect to the cluster using `kubectl`, and run the following command to view the Secret list of add-on releases:

```
kubectl get secret -A |grep cceaddon
```

```
[root@cce-123-vpc-node2 ~]# kubectl get secret -nkube-system |grep cceaddon
sh.helm.release.v1.cceaddon-autoscaler.v1    helm.sh/release.v1    1    61s
sh.helm.release.v1.cceaddon-autoscaler.v2    helm.sh/release.v1    1    47s
sh.helm.release.v1.cceaddon-coredns.v1      helm.sh/release.v1    1    6h2m
sh.helm.release.v1.cceaddon-everest.v1      helm.sh/release.v1    1    6h2m
[root@cce-123-vpc-node2 ~]#
```

The Secret name of an add-on release is in the format of **sh.helm.release.v1.cceaddon-*{add-on name}*.v***. If there are multiple release versions, you can delete their Secrets at the same time.

Step 2 Run the **release secret** command to delete the Secrets.

Example:

**kubectl delete secret sh.helm.release.v1.cceaddon-autoscaler.v1
sh.helm.release.v1.cceaddon-autoscaler.v2 -nkube-system**

```
[root@cce-123-vpc-node2 ~]# kubectl delete secret sh.helm.release.v1.cceaddon-autoscaler.v1 sh.helm.release.v1.cceaddon-autoscaler.v2 -nkube-system
secret "sh.helm.release.v1.cceaddon-autoscaler.v1" deleted
secret "sh.helm.release.v1.cceaddon-autoscaler.v2" deleted
[root@cce-123-vpc-node2 ~]#
```

Step 3 If the add-on is created when Helm v2 is used, CCE automatically bumps the v2 release in ConfigMaps to v3 release in Secrets when viewing the add-ons and their details. The v2 release in the original ConfigMap is not deleted. Run the following command to view the ConfigMap list of add-on releases:

kubectl get configmap -A | grep cceaddon

```
cluster-autoscaler-th-config    1    7d10h
[paas@192-168-0-64 ~]$ kubectl get configmap -nkube-system | grep cceaddon
cceaddon-autoscaler.v1          1    7d10h
cceaddon-autoscaler.v2          1    52m
cceaddon-coredns.v1            1    14d
cceaddon-everest.v1            1    14d
[paas@192-168-0-64 ~]$
```

The ConfigMap name of an add-on release is in the format of **cceaddon-*{add-on name}*.v***. If there are multiple release versions, you can delete their ConfigMaps at the same time.

Step 4 Run the **release configmap** command to delete the ConfigMaps.

Example:

kubectl delete configmap cceaddon-autoscaler.v1 cceaddon-autoscaler.v2 -nkube-system

```
[paas@192-168-0-64 ~]$ kubectl delete configmap cceaddon-autoscaler.v1 cceaddon-autoscaler.v2 -nkube-system
configmap "cceaddon-autoscaler.v1" deleted
configmap "cceaddon-autoscaler.v2" deleted
[paas@192-168-0-64 ~]$
```

⚠ CAUTION

Deleting resources in kube-system is a high-risk operation. Ensure that the command is correct before running it to prevent resources from being deleted by mistake.

Step 5 On the CCE console, install the add-on and then uninstall it. Ensure that the residual add-on resources are cleared. After the uninstallation is complete, install the add-on again.

 NOTE

When installing the add-on for the first time, you may find it abnormal after the installation due to the residual resources of the previous add-on release, which is normal. In this case, you can uninstall the add-on on the console to ensure that the residual resources are cleared and the add-on can run properly after being installed again.

----End

10.2 How Do I Configure the Add-on Resource Quotas Based on Cluster Scale?

After changing the cluster scale, adjust the add-on resource quotas based on the cluster scale to ensure that the add-on pods can run properly. For example, if you expand the cluster scale from 50 worker nodes to 200 worker nodes or more, increase the CPU and memory quotas of the add-on pods to avoid exceptions such as OOM caused by too many nodes required for scheduling the add-on pods.

Configuring Resource Quotas for coredns

Queries per Second (QPS) of the coredns add-on is positively correlated with the CPU consumption. If the number of nodes or containers in the cluster grows, the coredns pod will bear heavier workloads. Adjust the number of add-on pods and their CPU and memory quotas based on the cluster scale.

Table 10-1 Recommended values for coredns

Nodes	Recommended Configuration (QPS)	Pods	CPU Request (m)	CPU Limit (m)	Memory Request (MiB)	Memory Limit (MiB)
50	2500	2	500	500	512	512
200	5000	2	1000	1000	1024	1024
1000	10000	2	2000	2000	2048	2048
2000	20000	4	2000	2000	2048	2048

Configuring Resource Quotas for everest

After the cluster scale is adjusted, the everest specifications need to be modified based on the cluster scale and the number of PVCs. The requested CPU and memory can be increased based on the number of nodes and PVCs. For details, see [Table 10-2](#).

In non-typical scenarios, the formulas for estimating the limit values are as follows:

- everest-csi-controller

- CPU limit: 250m for 200 or fewer nodes, 350m for 1000 nodes, and 500m for 2000 nodes
- Memory limit = (200 MiB + Number of nodes x 1 MiB + Number of PVCs x 0.2 MiB) x 1.2
- everest-csi-driver
 - CPU limit: 300m for 200 or fewer nodes, 500m for 1000 nodes, and 800m for 2000 nodes
 - Memory limit: 300 MiB for 200 or fewer nodes, 600 MiB for 1000 nodes, and 900 MiB for 2000 nodes

Table 10-2 Recommended configuration limits in typical scenarios

Configuration Scenario			everest-csi-controller		everest-csi-driver	
Nodes	PVs/PVCs	Add-on Pods	CPU Cores (Limit = Request)	Memory (Limit = Request)	CPU Cores (Limit = Request)	Memory (Limit = Request)
50	1000	2	250m	600 MiB	300m	300 MiB
200	1000	2	250m	1 GiB	300m	300 MiB
1000	1000	2	350m	2 GiB	500m	600 MiB
1000	5000	2	450m	3 GiB	500m	600 MiB
2000	5000	2	550m	4 GiB	800m	900 MiB
2000	10000	2	650m	5 GiB	800m	900 MiB

Configuring Resource Quotas for autoscaler

autoscaler automatically adjusts the number of nodes in a cluster based on workloads. Adjust the number of add-on pods and their CPU and memory quotas based on the cluster scale.

Table 10-3 Recommended values for autoscaler

Node	Pod	CPU Request (m)	CPU Limit (m)	Memory Request (MiB)	Memory Limit (MiB)
50	2	1000	1000	1000	1000
200	2	4000	4000	2000	2000
1000	2	8000	8000	8000	8000
2000	2	8000	8000	8000	8000

Configuring Resource Quotas for volcano

After the cluster scale is increased, the resource quotas required by volcano need to be modified based on the cluster scale.

- If the number of nodes is less than 100, retain the default configuration. The requested CPU is 500m, and the limit is 2000m. The requested memory is 500 MiB, and the limit is 2000 MiB.
- If the number of nodes is greater than 100, increase the requested CPU by 500m and the requested memory by 1000 MiB each time 100 nodes (10,000 pods) are added. Increase the CPU limit by 1500m and the memory limit by 1000 MiB.

NOTE

Formulas for calculating the requests:

- CPU request: Calculate the number of nodes multiplied by the number of pods, perform interpolation search using the product of the number of nodes in the cluster multiplied by the number of pods in [Table 10-4](#), and round up the request and limit that are closest to the specifications.

For example, for 2000 nodes (20,000 pods), the product of the number of nodes multiplied by the number of pods is 40 million, which is close to 700/70,000 in the specification (Number of nodes x Number of pods = 49 million). Set the CPU request to 4000m and the limit to 5500m.

- Memory request: Allocate 2.4 GiB of memory to every 1000 nodes and 1 GiB of memory to every 10,000 pods. The memory request is the sum of the two values. (The obtained value may be different from the recommended value in [Table 10-4](#). You can use either of them.)

Memory request = Number of nodes/1000 x 2.4 GiB + Number of pods/10000 x 1 GiB

For example, for 2000 nodes and 20,000 pods, the memory request value is 6.8 GiB (2000/1000 x 2.4 GiB + 20000/10000 x 1 GiB).

Table 10-4 Recommended requested resources and resource limits for volcano-controller and volcano-scheduler

Nodes/Pods in a Cluster	CPU Request (m)	CPU Limit (m)	Memory Request (MiB)	Memory Limit (MiB)
50/5000	500	2000	500	2000
100/10000	1000	2500	1500	2500
200/20000	1500	3000	2500	3500
300/30000	2000	3500	3500	4500
400/40000	2500	4000	4500	5500
500/50000	3000	4500	5500	6500
600/60000	3500	5000	6500	7500
700/70000	4000	5500	7500	8500

Configuring Resource Quotas for Other Add-ons

Resource quotas of other add-ons may also be insufficient due to cluster scale expansion. If, for example, the CPU or memory usage of the add-on pods increases and even OOM occurs, modify the resource quotas as required.

For example, the resources occupied by the kube-prometheus-stack add-on are related to the number of pods in the cluster. If the cluster scale is expanded, the number of pods may also grow. In this case, increase the resource quotas of the prometheus pods.

10.3 How Can I Clean Up Residual Resources After the NGINX Ingress Controller Add-on in the Unknown State Is Deleted?

Symptom

The NGINX Ingress Controller add-on is in the unknown state, and after this add-on is uninstalled, residual components still remain.

Involved Kubernetes resources include:

- Namespace-level resources: secret, ConfigMap, Deployment, Service, Role, RoleBinding, lease, ServiceAccount, and job
- Cluster-level resources: ClusterRole, ClusterRoleBinding, IngressClass, and ValidatingWebhookConfiguration

Solution

Step 1 Use kubectl to access a cluster.

Step 2 Search for related resources.

```
className="nginx"
namespace="kube-system"
className=`if [[ ${className} == "nginx" ]]; then echo ""; else echo "-${className}";fi`
kubectl get -n ${namespace} secret sh.helm.release.v1.cceaddon-nginx-ingress${className}.v1 cceaddon-
nginx-ingress${className}-admission
kubectl get -n ${namespace} cm cceaddon-nginx-ingress${className}-controller
kubectl get -n ${namespace} deploy cceaddon-nginx-ingress${className}-controller cceaddon-nginx-ingress
${className}-default-backend
kubectl get -n ${namespace} svc cceaddon-nginx-ingress${className}-controller-admission cceaddon-nginx-
ingress${className}-default-backend cceaddon-nginx-ingress${className}-controller
kubectl get -n ${namespace} role cceaddon-nginx-ingress${className}
kubectl get -n ${namespace} rolebinding cceaddon-nginx-ingress${className}
kubectl get -n ${namespace} lease ingress-controller-leader${className}
kubectl get -n ${namespace} serviceAccount cceaddon-nginx-ingress${className}
kubectl get clusterRole cceaddon-nginx-ingress${className}
kubectl get clusterRoleBinding cceaddon-nginx-ingress${className}
kubectl get ingressClass ${className}
kubectl get ValidatingWebhookConfiguration cceaddon-nginx-ingress${className}-admission
```

className specifies the name of a controller. **namespace** specifies the namespace where NGINX Ingress Controller was installed.

Step 3 Manually delete the residual resources if the preceding resources are present.

----End

10.4 Why TLS v1.0 and v1.1 Cannot Be Used After the NGINX Ingress Controller Add-on Is Upgraded?

Symptom

After the NGINX Ingress Controller add-on is upgraded to 2.3.3 or later, if the TLS version of the client is earlier than v1.2, an error is reported during the negotiation between the client and NGINX Ingress Controller.

```
[root@~]# curl -I --tls-max 1.1 -kv https://192.168.0.141:443
* Trying 192.168.0.141:443...
* Connected to 192.168.0.141 (192.168.0.141) port 443 (#0)
* ALPN, offering h2
* ALPN, offering http/1.1
* successfully set certificate verify locations:
* CAfile: /etc/pki/tls/certs/ca-bundle.crt
* CApath: none
* TLSv1.1 (OUT), TLS handshake, Client hello (1):
* TLSv1.1 (IN), TLS alert, protocol version (582):
* error:1409442E:SSL routines:ssl3_read_bytes:tlsv1 alert protocol version
* Closing connection 0
curl: (35) error:1409442E:SSL routines:ssl3_read_bytes:tlsv1 alert protocol version
```

Solution

NGINX Ingress Controller 2.3.3 and later versions support only TLS v1.2 and v1.3 by default. If additional TLS versions are needed, you can add the `@SECLEVEL=0` field to the `ssl-ciphers` parameter configured for the NGINX Ingress Controller add-on. For details, see [TLS/HTTPS](#).

- Step 1** Log in to the CCE console and click the cluster name to access the cluster console. In the navigation pane, choose Add-ons, locate the NGINX Ingress Controller add-on, and click **Manage**.
- Step 2** Click **Edit** of the corresponding instance.
- Step 3** Add the following configuration to the **Nginx Parameters**:

```
{
  "ssl-ciphers": "@SECLEVEL=0 ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-CHACHA20-POLY1305:ECDHE-RSA-CHACHA20-POLY1305:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:DHE-RSA-CHACHA20-POLY1305:ECDHE-ECDSA-AES128-SHA256:ECDHE-RSA-AES128-SHA256:ECDHE-ECDSA-AES128-SHA:ECDHE-RSA-AES128-SHA:ECDHE-ECDSA-AES256-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-ECDSA-AES256-SHA:ECDHE-RSA-AES256-SHA:DHE-RSA-AES128-SHA256:DHE-RSA-AES256-SHA256:AES128-GCM-SHA256:AES256-GCM-SHA384:AES128-SHA256:AES256-SHA256:AES128-SHA:AES256-SHA:DES-CBC3-SHA",
  "ssl-protocols": "TLSv1 TLSv1.1 TLSv1.2 TLSv1.3"
}
```

- Step 4** Click **OK**.
- Step 5** Use TLS v1.1 for access again. The response is normal.


```
[root@... ]# curl -I --tls-max 1.1 -kv https://192.168.0.141:443
* Trying 192.168.0.141:443...
* Connected to 192.168.0.141 (192.168.0.141) port 443 (#0)
* ALPN, offering h2
* ALPN, offering http/1.1
* successfully set certificate verify locations:
* CAfile: /etc/pki/tls/certs/ca-bundle.crt
* CApath: none
* TLSv1.1 (OUT), TLS handshake, Client hello (1):
* TLSv1.1 (IN), TLS handshake, Server hello (2):
* TLSv1.1 (IN), TLS handshake, Certificate (11):
* TLSv1.1 (IN), TLS handshake, Server key exchange (12):
* TLSv1.1 (IN), TLS handshake, Server finished (14):
* TLSv1.1 (OUT), TLS handshake, Client key exchange (16):
* TLSv1.1 (OUT), TLS change cipher, Change cipher spec (1):
* TLSv1.1 (OUT), TLS handshake, Finished (20):
* TLSv1.1 (IN), TLS handshake, Finished (20):
```

----End

11 API & kubectl FAQs

11.1 How Can I Access a Cluster API Server?

You can use either of the following methods to access a cluster API server:

- (Recommended) Through the cluster API. This access mode uses certificate authentication. It is suitable for API calls on scale thanks to its direct connection to the API Server. This is a recommended option.
- API Gateway. This access mode uses token authentication. You need to obtain a token using your account. This access mode applies to small-scale API calls. API gateway flow control may be triggered when APIs are called on scale.

11.2 Can the Resources Created Using APIs or kubectl Be Displayed on the CCE Console?

The CCE console does not support the display of the following Kubernetes resources: DaemonSets, ReplicationControllers, ReplicaSets, and endpoints.

To query these resources, run the kubectl commands.

In addition, Deployments, StatefulSets, Services, and pods can be displayed on the console only when the following conditions are met:

- Deployments and StatefulSets: At least one label uses **app** as its key.
- Pods: Pods are displayed on the **Pods** tab page in the workload details only after a Deployment or StatefulSet has been created.
- Services: Services are displayed on the **Access Mode** tab page in the Deployment or StatefulSet details.

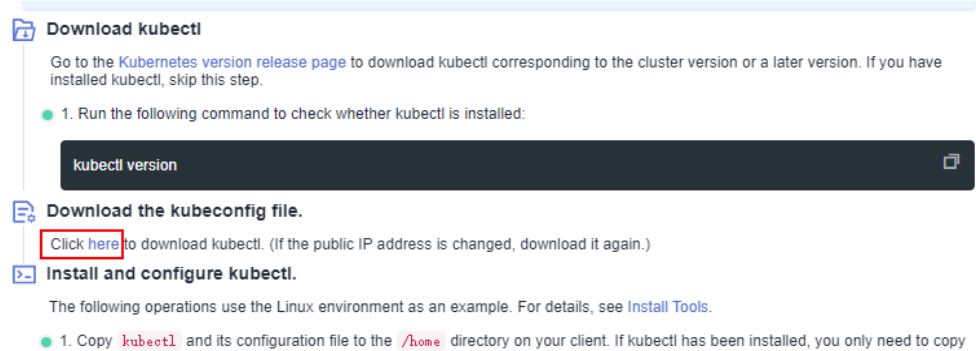
The Services displayed on this tab page are associated with the workload.

- a. At least one label of the workload uses **app** as its key.
- b. The label of a Service is the same as that of the workload.

11.3 How Do I Download kubeconfig for Connecting to a Cluster Using kubectl?

- Step 1** Log in to the CCE console and click the target cluster to access the cluster console.
- Step 2** In the **Connection Information** area, view the kubectl connection mode.
- Step 3** In the window that is displayed, download the kubectl configuration file (**kubeconfig.json**).

Figure 11-1 Downloading kubeconfig.json



----End

11.4 How Do I Rectify the Error Reported When Running the kubectl top node Command?

Symptom

The error message "Error from server (ServiceUnavailable): the server is currently unable to handle the request (get nodes.metrics.k8s.io)" is displayed after the **kubectl top node** command is executed.

Possible Causes

"Error from server (ServiceUnavailable)" indicates that the cluster is not connected. In this case, you need to check whether the network between kubectl and the master node in the cluster is normal.

Solution

- If the kubectl command is executed outside the cluster, check whether the cluster is bound to an EIP. If yes, download the **kubeconfig** file and run the kubectl command again.
- If the kubectl command is executed on a node in the cluster, check the security group of the node and check whether the TCP/UDP communication between the worker node and master node is allowed. For details about the security group, see [How Can I Configure a Security Group Rule in a Cluster?](#)

11.5 Why Is "Error from server (Forbidden)" Displayed When I Use kubectl?

Symptom

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"
```

Possible Cause

This user has no permissions to operate Kubernetes resources.

Solution

Assign permissions to the user.

- Step 1** Log in to the CCE console. In the navigation pane, choose **Permissions**.
- Step 2** Select a cluster for which you want to add permissions from the drop-down list on the right.
- Step 3** Click **Add Permissions** in the upper right corner.
- Step 4** 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.

NOTE

If you do not have IAM permissions, you cannot select users or user groups when configuring permissions for other users or user groups. In this case, you can enter a user ID or user group ID.

Permissions can be customized as required. After selecting **Custom** for **Permission Type**, click **Add Custom Role** on the right of the **Custom** parameter. In the dialog box displayed, enter a name and select a rule. After the custom rule is created, you can select a value from the **Custom** drop-down list box.

Custom permissions are classified into ClusterRole and Role. Each ClusterRole or Role contains a group of rules that represent related permissions. For details, see [Using RBAC Authorization](#).

- A ClusterRole is a cluster-level resource that can be used to configure cluster access permissions.
- A Role is used to configure access permissions in a namespace. When creating a Role, specify the namespace to which the Role belongs.

- Step 5** Click **OK**.

----End

12 DNS FAQs

12.1 What Should I Do If Domain Name Resolution Fails in a CCE Cluster?

Check Item 1: Whether the coredns Add-on Has Been Installed

- Step 1** Log in to the CCE console and click the cluster name to access the cluster console.
- Step 2** In the navigation pane, choose **Add-ons** and check whether the CoreDNS add-on has been installed.
- Step 3** If not, install the add-on. For details, see [Why Does a Container in a CCE Cluster Fail to Perform DNS Resolution?](#)

----End

Check Item 2: Whether the coredns Instance Reaches the Performance Limit

CoreDNS QPS is positively correlated with the CPU usage. If the QPS is high, adjust the coredns instance specifications based on the QPS.

- Step 1** Log in to the CCE console and click the cluster name to access the cluster console.
- Step 2** In the navigation pane, choose **Add-ons** and verify that CoreDNS is running.
- Step 3** Click the CoreDNS add-on name to view the add-on pod list.
- Step 4** Click **Monitor** of the add-on pods to view the CPU and memory usage.

If the add-on performance reaches the bottleneck, adjust the coredns add-on specifications.

----End

Check Item 3: Whether the External Domain Name Resolution Is Slow or Times Out

If the domain name resolution failure rate is lower than 1/10000, optimize parameters by referring to [How Do I Optimize the Configuration If the External](#)

[Domain Name Resolution Is Slow or Times Out?](#) or add a retry policy in the service.

Check Item 4: Whether UnknownHostException Occurs

When service requests in the cluster are sent to an external DNS server, a domain name resolution error occurs due to occasional UnknownHostException. UnknownHostException is a common exception. When this exception occurs, check whether there is any domain name-related error or whether you have entered a correct domain name.

To locate the fault, perform the following steps:

- Step 1** Check the host name carefully (spelling and extra spaces).
- Step 2** Check the DNS settings. Before running the application, run the **ping hostname** command to ensure that the DNS server has been started and running. If the host name is new, you need to wait for a period of time before the DNS server is accessed.
- Step 3** Check the CPU and memory usage of the coredns add-on to determine whether the performance bottleneck has been reached. For details, see [Check Item 2: Whether the coredns Instance Reaches the Performance Limit](#).
- Step 4** Check whether traffic limiting is performed on the coredns add-on. If traffic limiting is triggered, the processing time of some requests may be prolonged. In this case, you need to adjust the coredns add-on specifications.

Log in to the node where the coredns add-on is installed and view the following content:

```
cat /sys/fs/cgroup/cpu/kubepods/pod<pod_uid>/<coredns container ID>/cpu.stat
```

- `<pod uid>` indicates the pod UID of the coredns add-on, which can be obtained by running the following command:

```
kubectl get po <pod name> -nkube-system -ojsonpath='{.metadata.uid}'
```

In the preceding command, `<pod name>` indicates the name of the coredns add-on running on the current node.

- `<coredns container ID>` must be a complete container ID, which can be obtained by running the following command:

Docker nodes:

```
docker ps --no-trunc | grep k8s_coredns | awk '{print $1}'
```

containerd nodes:

```
crictl ps --no-trunc | grep k8s_coredns | awk '{print $1}'
```

Example:

```
cat /sys/fs/cgroup/cpu/kubepods/  
pod27f58662-3979-448e-8f57-09b62bd24ea6/6aa98c323f43d689ac47190bc84cf4fadd23bd8dd25307f773df2  
5003ef0eef0/cpu.stat
```

Pay attention to the following metrics:

- **nr_throttled**: number of times that traffic is limited.
- **throttled_time**: total duration of traffic limiting, in nanoseconds.

----End

If the host name and DNS settings are correct, you can use the following optimization policies.

Optimization policies:

1. Change the `coredns` cache time.
2. Configure the stub domain.
3. Modify the value of `ndots`.

NOTE

- **Increasing the cache time of `coredns`** helps resolve the same domain name for the *N* time, reducing the number of cascading DNS requests.
- **Configuring the stub domain** can reduce the number of DNS request links.

How to modify:

1. Modifying the `coredns` cache time and configuring the stub domain:
Restart the `coredns` add-on after you modify the configurations.
2. Modifying `ndots`:

[How Do I Optimize the Configuration If the External Domain Name Resolution Is Slow or Times Out?](#)

Example:

```
dnsConfig:
  options:
    - name: timeout
      value: '2'
    - name: ndots
      value: '5'
    - name: single-request-reopen
```

You are advised to change the value of `ndots` to 2.

12.2 Why Does a Container in a CCE Cluster Fail to Perform DNS Resolution?

Symptom

A customer bound its domain name to the private domain names in the DNS service and also to a specific VPC. It is found that the ECSs in the VPC can properly resolve the private domain name but the containers in the VPC cannot.

Application Scenario

Containers in a VPC cannot resolve domain names.

Solution

According to the resolution rules of private domain names, the subnet DNS in the VPC must be set to the cloud DNS. You can find the details of the private network DNS service on its console.

The customer can perform domain name resolution on the ECSs in the VPC subnet, which indicates that the preceding configuration has been completed in the subnet.

```
bash-4.4# exit
exit
[root@global-skyworth1-vpn ~]# ping [REDACTED]
PING [REDACTED] (10.247.11.29) 56(84) bytes of data.

^C
--- ota.skyworth.web ping statistics ---
```

However, when the domain name resolution is performed in a container, the message "bad address" is displayed, indicating that the domain name cannot be resolved.

```
[root@global-skyworth1-vpn ~]#
[root@global-skyworth1-vpn ~]# docker exec -it 86cf062a5ba3 bash
bash-4.4# ping [REDACTED]
ping: bad address: '[REDACTED]'
bash-4.4#
```

Log in to the CCE console and check the add-ons installed in the cluster.

If you find that the coredns add-on does not exist in **Add-ons Installed**, the coredns add-on may have been incorrectly uninstalled.

Install it and add the corresponding domain name and DNS service address to resolve the domain name.

12.3 How Do I Optimize the Configuration If the External Domain Name Resolution Is Slow or Times Out?

The following is an example **resolv.conf** file for a container in a workload:

```
root@test-5dffddd95-vpt4m:/# cat /etc/resolv.conf
nameserver 10.247.3.10
search istio.svc.cluster.local svc.cluster.local cluster.local
options ndots:5 single-request-reopen timeout:2
```

In the preceding information:

- **nameserver**: IP address of the DNS. Set this parameter to the cluster IP address of CoreDNS.
- **search**: domain name search list, which is a common suffix of Kubernetes.
- **ndots**: If the number of dots (.) is less than the domain name, **search** is preferentially used for resolution.
- **timeout**: timeout interval.
- **single-request-reopen**: indicates that different source ports are used to send different types of requests.

By default, when you create a workload on the CCE console, the preceding parameters are configured as follows:

```
dnsConfig:
  options:
    - name: timeout
      value: '2'
    - name: ndots
      value: '5'
    - name: single-request-reopen
```

These parameters can be optimized or modified based on service requirements.

Scenario 1: Slow External Domain Name Resolution

Optimization Solution

1. If the workload does not need to access the Kubernetes Service in the cluster, see [How Do I Configure a DNS Policy for a Container?](#).
2. If the number of dots (.) in the domain name used by the working Service to access other Kubernetes Services is less than 2, set **ndots** to 2.

Scenario 2: External Domain Name Resolution Timeout

Optimization Solution

1. Generally, the timeout of a Service must be greater than the value of **timeout** multiplied by **attempts**.
2. If it takes more than 2s to resolve the domain name, you can set **timeout** to a larger value.

12.4 How Do I Configure a DNS Policy for a Container?

CCE uses **dnsPolicy** to identify different DNS policies for each pod. The value of **dnsPolicy** can be either of the following:

- **None:** No DNS policy is configured. In this mode, you can customize the DNS configuration, and **dnsPolicy** needs to be used together with **dnsConfig** to customize the DNS.
- **Default:** The pod inherits the name resolution configuration from the node where the pod is running. The container's DNS configuration file is the DNS configuration file that the kubelet's **--resolv-conf** flag points to. In this case, a cloud DNS is used for CCE clusters.
- **ClusterFirst:** In this mode, the DNS in the pod uses the DNS service configured in the cluster. That is, the kube-dns or CoreDNS service in the Kubernetes is used for domain name resolution. If the resolution fails, the DNS configuration of the host machine is used for resolution.

If the type of **dnsPolicy** is not specified, **ClusterFirst** is used by default.

- If the type of **dnsPolicy** is set to **Default**, the name resolution configuration is inherited from the worker node where the pod is running.
- If the type of **dnsPolicy** is set to **ClusterFirst**, DNS queries will be sent to the kube-dns service.

The kube-dns service responds to queries on the domains that use the configured cluster domain suffix as the root. All other queries (for example, `www.kubernetes.io`) are forwarded to the upstream name server inherited from the node. Before this feature was supported, stub domains were typically introduced by a custom resolver, instead of the upstream DNS. However, this causes the custom resolver itself to be the key path to DNS resolution, where scalability and availability issues can make the DNS functions unavailable to the cluster. This feature allows you to introduce custom resolvers without taking over the entire resolution path.

If a workload does not need to use CoreDNS in the cluster, you can use `kubectl` or call the APIs to set the `dnsPolicy` to `Default`.

13 Image Repository FAQs

13.1 How Do I Upload My Images to CCE?

SWR manages images for CCE. It provides the following ways to upload images:

- [Uploading an Image Through the Client](#)
- [Uploading an Image Through the SWR Console](#)

14 Permissions

14.1 Can I Configure Only Namespace Permissions Without Cluster Management Permissions?

Namespace permissions and cluster management permissions are independent and complementary to each other.

- Namespace permissions: apply to clusters and are used to manage operations on cluster resources (such as creating workloads).
- Cluster management (IAM) permissions: apply to cloud services and used to manage CCE clusters and peripheral resources (such as VPC, ELB, and ECS).

Administrators of the IAM Admin user group can grant cluster management permissions (such as CCE Administrator and CCE FullAccess) to IAM users or grant namespace permissions on a cluster on the CCE console. However, the permissions you have on the CCE console are determined by the IAM system policy. If the cluster management permissions are not configured, you do not have the permissions for accessing the CCE console.

If you only run `kubectl` commands to work on cluster resources, you only need to obtain the `kubeconfig` file with the namespace permissions. For details, see [Can I Use kubectl If the Cluster Management Permissions Are Not Configured?](#).

Note that information leakage may occur when you use the `kubeconfig` file.

14.2 Can I Use CCE APIs If the Cluster Management Permissions Are Not Configured?

CCE has cloud service APIs and cluster APIs.

- Cloud service APIs: You can perform operations on the infrastructure (such as creating nodes) and cluster resources (such as creating workloads).

When using cloud service APIs, the IAM permissions must be configured.

- Cluster APIs: You can perform operations on cluster resources (such as creating workloads) through the Kubernetes native API server, but not on cloud infrastructure resources (such as creating nodes).

When using cluster APIs, you only need to add the cluster certificate. Only the users with the IAM permissions can download the cluster certificate. Note that information leakage may occur during certificate transmission.

14.3 Can I Use kubectl If the Cluster Management Permissions Are Not Configured?

IAM authentication is not required for running kubectl commands. Therefore, you can run kubectl commands without configuring cluster management (IAM) permissions. However, you need to obtain the kubectl configuration file (kubeconfig) with the namespace permissions. In the following scenarios, information leakage may occur during file transmission.

- Scenario 1

If an IAM user has been configured with the cluster management permissions and namespace permissions, downloads the kubeconfig authentication file and then deletes the cluster management permissions (reserving the namespace permissions), kubectl can still be used to perform operations on Kubernetes clusters. Therefore, if you want to permanently delete the permission of a user, you must also delete the cluster management permissions and namespace permissions of the user.

- Scenario 2

An IAM user has certain cluster management and namespace permissions and downloads the kubeconfig authentication file. In this case, CCE determines which Kubernetes resources can be accessed by kubectl based on the user information. That is, the authentication information of a user is recorded in kubeconfig. Anyone can use kubeconfig to access the cluster.