ALIBABA CLOUD

Alibaba Cloud

Elastic Container Instance

Kubernetes

Document Version: 20211201

Alibaba Cloud

Legal disclaimer

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

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

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

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

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

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

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

Document conventions

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

Table of Contents

1.Connect Kubernetes to Elastic Container Instance ---- 05
1.1. Connect Kubernetes to Elastic Container Instance ---- 05
1.2. Use Elastic Container Instance in ASK clusters ---- 08
1.3. Use Elastic Container Instance in ACK clusters ---- 09
1.4. Use Elastic Container Instance in self-managed Kubernetes... ---- 10
1.5. Use Elastic Container Instance in offline Kubernetes cluster... ---- 18
1.6. FAQ related to Virtual Kubelet deployment in an on-premi... ---- 26
1.7. Update Virtual Kubelet ---- 29
2.Use Elastic Container Instance ---- 32
2.1. Overview ---- 32
2.2. Schedule pods to Elastic Container Instance ---- 33
2.3. Configure Elastic Container Instance Profile ---- 37
2.4. Install the virtual-kubelet-autoscaler add-on in an ACK clu... ---- 44
2.5. Deploy and use ack-kubernetes-elastic-workload in an ACK... ---- 45
2.6. ECI Pod Annotation ---- 51
2.7. Custom events of Elastic Container Instance ---- 67

1.Connect Kubernetes to Elastic Container Instance

1.1. Connect Kubernetes to Elastic Container Instance

Elastic Container Instance can be used to provide a basic runtime environment of pods for Kubernetes clusters. Other capabilities such as dependencies between business, load balancing, auto scaling, and regular scheduling must still be provided by Kubernetes. This topic describes how to connect Kubernetes to Elastic Container Instance and use Elastic Container Instance to run pods.

Connection methods

Elastic Container Instance provides a hierarchical solution to manage Kubernetes resources. Elastic Container Instance schedules and manages pods at the infrastructure layer, whereas Kubernetes manages workloads such as Deployments, Services, StatefulSets, and CronJobs on the platform layer.

After you connect Elastic Container Instance to Kubernetes, Elastic Container Instance takes over the management of pods, including the infrastructure and resource availability. Kubernetes no longer needs to manage the lifecycle and resources of the underlying virtual machines (VMs).

Elastic Container Instance seamlessly connects to Kubernetes by using virtual nodes based on Virtual Kubelet provided by the Kubernetes community. Elastic Container Instance provides high elasticity to Kubernetes clusters, which allows you to break free from the limits of the computing capacities of the clusters on which Kubernetes is deployed. For more information, visit Virtual Kubelet.

You can use one of the following methods to connect Elastic Container Instance to Kubernetes:

Connection method	Description
Run Kubernetes pods exclusively on Elastic Container Instance	You need to deploy a serverless Kubernetes (ASK) cluster, and then the entire ASK cluster runs on Elastic Container Instance. This deployment method eliminates the need to focus on O&M and capacity issues of the underlying VMs.
Run Kubernetes pods on both Elastic Container Instance and traditional servers	Elastic Container Instance connects to Kubernetes clusters as virtual nodes. You can use this method to improve the resource usage and elasticity of clusters and reduce running costs.

Run Kubernetes pods exclusively on Elastic Container Instance by deploying an ASK cluster

In this deployment method, all Kubernetes pods run on Elastic Container Instance, which manages the underlying infrastructure. Kubernetes needs only to manage workloads to ensure the reliability of the business.

If you choose this deployment method, we recommend that you select an ASK cluster. ASK is optimized to run entirely on Elastic Container Instance. It is a fully managed and cost-effective Kubernetes environment that is suitable for online and offline business, simulation environments, and development and test environments.

For more information about how to use Elastic Container Instance to deploy ASK clusters, see Use Elastic Container Instance in ASK clusters.

Run Kubernetes pods on both Elastic Container Instance and traditional servers

If you have an existing Kubernetes cluster, you can add Elastic Container Instance as a virtual node to the cluster. You can schedule the excess traffic of long-running business loads to Elastic Container Instance. This allows you to reduce idle resources and ensure fast and flexible scalability. When business traffic decreases, the Kubernetes cluster can release the pods deployed in Elastic Container Instance to reduce costs.

You can use one of the following methods to connect to Elastic Container Instance based on your existing Kubernetes clusters:

- If you use Container Service for Kubernetes (ACK) to deploy clusters, you can add virtual nodes in the ACK console to use Elastic Container Instance. For more information, see Use Elastic Container Instance in ACK clusters.
- If you have created Kubernetes clusters in Elastic Compute Service (ECS), you must deploy Virtual Kubelet to use Elastic Container Instance. For more information, see Use Elastic Container Instance in self-managed Kubernetes clusters.
- If you deploy Kubernetes clusters in data centers or other cloud platforms, you must deploy Virtual Kubelet to use Elastic Container Instance. For more information, see Use Elastic Container Instance in offline Kubernetes clusters. For further help, consult solution architects of Alibaba Cloud.

Management tools

You can use the following methods to manage Kubernetes and view the running status of elastic container instances:

- Elastic Container Instance console

  You can view the running status of elastic container instances in the Elastic Container Instance console. Perform the following steps:

  i. Log on to the Elastic Container Instance console.

  ii. In the top navigation bar, select a region.

  iii. The **Container Group** page appears. You can view the elastic container instances that are created in the selected region.

- ACK console

  You can manage ASK or ACK clusters and view the running status of elastic container instances in the ACK console. To view the running status of elastic container instances, perform the following steps:

  i. Log on to the ACK console.

ii. In the left-side navigation pane, click **Clusters**.

iii. In the cluster list, find the cluster that you want to view and click the cluster ID to go the details page.

iv. In the left-side navigation pane, choose **Workloads > Pods**.

v. On the **Pods** page, select a namespace from the drop-down list. Then, you can view elastic container instances of the namespace.

- Cloud Shell

  You can manage Kubernetes clusters by using Cloud Shell provided by Alibaba Cloud. For more information, see Use kubectl on Cloud Shell to manage ACK clusters.

- kubectl client

  You can use the kubectl client to connect to a remote Kubernetes cluster from your computer. For more information, see Connect to ACK clusters by using kubectl.

Limits

Elastic Container Instance does not support some Kubernetes features such as hostPaths and DaemonSets due to security limits of Alibaba Cloud public cloud and limits imposed by virtual nodes. The following table describes the unsupported features.

Item	Description	Recommended alternative
HostPath	Allows you to mount files from on-premises hosts to containers.	Use emptyDir volumes, disks, or Apsara File Storage NAS (NAS) file systems.
HostNetwork	Allows you to map a host port to a container.	Create a Service of the LoadBalancer type.
DaemonSet	Allows you to deploy a static pod on the host of a container.	Deploy multiple images in a pod by using sidecar containers.
Privileged permissions	Allows you to grant privileged permissions to a container.	Use a security context to grant permissions to a pod.
Service of the NodePort type	Allows you to map a host port to a container.	Create a Service of the LoadBalancer type.

If you deploy Kubernetes exclusively on Elastic Container Instance or use virtual nodes to expand the scalability of your Kubernetes cluster, take note of the following items:

- ASK clusters share an image repository with Kubernetes clusters. To simplify image pulling, you can upload your container images to the image repository in advance. We recommend that you use Alibaba Cloud Container Registry (ACR) and the image address (registry-vpc.xxx) in virtual private cloud (VPC) to upload your container images.
- Both deployment methods support common controllers such as Deployments, ReplicaSets, Jobs, CronJobs, and StatefulSets. Applications of these types can run directly on ASK and Kubernetes clusters.
- Both deployment methods use PrivateZone to implement service discovery. We recommend that you enable PrivateZone when you create a cluster.
- Both deployment methods support loading balancing, which indicates that the type of Service is LoadBalancer.

1.2. Use Elastic Container Instance in ASK clusters

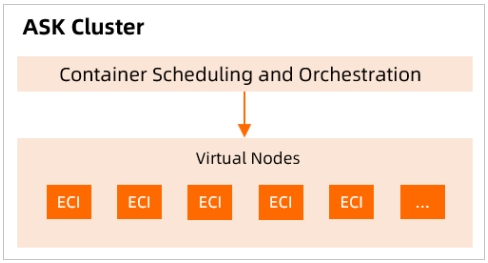
This topic describes how to use Elastic Container Instance in Serverless Kubernetes (ASK) clusters. ASK is integrated with a variety of Alibaba Cloud services, including Elastic Container Instance. You can create an ASK cluster deployed on Elastic Container Instance in a fast and simple manner.

Introduction to ASK

ASK is a serverless solution for Kubernetes containers that is provided by Alibaba Cloud on top of the elastic compute architecture. ASK allows you to deploy containerized applications without the need to purchase, manage, or plan nodes. You are charged only for the amount of CPU and memory resources that you allocate for your applications.

ASK is fully compatible with Kubernetes. You can use ASK API or command lines to manage containerized applications. ASK is also integrated with Alibaba Cloud services to help you simplify development on Kubernetes and focus on your applications instead of managing underlying resources.

In ASK clusters, you can create pods without manually deploying virtual nodes. Pods in ASK clusters run in a secure and isolated container runtime environment that is built on an elastic container instance. Each pod runs on a dedicated elastic container instance, as shown in the following figure.



For more information, see ASK overview.

Procedure

You can create an ASK cluster in the ACK console and deploy applications in the cluster.

> **Note**
>
> Each pod that is created in an ASK cluster runs on a dedicated elastic container instance. You can view information about each pod in the Elastic Container Instance console. To use Elastic Container Instance features when you create a pod, you can add annotations. For more information, see Pod annotations supported by Elastic Container Instance.

For the procedure about using Elastic Container Instance in ASK clusters, see the following topics:

1. Create an ASK cluster
2. Create an application from an image
3. Create a Service
4. View pods

1.3. Use Elastic Container Instance in ACK clusters

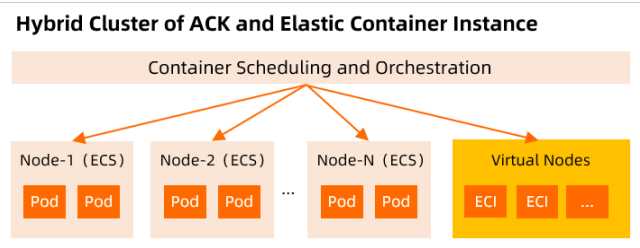
This topic describes how to deploy virtual nodes to connect Elastic Container Instance to Container Service for Kubernetes (ACK) clusters. Virtual nodes allow you to create pods on demand in a dynamic and flexible manner. This eliminates the need to plan nodes and helps you reduce costs and improve efficiency.

Introduction to ACK

ACK is one of the first batch of services that have passed the Certified Kubernetes Conformance Program and provides high-performance containerized application management service. ACK integrates the virtualization, storage, networking, and security capabilities provided by Alibaba Cloud, simplifies the creation and expansion of clusters, and enables you to focus on the development and management of containerized applications.

You can deploy virtual nodes to connect Elastic Container Instance to your ACK clusters. After you deploy virtual nodes, you can create elastic container instances on demand in the virtual nodes to extend your ACK clusters, without the need to plan the computing capacities of the nodes. The elastic container instances can communicate with the pods in real nodes of the clusters.

You must manually deploy virtual nodes in an ACK cluster before you can create pods in the virtual nodes. Pods in ACK clusters run in a secure and isolated container runtime environment that is built on elastic container instances. Each pod runs on a dedicated elastic container instance, as shown in the following figure.



For more information, see Overview of ACK clusters.

Procedure

If you have an ACK cluster, you can deploy virtual nodes in the cluster to use Elastic Container Instance. Perform the following steps:

1. Deploy virtual nodes.

   You must install the ack-virtual-node component to deploy virtual nodes. For more information, see the "Step 1: Deploy ack-virtual-node in ACK clusters" section in Deploy the virtual node controller and use it to create Elastic Container Instance-based pods.

   > **Note**
   >
   > To use Elastic Container Instance features, you must update ack-virtual-node. For managed ACK clusters, the system automatically updates ack-virtual-node. For dedicated ACK clusters, you must manually update ack-virtual-node. For more information, see Update Virtual Kubelet.

2. Schedule pods to the virtual nodes.

   After you deploy the virtual nodes in your ACK cluster, you can schedule pods to run on the virtual nodes. For more information, see Schedule pods to elastic container instances.

   > **Note**
   >
   > Each pod that is connected to a virtual node runs on a dedicated elastic container instance. You can view information about each pod in the Elastic Container Instance console. To use Elastic Container Instance features when you create a pod, you can add annotations. For more information, see Pod annotations supported by Elastic Container Instance.

For more information about how to use Elastic Container Instance to deploy applications, see the following topics:

- Use an elastic container instance to run a Job
- Deploy applications that provide services by using Ingresses

1.4. Use Elastic Container Instance in self-managed Kubernetes clusters

If your Kubernetes cluster is deployed on Elastic Computer Service (ECS), you must deploy virtual nodes (Virtual Kubelet) in the cluster before you can use Elastic Container Instance in your Kubernetes clusters. This topic describes how to connect an elastic container instance to a self-managed Kubernetes cluster that is deployed on ECS. This topic provides information on basic operations including how to register your Kubernetes cluster in the Container Service for Kubernetes (ACK) console and deploy virtual nodes, or deploy Virtual Kubelet in your self-managed cluster.

Background information

You can connect an elastic container instance to a Kubernetes cluster through the open source Virtual Kubelet implementation. This allows you to implement high elasticity for your Kubernetes cluster. To connect an elastic container instance to a self-managed Kubernetes cluster, use the following methods:

- Method 1: Register the cluster in the ACK console and deploy virtual nodes
- Method 2: Deploy Virtual Kubelet in the Kubernetes cluster

> **Note**
>
> We recommend that you use Method 1. This method allows you to update the Virtual Kubelet version (the ack-virtual-node component version) to use new features of elastic container instances. For more information, see Update Virtual Kubelet.

Preparations

You must obtain the information of the following parameters before you deploy virtual nodes.

Parameter	Description	Method to obtain the parameter value
ECI_ACCESS_KEY and ECI_SECRET_KEY	The AccessKey ID and corresponding AccessKey secret. The values of these parameters are the credentials used to manage the elastic container instance in virtual nodes.	For more information, see Obtain an AccessKey pair.
ALIYUN_CLUSTERID	The ID of the cluster, which is the unique identifier of the cluster.	• If you use Method 1, the cluster ID is generated by the system. • If you use Method 2, you must enter the cluster ID. The cluster ID consists of letters and digits. Make sure that the specified cluster ID is unique within the Alibaba Cloud account.
ECI_REGION	The region ID. The elastic container instance is deployed in this region.	You can query the regions where Elastic Container Instance is available by using the Elastic Container Instance console or by calling the DescribeRegions operation.
ECI_VPC	The ID of the virtual private cloud (VPC). The elastic container instance is deployed within this VPC.	You can create and view VPCs on the **VPCs** page in the VPC console.

Parameter	Description	Method to obtain the parameter value
ECI_VSWITCH	The ID of the vSwitch. The elastic container instance is associated with this vSwitch.	You can create and view vSwitches on the **vSwitch** page in the VPC console and select a corresponding vSwitch based on the selected VPC.
ECI_SECURITY_GROUP	The ID of the security group. The elastic container instance is added to this security group.	You can create and view security groups on the **Security Groups** page in the ECS console and select a corresponding security group based on the selected VPC.

Method 1: Register the cluster in the ACK console and deploy virtual nodes

You can register your self-managed cluster in the ACK console and then deploy virtual nodes in the cluster to use elastic container instances. Perform the following steps:

1. Log on to the ACK console.
2. In the left-side navigation pane, click **Clusters**.
3. Create a Kubernetes cluster and register it on the ACK console.
    i. On the **Clusters** page, click **Create Kubernetes Cluster**.

ii. Click the **Register Cluster** tab. Specify the parameters for the cluster and click **Create Cluster**.

The following table describes the parameters required to create a Kubernetes cluster. For more information, see Register an external Kubernetes cluster.

Parameter	Description
Region, VPC, and vSwitch	Select a region, VPC, and vSwitch for the cluster.
Access to API Server	Select a Server Load Balancer (SLB) instance type to use for the API server. By default, an internal-facing SLB instance is created for the API server.
EIP	Specify whether to create and associate an elastic IP address (EIP) to connect to the cluster.
Security group	A security group is automatically created to isolate security domains and control network traffic.
Deletion Protection	Specify whether to enable deletion protection for the cluster. If you enable deletion protection, the cluster cannot be deleted by using the ACK console or by calling API operations.

4. Register the cluster.
   i. On the **Clusters** page, find the cluster that you created and click the cluster name.
   ii. On the **Cluster Information** page, click the **Connection Information** tab.
   iii. Create a ConfigMap in the cluster.

   Click the **Public Network** or **Internal Network** tab based on your network. Copy the content to a YAML configuration file such as agent.yaml and run the `kubectl apply -f agent.yaml` command in the cluster to create a ConfigMap.
   iv. Run the following command in the cluster to check the connection status:

```
kubectl -n kube-system get pod |grep ack-cluster-agent
```

   The following command output is returned:

```
ack-cluster-agent-5f7d568f6-6fc4k       1/1   Running  0      9s
ack-cluster-agent-5f7d568f6-tf6fp       1/1   Running  0      9s
```

5. Deploy virtual nodes.
   i. On the **Clusters** page, click the name of the cluster in which you want to deploy a virtual node.

ii. In the left-side navigation pane of the details page, choose **Operations > Add-ons**.

iii. Click the **Others** tab, find ack-virtual-node, and then click **Install**.

Method 2: Deploy Virtual Kubelet in the Kubernetes cluster

You can deploy Virtual Kubelet in your clusters to use elastic container instances. You must deploy the latest version of Virtual Kubelet. For more information, see ack-virtual-node.

Perform the following steps to deploy Virtual Kubelet:

1. Prepare the vk.yaml configuration file required to deploy Virtual Kubelet.

   The following code provides an example of a YAML configuration file. You must delete the comments and replace the parameter values with the actual values based on your business requirements.

   > **Notice**
   >
   > The version of Virtual Kubelet must be V2.0.0.121-eff0e01c0-aliyun or later.

```
apiVersion: v1
kind: Secret
metadata:
 name: vk-accesskey
 namespace: kube-system
type: Opaque
data:
 # Specify the Base64-encoded AccessKey ID.
 accesskey.id: {{ ECI_ACCESS_KEY Base64 Encoding }}
 # Specify the Base64-encoded AccessKey secret.
 accesskey.secret: {{ ECI_SECRET_KEY Base64 Encoding }}
---
apiVersion: v1
kind: ConfigMap
metadata:
 name: eci-profile
 namespace: kube-system
data:
 # If the elastic container instance cannot be connected to the API server, set this parameter to true.
 # Specify whether to run pods of Kubernetes clusters on both the elastic container instance and traditi
onal servers. For example, if the offline self-managed Kubernetes cluster is not connected to a VPC, set t
his parameter to true. If the API server of the self-managed cluster and the elastic container instance ar
e deployed in different VPCs, set this parameter to true.
 enableHybridMode: "false"
 # Specify whether to enable the cluster IP feature of Kubernetes. If you want to use this feature, set thi
s parameter to true.
 enableClusterIp: "true"
 # Specify whether to enable PrivateZone. If you want to use this feature, set this parameter to true.
 enablePrivateZone: "false"
 # Specify the ID of the resource group. If you do not want to use this feature, set this parameter to "".
 resourceGroupId: ""
 # Specify the required security group ID.
 securityGroupId: "sg-2ze**********"
 # Specify the scheduling selector to define which pods are automatically scheduled to the elastic conta
iner instance. For more information, see Configure an Elastic Container Instance profile.
```

```
iner instance. For more information, see Configure an Elastic Container Instance profile.
 selectors: ""
 # Specify the required vSwitch IDs.
 vSwitchIds: "vsw-2zeq**********"
 # Specify the required VPC ID.
 vpcId: "vpc-2ze0z**********"
---
apiVersion: v1
kind: Service
metadata:
 labels:
  role: webhook
 name: vk-webhook
 namespace: kube-system
spec:
 ports:
 - port: 443
  targetPort: 443
 selector:
  app: virtual-kubelet
---
apiVersion: v1
kind: ServiceAccount
metadata:
 name: virtual-kubelet
 namespace: kube-system
---
kind: ClusterRoleBinding
apiVersion: rbac.authorization.k8s.io/v1beta1
metadata:
 name: virtual-kubelet
subjects:
- kind: ServiceAccount
 name: virtual-kubelet
 namespace: kube-system
roleRef:
 kind: ClusterRole
 name: cluster-admin
 apiGroup: rbac.authorization.k8s.io
---
apiVersion: apps/v1
kind: Deployment
metadata:
 name: ack-virtual-node-controller
 namespace: kube-system
 labels:
  app: virtual-kubelet
spec:
 replicas: 1
 selector:
  matchLabels:
   app: virtual-kubelet
 template:
  metadata:
   labels:
```

```
    labels:
      app: virtual-kubelet
  spec:
    affinity:
      nodeAffinity:
        requiredDuringSchedulingIgnoredDuringExecution:
          nodeSelectorTerms:
          - matchExpressions:
            - key: type
              operator: NotIn
              values:
              - virtual-kubelet
      podAntiAffinity:
        preferredDuringSchedulingIgnoredDuringExecution:
        - podAffinityTerm:
            labelSelector:
              matchExpressions:
              - key: app
                operator: In
                values:
                - virtual-node-eci
            topologyKey: kubernetes.io/hostname
          weight: 100
    containers:
    - name: vk
      # Replace region-id with the ID of the region where the Virtual Kubelet is located. Replace vk-tag wit
h the latest Virtual Kubelet version. You can find the latest Virtual Kubelet version in the version record
of Virtual Kubelet.
      # Note: After you deploy Virtual Kubelet, pay attention to the release of Virtual Kubelet version and
update Virtual Kubelet in a timely manner to obtain continuous support from Alibaba Cloud.
      image: registry-vpc.{{ region-id }}.aliyuncs.com/acs/virtual-nodes-eci:{{ vk-tag }}
      imagePullPolicy: IfNotPresent
      args:
      - --provider
      - alibabacloud
      - --nodename
      - virtual-kubelet
      resources:
        requests:
          memory: 30M
          cpu: 100m
      env:
      # The CLUSTER TYPE parameter must be specified. If your Kubernetes cluster is created on ECS, set
this parameter to CLOUD. For on-premises self-managed clusters, set this parameter to IDC.
      -name: CLUSTER_TYPE
        value: {{ cluster-type }}
      - name: WEBHOOK
        value: "true"
      - name: VKUBELET_TAINT_KEY
        value: "virtual-kubelet.io/provider"
      - name: VKUBELET_TAINT_VALUE
        value: "alibabacloud"
      - name: VKUBELET_TAINT_EFFECT
        value: "NoSchedule"
      # Replace region-id with the ID of the region where the elastic container instance is located, such as
```

```
cn-beijing. You must specify this parameter.
    - name: ECI_REGION
      value: {{ region-id }}
    - name: ECI_QUOTA_CPU
      value: "1000000"
    - name: ECI_QUOTA_MEMORY
      value: 6400Ti
    - name: ECI_QUOTA_POD
      value: "10000"
    - name: ECI_KUBE_PROXY
      value: "true"
    - name: ECI_ACCESS_KEY
      valueFrom:
        secretKeyRef:
          name: vk-accesskey
          key: accesskey.id
    - name: ECI_SECRET_KEY
      valueFrom:
        secretKeyRef:
          name: vk-accesskey
          key: accesskey.secret
    - name: ALIYUN_CLUSTERID
      # Replace cluster-id with the ID of the self-managed Kubernetes cluster. The ID of each Kubernetes
cluster in an Alibaba Cloud account must be unique. You must specify this parameter.
      value: {{ cluster-id }}
    - name: KUBELET_PORT
      value: "10250"
    - name: VKUBELET_POD_IP
      valueFrom:
        fieldRef:
          apiVersion: v1
          fieldPath: status.podIP
  terminationGracePeriodSeconds: 60
  serviceAccountName: virtual-kubelet
```

2. Deploy Virtual Kubelet.

```
kubectl apply -f vk.yaml
```

3. View the deployment status.

```
kubectl get deploy/virtual-node-controller -n kube-system
```

The following code provides an example of the command output if Virtual Kubelet is deployed:

```
NAME                 READY  UP-TO-DATE  AVAILABLE  AGE
virtual-node-controller  1/1     1           1          161m
```

4. View the node information after Virtual Kubelet is deployed.

```
kubectl get node -o wide
```

After Virtual Kubelet is deployed, the generated virtual node is named virtual-kubelet. If the state of virtual-kubelet is Ready, Virtual Kubelet is deployed. The following code provides an example of the command output if Virtual Kubelet is deployed.

```
NAME         STATUS  ROLES   AGE  VERSION   INTERNAL-IP  EXTERNAL-IP  OS-IMAGE          KERNEL-
VERSION          CONTAINER-RUNTIME
k8s-master01   Ready   <none>  50d  v1.14.2   192.168.*.*  <none>       CentOS Linux 7 (Core)  3.10.0
-957.21.3.el7.x86_64  docker://18.9.6
k8s-master02   Ready   <none>  50d  v1.14.2   192.168.*.*  <none>       CentOS Linux 7 (Core)  3.10.0
-957.21.3.el7.x86_64  docker://18.9.6
k8s-master03   Ready   <none>  50d  v1.14.2   192.168.*.*  <none>       CentOS Linux 7 (Core)  3.10.0
-957.21.3.el7.x86_64  docker://18.9.6
virtual-kubelet  Ready   agent   82m  v1.11.2   172.30.*.*   <none>       <unknown>          <unknow
n>             <unknown>
```

References

After you connect the elastic container instance to the Kubernetes cluster, you may need to schedule pods to run on the elastic container instance or use the features provided by Elastic Container Instance. For more information about how to use Elastic Container Instance, see Overview.

1.5. Use Elastic Container Instance in offline Kubernetes clusters

If you have self-managed Kubernetes clusters in your on-premises data centers or other offline environments, you must deploy virtual nodes (Virtual Kubelet) in the clusters before you can use Elastic Container Instance in the clusters. This topic describes how to connect offline self-managed clusters to Elastic Container Instance, including how to register self-managed clusters in the Container Service for Kubernetes (ACK) console and deploy virtual nodes, or deploy Virtual Kubelet in self-managed clusters.

Prerequisites

An Alibaba Cloud account is created. For more information, see Sign up with Alibaba Cloud.

Background information

You can connect an elastic container instance to a Kubernetes cluster through the open source Virtual Kubelet implementation. This allows you to implement high elasticity for your Kubernetes cluster. To connect an elastic container instance to a self-managed Kubernetes cluster, use the following methods:

- Method 1: Register the cluster in the ACK console and deploy virtual nodes
- Method 2: Deploy Virtual Kubelet in the Kubernetes cluster

> **Note**
>
> We recommend that you use Method 1. This method allows you to update the Virtual Kubelet version (the ack-virtual-node component version) to use new features of elastic container instances. For more information, see Update Virtual Kubelet.

Limits

The following limits apply to hybrid cloud environments:

- Annotations update is not supported.
- Labels update is not supported.
- Spec.ActiveDeadlineSeconds update is not supported.
- ConfigMap and Secret update is not supported.
- The kubectl logs-f command is not supported. The kubectl logs command is supported.
- The kubectl attach command is not supported.
- The kubectl port-forward command is not supported.
- Some parameters of Downward API such as status.hostIP are unavailable.
- ClusterIP

  You must establish connections between offline clusters and Alibaba Cloud networks by using Express Connect or Smart Access Gateway. For more information, see Express Connect or Smart Access Gateway.

Preparations

You must obtain the information of the following parameters before you deploy virtual nodes.

Parameter	Description	Method to obtain the parameter value
ECI_ACCESS_KEY and ECI_SECRET_KEY	The AccessKey ID and corresponding AccessKey secret. The values of these parameters are the credentials used to manage the elastic container instance in virtual nodes.	For more information, see Obtain an AccessKey pair.
ALIYUN_CLUSTERID	The ID of the cluster, which is the unique identifier of the cluster.	• If you use Method 1, the cluster ID is generated by the system. • If you use Method 2, you must enter the cluster ID. The cluster ID consists of letters and digits. Make sure that the specified cluster ID is unique within the Alibaba Cloud account.
ECI_REGION	The region ID. The elastic container instance is deployed in this region.	You can query the regions where Elastic Container Instance is available by using the Elastic Container Instance console or by calling the DescribeRegions operation.

Parameter	Description	Method to obtain the parameter value
ECI_VPC	The ID of the virtual private cloud (VPC). The elastic container instance is deployed within this VPC.	You can create and view VPCs on the **VPCs** page in the VPC console.
ECI_VSWITCH	The ID of the vSwitch. The elastic container instance is associated with this vSwitch.	You can create and view vSwitches on the **vSwitch** page in the VPC console and select a corresponding vSwitch based on the selected VPC.
ECI_SECURITY_GROUP	The ID of the security group. The elastic container instance is added to this security group.	You can create and view security groups on the **Security Groups** page in the ECS console and select a corresponding security group based on the selected VPC.

Method 1: Register the cluster in the ACK console and deploy virtual nodes

You can register your self-managed cluster in the ACK console and then deploy virtual nodes in the cluster to use elastic container instances. Perform the following steps:

1. Log on to the ACK console.
2. In the left-side navigation pane, click **Clusters**.
3. Create a Kubernetes cluster and register it on the ACK console.
    i. On the **Clusters** page, click **Create Kubernetes Cluster**.

ii. Click the **Register Cluster** tab. Specify the parameters for the cluster and click **Create Cluster**.

The following table describes the parameters required to create a Kubernetes cluster. For more information, see Register an external Kubernetes cluster.

Parameter	Description
Region, VPC, and vSwitch	Select a region, VPC, and vSwitch for the cluster.
Access to API Server	Select a Server Load Balancer (SLB) instance type to use for the API server. By default, an internal-facing SLB instance is created for the API server.
EIP	Specify whether to create and associate an elastic IP address (EIP) to connect to the cluster.
Security group	A security group is automatically created to isolate security domains and control network traffic.
Deletion Protection	Specify whether to enable deletion protection for the cluster. If you enable deletion protection, the cluster cannot be deleted by using the ACK console or by calling API operations.

4. Register the cluster.
   i. On the **Clusters** page, find the cluster that you created and click the cluster name.
   ii. On the **Cluster Information** page, click the **Connection Information** tab.
   iii. Create a ConfigMap in the cluster.

   Click the **Public Network** or **Internal Network** tab based on your network. Copy the content to a YAML configuration file such as agent.yaml and run the `kubectl apply -f agent.yaml` command in the cluster to create a ConfigMap.
   iv. Run the following command in the cluster to check the connection status:

```
kubectl -n kube-system get pod |grep ack-cluster-agent
```

   The following command output is returned:

```
ack-cluster-agent-5f7d568f6-6fc4k       1/1   Running  0      9s
ack-cluster-agent-5f7d568f6-tf6fp       1/1   Running  0      9s
```

5. Deploy virtual nodes.
   i. On the **Clusters** page, click the name of the cluster in which you want to deploy a virtual node.

ii. In the left-side navigation pane of the details page, choose **Operations > Add-ons**.

iii. Click the **Others** tab, find ack-virtual-node, and then click **Install**.

Method 2: Deploy Virtual Kubelet in the Kubernetes cluster

You can deploy Virtual Kubelet in your clusters to use elastic container instances. You must deploy the latest version of Virtual Kubelet. For more information, see ack-virtual-node.

Perform the following steps to deploy Virtual Kubelet:

1. Prepare the vk.yaml configuration file required to deploy Virtual Kubelet.

   The following code provides an example of a YAML configuration file. You must delete the comments and replace the parameter values with the actual values based on your business requirements.

   > **Notice**
   >
   > The version of Virtual Kubelet must be V2.0.0.121-eff0e01c0-aliyun or later.

```
apiVersion: v1
kind: Secret
metadata:
 name: vk-accesskey
 namespace: kube-system
type: Opaque
data:
 # Specify the Base64-encoded AccessKey ID.
 accesskey.id: {{ ECI_ACCESS_KEY Base64 Encoding }}
 # Specify the Base64-encoded AccessKey secret.
 accesskey.secret: {{ ECI_SECRET_KEY Base64 Encoding }}
---
apiVersion: v1
kind: ConfigMap
metadata:
 name: eci-profile
 namespace: kube-system
data:
 # If the elastic container instance cannot be connected to the API server, set this parameter to true.
 # Specify whether to run pods of Kubernetes clusters on both the elastic container instance and traditi
onal servers. For example, if the offline self-managed Kubernetes cluster is not connected to a VPC, set t
his parameter to true. If the API server of the self-managed cluster and the elastic container instance ar
e deployed in different VPCs, set this parameter to true.
 enableHybridMode: "false"
 # Specify whether to enable the cluster IP feature of Kubernetes. If you want to use this feature, set thi
s parameter to true.
 enableClusterIp: "true"
 # Specify whether to enable PrivateZone. If you want to use this feature, set this parameter to true.
 enablePrivateZone: "false"
 # Specify the ID of the resource group. If you do not want to use this feature, set this parameter to "".
 resourceGroupId: ""
 # Specify the required security group ID.
 securityGroupId: "sg-2ze**********"
 # Specify the scheduling selector to define which pods are automatically scheduled to the elastic conta
iner instance. For more information, see Configure an Elastic Container Instance profile.
```

```
iner instance. For more information, see Configure an Elastic Container Instance profile.
 selectors: ""
 # Specify the required vSwitch IDs.
 vSwitchIds: "vsw-2zeq**********"
 # Specify the required VPC ID.
 vpcId: "vpc-2ze0z**********"
---
apiVersion: v1
kind: Service
metadata:
 labels:
  role: webhook
 name: vk-webhook
 namespace: kube-system
spec:
 ports:
 - port: 443
  targetPort: 443
 selector:
  app: virtual-kubelet
---
apiVersion: v1
kind: ServiceAccount
metadata:
 name: virtual-kubelet
 namespace: kube-system
---
kind: ClusterRoleBinding
apiVersion: rbac.authorization.k8s.io/v1beta1
metadata:
 name: virtual-kubelet
subjects:
- kind: ServiceAccount
 name: virtual-kubelet
 namespace: kube-system
roleRef:
 kind: ClusterRole
 name: cluster-admin
 apiGroup: rbac.authorization.k8s.io
---
apiVersion: apps/v1
kind: Deployment
metadata:
 name: ack-virtual-node-controller
 namespace: kube-system
 labels:
  app: virtual-kubelet
spec:
 replicas: 1
 selector:
  matchLabels:
   app: virtual-kubelet
 template:
  metadata:
   labels:
```

```
    labels:
      app: virtual-kubelet
  spec:
    affinity:
      nodeAffinity:
        requiredDuringSchedulingIgnoredDuringExecution:
          nodeSelectorTerms:
          - matchExpressions:
            - key: type
              operator: NotIn
              values:
              - virtual-kubelet
      podAntiAffinity:
        preferredDuringSchedulingIgnoredDuringExecution:
        - podAffinityTerm:
            labelSelector:
              matchExpressions:
              - key: app
                operator: In
                values:
                - virtual-node-eci
            topologyKey: kubernetes.io/hostname
          weight: 100
    containers:
    - name: vk
      # Replace region-id with the ID of the region where the Virtual Kubelet is located. Replace vk-tag wit
h the latest Virtual Kubelet version. You can find the latest Virtual Kubelet version in the version record
of Virtual Kubelet.
      # Note: After you deploy Virtual Kubelet, pay attention to the release of Virtual Kubelet version and
update Virtual Kubelet in a timely manner to obtain continuous support from Alibaba Cloud.
      image: registry-vpc.{{ region-id }}.aliyuncs.com/acs/virtual-nodes-eci:{{ vk-tag }}
      imagePullPolicy: IfNotPresent
      args:
      - --provider
      - alibabacloud
      - --nodename
      - virtual-kubelet
      resources:
        requests:
          memory: 30M
          cpu: 100m
      env:
      # The CLUSTER TYPE parameter must be specified. If your Kubernetes cluster is created on ECS, set
this parameter to CLOUD. For on-premises self-managed clusters, set this parameter to IDC.
      -name: CLUSTER_TYPE
        value: {{ cluster-type }}
      - name: WEBHOOK
        value: "true"
      - name: VKUBELET_TAINT_KEY
        value: "virtual-kubelet.io/provider"
      - name: VKUBELET_TAINT_VALUE
        value: "alibabacloud"
      - name: VKUBELET_TAINT_EFFECT
        value: "NoSchedule"
      # Replace region-id with the ID of the region where the elastic container instance is located, such as
```

```
cn-beijing. You must specify this parameter.
    - name: ECI_REGION
      value: {{ region-id }}
    - name: ECI_QUOTA_CPU
      value: "1000000"
    - name: ECI_QUOTA_MEMORY
      value: 6400Ti
    - name: ECI_QUOTA_POD
      value: "10000"
    - name: ECI_KUBE_PROXY
      value: "true"
    - name: ECI_ACCESS_KEY
      valueFrom:
        secretKeyRef:
          name: vk-accesskey
          key: accesskey.id
    - name: ECI_SECRET_KEY
      valueFrom:
        secretKeyRef:
          name: vk-accesskey
          key: accesskey.secret
    - name: ALIYUN_CLUSTERID
      # Replace cluster-id with the ID of the self-managed Kubernetes cluster. The ID of each Kubernetes
cluster in an Alibaba Cloud account must be unique. You must specify this parameter.
      value: {{ cluster-id }}
    - name: KUBELET_PORT
      value: "10250"
    - name: VKUBELET_POD_IP
      valueFrom:
        fieldRef:
          apiVersion: v1
          fieldPath: status.podIP
  terminationGracePeriodSeconds: 60
  serviceAccountName: virtual-kubelet
```

2. Deploy Virtual Kubelet.

```
kubectl apply -f vk.yaml
```

3. View the deployment status.

```
kubectl get deploy/virtual-node-controller -n kube-system
```

The following code provides an example of the command output if Virtual Kubelet is deployed:

```
NAME                 READY  UP-TO-DATE  AVAILABLE  AGE
virtual-node-controller  1/1     1           1          161m
```

4. View the node information after Virtual Kubelet is deployed.

```
kubectl get node -o wide
```

After Virtual Kubelet is deployed, the generated virtual node is named virtual-kubelet. If the state of virtual-kubelet is Ready, Virtual Kubelet is deployed. The following code provides an example of the command output if Virtual Kubelet is deployed.

```
NAME          STATUS  ROLES   AGE  VERSION   INTERNAL-IP  EXTERNAL-IP  OS-IMAGE          KERNEL-
VERSION          CONTAINER-RUNTIME
k8s-master01  Ready   <none>  50d  v1.14.2   192.168.*.*  <none>       CentOS Linux 7 (Core)  3.10.0
-957.21.3.el7.x86_64  docker://18.9.6
k8s-master02  Ready   <none>  50d  v1.14.2   192.168.*.*  <none>       CentOS Linux 7 (Core)  3.10.0
-957.21.3.el7.x86_64  docker://18.9.6
k8s-master03  Ready   <none>  50d  v1.14.2   192.168.*.*  <none>       CentOS Linux 7 (Core)  3.10.0
-957.21.3.el7.x86_64  docker://18.9.6
virtual-kubelet  Ready  agent  82m  v1.11.2   172.30.*.*   <none>       <unknown>          <unknow
n>            <unknown>
```

References

After you connect the elastic container instance to the Kubernetes cluster, you may need to schedule pods to run on the elastic container instance or use the features provided by Elastic Container Instance. For more information about how to use Elastic Container Instance, see Overview.

1.6. FAQ related to Virtual Kubelet deployment in an on-premises Kubernetes cluster

To use an elastic container instance together with an on-premises Kubernetes cluster, you must deploy Virtual Kubelet in the Kubernetes cluster. Then, you can connect the elastic container instance to the cluster through Virtual Kubelet. This topic describes the commonly asked questions about Virtual Kubelet deployment in an on-premises Kubernetes cluster.

- FAQ related to networks
  - How do cloud services access the IP addresses of on-premises pods?
  - How do on-premises services access the IP addresses of cloud pods?
  - What do I do if an internal network domain name cannot be resolved?
  - Why on-premises services cannot access cloud services?
- FAQ related to image pulling
  - What do I do if I cannot pull images from a self-managed container image repository?
- FAQ related to pod scheduling
  - How do I schedule pods to Virtual Kubelet?
  - Why do DaemonSet pods remain in the Pending state after they are scheduled to Virtual Kubelet?
  - Why does the scheduling fail when I attempt to schedule pods to Virtual Kubelet by configuring pod labels?

How do cloud services access the IP addresses of on-premises pods?

If you use leased lines of Express Connect to connect your cloud and on-premises networks, the cloud and on-premises services can learn routing rules from each other by using Border Gateway Protocol (BGP). Then, the on-premises equipment can broadcast the pod IP addresses to the cloud service by using BGP. As a result, the cloud service can access the IP addresses of the on-premises pods. For more information, see Configure BGP.

How do on-premises services access the IP addresses of cloud pods?

If you use leased lines of Express Connect to connect your cloud and on-premises networks, the cloud and on-premises services can learn routing rules from each other by using BGP. You can deploy a cloud controller manager (CCM) to automatically synchronize pod IP addresses on the cloud to the virtual private cloud (VPC) route table. For more information about a CCM, see Cloud Controller Manager.

After you deploy a CCM in a self-managed or an on-premises cluster, you can synchronize the routing IP addresses of the Kubernetes pods to the VPC route table. When you deploy the CCM, take note of the following items:

- Change the value of providerID of the Kubernetes cluster nodes to the `<region-id>.<ecs-id>` format, for example, `cn-shanghai.i-ankb8zjh2nzchf*******`.
- Ensure that the pod IP addresses of the cluster nodes are all within the pod CIDR blocks of the nodes. For example, you must configure the Calico IPAM configuration file as the `host-local` type. This configuration specifies that the pod CIDR field of Kubernetes cluster nodes is obtained from the Kubernetes API. This ensures all the pod IP addresses of the cluster nodes are within the pod CIDR blocks of the nodes.

  You can check the pod CIDR blocks in the spec data of the nodes.

```
spec:
 podCIDR: 172.23.XX.0/26
 podCIDRs:
 - 172.23.XX.0/26
 providerID: cn-shanghai.i-ankb8zjh2nzchfxxxxxxx
```

What do I do if an internal network domain name cannot be resolved?

Problem description

Cloud and on-premises services cannot mutually invoke services because the internal network domain names of the services cannot be resolved. The failure to resolve internal network domain names includes:

- Cloud services cannot resolve the internal network domain names of on-premises networks.
- On-premises services cannot resolve cloud PrivateZone domain names.

Solution

On-premises solutions and Alibaba Cloud VPC are deployed in different network environments. If cloud and on-premises services can communicate each other only after the internal network domain names are resolved by using Alibaba Cloud DNS, you can configure Alibaba Cloud DNS PrivateZone to resolve the internal network domain names. For more information, see Use Alibaba Cloud DNS PrivateZone and VPN Gateway to allow ECS instances in a VPC to access an on-premises DNS.

Why on-premises services cannot access cloud services?

Problem description

On-premises services cannot access Alibaba Cloud services such as ApsaraDB RDS, Object Storage Service (OSS), and Log Service.

Solutions

You can use one of the following solutions to solve the problem. We recommend that you use Solution 1.

- Solution 1

  Configure the domain name of the cloud service on the cloud. Then, the virtual border router (VBR) publishes the route to the on-premises network over BGP. For more information, see Access cloud services.

- Solution 2: Add a static route to the on-premises network to route 100.64.0.0/10 to the leased line.

What do I do if I cannot pull images from a self-managed container image repository?

Problem description

When I try to pull images from a self-managed container image repository, the following error is reported:

```
Failed to pull image "harbor.example.com/default/nginx:latest": rpc error: code = Unknown
desc = failed to pull and unpack image "ha[redacted]ample.com/default/nginx:latest": failed
to resolve reference "har[redacted]ple.com/default/nginx:latest": failed to do request: Head
"https://harb[redacted]ple.com/default/nginx/manifests/latest": x509: certificate is valid
for ingress.local, not harbor-beta.cm.service.163.org
```

Solution

The reason of the preceding problem is that the image repository uses certificates that are issued by you. The certificates that are issued by you are ineligible. Therefore, the certificate-based authentication fails when you pull images. When you create a pod, you can add the following annotation to skip the certificate-based authentication:

```
"k8s.aliyun.com/insecure-registry": "<host-name>"
```

For example, if the endpoint of the NGINX image in the private image repository is `test.example.com/test/nginx:apline`, you can add the `"k8s.aliyun.com/insecure-registry": "test.example.com"` annotation to skip the certificate-based authentication.

How do I schedule pods to Virtual Kubelet?

You can select one of the following methods to schedule pods to Virtual Kubelet based on your business requirements. Then, the pods that are scheduled to Virtual Kubelet are run on your elastic container instance.

- Manually schedule pods

You can manually schedule pods to make pods run on your elastic container instance by configuring namespace labels, pod labels, or elastic scheduling of Elastic Container Instance. For more information, see Schedule pods to elastic container instances.

- Automatically schedule pods

  Elastic Container Instance Profile provides the Elastic Container Instance Scheduler feature to implement a new scheduling mechanism based on Mutating Webhook. You can declare the namespace or pod labels to be matched in Elastic Container Instance Profile. Pods that have the declared labels are automatically scheduled to elastic container instances. For more information, see Configure Elastic Container Instance Profile.

Why do DaemonSet pods remain in the Pending state after they are scheduled to Virtual Kubelet?

Virtual Kubelet creates a virtual node and does not support DaemonSets. When you create a DaemonSet, you must configure an anti-affinity scheduling policy to prevent DaemonSet pods from being scheduled to Virtual Kubelet. The following sample code shows how to configure an anti-affinity scheduling policy.

```
spec:
 affinity:
  nodeAffinity:
   requiredDuringSchedulingIgnoredDuringExecution:
    nodeSelectorTerms:
    - matchExpressions:
     - key: type
       operator: NotIn
       values:
       - virtual-kubelet
```

Why does the scheduling fail when I attempt to schedule pods to Virtual Kubelet by configuring pod labels?

This problem occurs when the version of your Kubernetes cluster is earlier than v1.16.

```
// MutatingAdmission not support ObjectSelector if Kubernetes version is less than 1.16
```

1.7. Update Virtual Kubelet

Elastic Container Instance uses Virtual Kubelet to deploy virtual nodes and then uses the virtual nodes to connect to Kubernetes. Virtual Kubelet is provided and maintained by the Kubernetes community. To use Elastic Container Instance features, you must keep Virtual Kubelet up-to-date. This topic describes how to update Virtual Kubelet.

Background information

Different types of Kubernetes clusters require different operations to update Virtual Kubelet.

- For Serverless Kubernetes (ASK) clusters, the system automatically updates Virtual Kubelet.
- For Container Service for Kubernetes (ACK) clusters, the update method of Virtual Kubelet varies based on the edition of the clusters. For managed ACK clusters, the system automatically updates Virtual Kubelet. For dedicated ACK clusters, you must manually update Virtual Kubelet.

- For self-managed Kubernetes clusters, you must manually update Virtual Kubelet.

You can use one of the following methods to manually update Virtual Kubelet based on the type of your Kubernetes clusters and the deployment mode of Virtual Kubelet.

- Update ack-virtual-node in the ACK console

  This method for updating ack-virtual-node is applicable to dedicated ACK clusters and self-managed clusters that are registered in the ACK console.

- Run Kubelet commands to modify the configuration file of Virtual Kubelet

  For self-managed Kubernetes clusters that are connected by deploying Virtual Kubelet you can modify the YAML configuration file of Virtual Kubelet to update the tags of Virtual Kubelet images to the latest version. For more information about versions of Virtual Kubelet, see ack-virtual-node.

Update ack-virtual-node in the ACK console

1. Log on to the ACK console.
2. In the left-side navigation pane, click **Clusters**.
3. On the **Clusters** page, click the name of the cluster for which you want to update ack-virtual-node.
4. In the left-side navigation pane of the details page, choose **Operations > Add-ons**.
5. Find ack-virtual-node and click **Upgrade**.

Run Kubelet commands to modify the configuration file of Virtual Kubelet

1. Use Kubelet to connect your Kubernetes cluster.
2. Run the following command to modify the YAML configuration file of Virtual Kubelet:

   ```
   kubectl edit deployment -n kube-system virtual-node-controller
   ```

   Example of YAML configuration files:

```
apiVersion: apps/v1
kind: Deployment
metadata:
 name: virtual-node-controller
 namespace: kube-system
 labels:
  app: virtual-node-controller
spec:
 replicas: 1
 selector:
  matchLabels:
   app: virtual-node-controller
 template:
  metadata:
   labels:
    app: virtual-node-controller
  spec:
   containers:
   - name: virtual-node-controller
    image: registry.cn-beijing.aliyuncs.com/acs/virtual-nodes-eci:v2.******-aliyun  #Change the tag of th
e image to the latest version.
 env:
     - name: WEBHOOK
       value: "true"
```

2.Use Elastic Container Instance

2.1. Overview

After you use Virtual Kubelet to connect Elastic Container Instance to a Kubernetes cluster, you can schedule pods to run on Elastic Container Instance and configure annotations to use the features of Elastic Container Instance.

Schedule pods to Elastic Container Instance

For Serverless Kubernetes (ASK) clusters, the entire clusters run on Elastic Container Instance, so you do not need to schedule pods to Elastic Container Instance. For Kubernetes clusters that use both Elastic Container Instance and regular nodes, you can schedule pods to run on Elastic Container Instance. You can schedule pods to Elastic Container Instance in the following scenarios:

- Manually schedule pods to Elastic Container Instance

  You can manually schedule pods to run on Elastic Container Instance by configuring namespace labels, pod labels, or Elastic Container Instance elastic scheduling. For more information, see Schedule pods to Elastic Container Instance.

- Automatically schedule pods to Elastic Container Instance

  Elastic Container Instance profiles provide the Elastic Container Instance Scheduler feature to implement a new scheduling mechanism based on the Mutating Webhook mechanism. You can declare the namespace or pod labels to be matched in Elastic Container Instance profiles. Pods that have the declared labels are automatically scheduled to Elastic Container Instance. For more information, see Configure an Elastic Container Instance profile.

- Automatically re-schedule pods that fail to be scheduled to Elastic Container Instance

  After the virtual-kubelet-autoscaler add-on is deployed in Container Service for Kubernetes (ACK) clusters, the add-on automatically re-schedules pods that fail to be scheduled due to insufficient ACK resources to Elastic Container Instance. For more information, see Install the virtual-kubelet-autoscaler add-on in an ACK cluster.

- Automatically schedule pods with elastic workloads to ECI

  After the ack-kubernetes-elastic-workload add-on is deployed in an ACK cluster, the add-on controls the fine-grained distribution of loads such as Deployment and StatefulSet at the pod level. The add-on schedules the scaled-out pods to run on Elastic Container Instance to implement a more fine-grained scheduling policy. For more information, see Deploy and use ack-kubernetes-elastic-workload in an ACK cluster.

Use features of Elastic Container Instance

When you schedule pods of a Kubernetes cluster to Elastic Container Instance, you can add annotations to the pods to make full use of the features of Elastic Container Instance. Make sure that the annotations that you want to add comply with the Kubernetes syntax. Annotations must be added to the metadata of pods. The following table describes common annotations:

> **Note**
>
> You can manually add annotations when you schedule pods. Alternatively, you can configure an Elastic Container Instance profile. Then, you can use the Elastic Container Instance Effect feature of the Elastic Container Instance profile to automatically append annotations to pods that have the labels. For information about how to configure Elastic Container Instance Effect, see Configure an Elastic Container Instance profile.

Annotation	Example	Description
k8s.aliyun.com/eci-vswitch	vsw-bp1xpiowfm5vo8o3c****	Specifies the IDs of vSwitches. You can specify multiple vSwitches to specify multiple zones.
k8s.aliyun.com/eci-use-specs	2-4Gi,4-8Gi,ecs.c6.xlarge	Specifies the elastic container instance types. You can specify multiple elastic container instance types. An elastic container instance type can be a combination of vCPUs and memory or an ECS instance type.
k8s.aliyun.com/eci-image-cache	"true"	Specifies whether to automatically match the image cache.
k8s.aliyun.com/eci-with-eip	"true"	Specifies whether to automatically create and associate an Elastic IP Address (EIP).

For information about annotations supported by Elastic Container Instance and configuration examples of the annotations, see Pod annotations supported by Elastic Container Instance.

> **Note**
>
> If exceptions occur when you use the features of Elastic Container Instance, you can view the event information. For more information, see Custom events of Elastic Container Instance.

2.2. Schedule pods to Elastic Container Instance

For Serverless Kubernetes (ASK) clusters, the entire clusters run on Elastic Container Instance, so you do not need to schedule pods to Elastic Container Instance. For Container Service for Kubernetes (ACK) clusters or self-managed Kubernetes clusters where ECI and regular nodes are used, you can schedule pods to elastic container instances that are deployed on virtual nodes. This topic describes how to schedule a pod to Elastic Container Instance.

Overview of scheduling methods

When you use Elastic Container Instance together with regular nodes, you can use the following methods to schedule a pod to Elastic Container Instance:

> **Note**
>
> To use the following methods, you must modify existing resources. These modifications may cause vulnerabilities in your system. We recommend that you configure Elastic Container Instance Profile. Elastic Container Instance Profile provide the Elastic Container Instance Scheduler feature. When you use this feature, you can declare the pod or namespace labels to be matched in the eci-profile configuration file. If a pod has the declared labels, the pod is automatically scheduled to Elastic Container Instance. For more information, see Configure Elastic Container Instance Profile.

- Configure pod labels

  If you have several pods that need to be scheduled to run on elastic container instances, you can add labels to the pods. Then, the pods run on the elastic container instances that are connected in the virtual node.

- Configure namespace labels

  If you have many pods that you want to run on elastic container instances, you can create a namespace and add a label to the namespace. Then, all pods in the namespace are scheduled to run on the elastic container instances in the virtual node.

- Configure Elastic Container Instance elastic scheduling

  Elastic Container Instance elastic scheduling is an elastic scheduling policy provided by Alibaba Cloud. When you deploy services, you can add annotations to declare that only the resources of regular nodes or virtual nodes are used, or when the resources of regular nodes are insufficient, Elastic Container Instance resources are automatically used. This policy can be used to meet the different requirements for elastic resources in different scenarios.

- (Not recommended) Other methods such as configuring the node selector and tolerations and specifying a node name

  When Virtual Kubelet exists in a Kubernetes cluster, you can schedule pods to Virtual Kubelet by configuring the node selector and tolerations or specifying a node name. Then, Virtual Kubelet creates elastic container instances to run the pods.

Method 1: Configure pod labels

You can add an `alibabacloud.com/eci=true` label to a pod to schedule the pod to run on elastic container instances.

> **Note**
>
> You can also add an `eci=true` label, but we do not recommend you to do this.

The following example shows the procedure on how to configure pod labels:

1. Create the YAML configuration file of the pod.

   ```
   vim test-pod.yaml
   ```

   An example of the content of test-pod.yaml:

```
apiVersion: v1
kind: Pod
metadata:
 name: nginx1
 labels:
  alibabacloud.com/eci: "true" # Add a label
spec:
 containers:
 - image: nginx
  imagePullPolicy: Always
  name: nginx
```

2. Create a pod.

```
kubectl apply -f test-pod.yaml
```

Method 2: Configure namespace labels

You can create a namespace and add an `alibabacloud.com/eci=true` label to the namespace. Then, all pods in the namespace are scheduled to run on elastic container instances.

> **Note**
>
> You can also add a `virtual-node-affinity-injection=enabled` label, but we do not recommend you to do this.

The following example shows the procedure on how to configure namespace labels:

1. Create a namespace.

```
kubectl create ns vk
```

2. Add a label to the namespace.

```
kubectl label namespace vk alibabacloud.com/eci=true
```

3. Create the YAML configuration file of the pod.

```
vim test-pod.yaml
```

An example of the content of test-pod.yaml:

```
apiVersion: v1
kind: Pod
metadata:
 name: nginx
 namespace: vk # Specify the namespace to which you added a label in Step 2.
spec:
 containers:
 - image: nginx
  imagePullPolicy: Always
  name: nginx
```

4. Create a pod.

```
kubectl apply -f test-pod.yaml
```

Method 3: Configure Elastic Container Instance elastic scheduling

You can add annotations to pod metadata to declare the type of the elastic resources that you want to use. The configuration item is alibabacloud.com/burst-resource. Valid values of the configuration item:

- eci: Elastic container instances are used when the resources on the regular nodes of the cluster are insufficient.
- eci_only: Only elastic container instances are used.

The following example shows the procedure on how to configure Elastic Container Instance elastic scheduling:

1. Create the YAML configuration file to use for deployment.

```
vim test-deployment.yaml
```

An example of the content of test-deploymen.yaml:

```
apiVersion: apps/v1
kind: Deployment
metadata:
 name: nginx
 labels:
  app: nginx
spec:
 replicas: 4
 selector:
  matchLabels:
   app: nginx
 template:
  metadata:
   name: nginx
   annotations:
    alibabacloud.com/burst-resource: eci # This annotation specifies the type of resource that you wan
t to use.
   labels:
    app: nginx
  spec:
   containers:
   - name: nginx
    image: nginx
    resources:
     limits:
      cpu: 2
     requests:
      cpu: 2
```

2. Create a deployment.

```
kubectl apply -f test-deployment.yaml
```

(Not recommended) Other methods

You can use other methods such as configuring the node selector and tolerations parameters or specifying node names to schedule pods to elastic container instances.

The following example shows the procedure on how to use other methods to schedule pods to Elastic Container Instance:

1. Create the YAML configuration file of the pod.

   ```
   vim test-pod.yaml
   ```

   An example of the content of test-pod.yaml:

   - Configure node selector and tolerations.

     ```
     apiVersion: v1
     kind: Pod
     metadata:
       name: nginx
     spec:
       containers:
       - image: nginx
         imagePullPolicy: Always
         name: nginx
       nodeSelector:
         type: virtual-kubelet
       tolerations:
       - key: virtual-kubelet.io/provider
         operator: Exists
     ```

   - Specify the node name

     ```
     apiVersion: v1
     kind: Pod
     metadata:
       name: nginx
     spec:
       containers:
       - image: nginx
         imagePullPolicy: Always
         name: nginx
       nodeName: virtual-kubelet
     ```

2. Create a pod.

   ```
   kubectl apply -f test-pod.yaml
   ```

2.3. Configure Elastic Container Instance Profile

When you deploy an elastic container instance in Kubernetes based on Virtual Kubelet, pods may fail to be properly scheduled to or run on the elastic container instance. You can configure Elastic Container Instance Profile to resolve the issue. This topic describes what is Elastic Container Instance Profile and how to configure the feature.

Introduction to Elastic Container Instance Profile

When you deploy an elastic container instance in Kubernetes based on Virtual Kubelet, pods may fail to be properly scheduled to the elastic container instance. After pods are scheduled to the elastic container instance, you need to add pod annotations for Elastic Container Instance features such as image cache to take effect. Typically, cluster administrators fix the issue that pods fail to be scheduled, and add pod annotations to make Elastic Container Instance features take effect. R&D personnel can adjust the pod configuration files to prevent the scheduling failure or add the pod annotations.

Elastic Container Instance Profile allows cluster administrators to schedule pods and modify pod configuration files. Elastic Container Instance Profile provides the following features:

- ECI Scheduler

  If elastic container instances are used together with regular nodes, you can configure pod labels, namespace labels, and Elastic Container Instance elastic scheduling to schedule pods to the elastic container instances. However, to perform these operations, you must modify existing resources. These modifications may expose your system to vulnerabilities.

  Elastic Container Instance Scheduler implements a new scheduling mechanism based on mutating webhooks. You can declare the namespace or pod labels to be matched in Elastic Container Instance Profile. Pods that have the declared labels are automatically scheduled to elastic container instances.

- ECI Effect

  You must add annotations or labels to pods for some Elastic Container Instance features to take effect. Examples of the features include specifying Elastic Compute Service (ECS) instance types, enabling image caches, and configuring the Network Time Protocol (NTP) service. For more information, see Use pod annotations to implement features of Elastic Container Instance.

  Elastic Container Instance Effect can automatically add annotations and labels. In Elastic Container Instance Profile, you can declare the namespace or pod labels to be matched and specify the annotations and labels to be added. If a pod has the declared labels, the specified annotations and labels are automatically added to the pod.

- Configure hot update

  In Elastic Container Instance Profile, you can configure the Cluster IP, hybrid cloud mode, PrivateZone, resource group, security group, virtual private cloud (VPC), and vSwitch. You can also modify the parameter settings in Elastic Container Instance Profile based on your business requirements. The new parameter settings take effect without the need to restart Virtual Kubelet.

Preparations

Before you use Elastic Container Instance Profile, make sure that your Virtual Kubelet is of the latest version, and that mutating webhooks are enabled if you want to use Elastic Container Instance Scheduler.

Make preparations based on the types of your Kubernetes clusters.

- Serverless Kubernetes (ASK) clusters

  Virtual Kubelet is automatically updated to the latest version.

> **Note**
>
> Pods on ASK clusters are automatically scheduled to elastic container instances. You do not need to use Elastic Container Instance Scheduler to schedule pods to elastic container instances.

- Container Service for Kubernetes (ACK) clusters
  - For managed ACK clusters, Virtual Kubelet is automatically updated to the latest version. Mutating webhooks are automatically enabled.
  - For non-managed ACK clusters, you must modify the configuration file of Virtual Kubelet to update Virtual Kubelet to the latest version, and then enable mutating webhooks.
- Other clusters

  You must modify the configuration file of Virtual Kubelet to update Virtual Kubelet to the latest version, and then enable mutating webhooks.

For information about how to update Virtual Kubelet, see Update Virtual Kubelet.

Parameter description

Elastic Container Instance Profile reads the eci-profile configuration file from the kube-system namespace and matches pods based on the selectors specified in the configuration file. If a pod has the labels contained in the selectors, the pod is automatically scheduled to the elastic container instance, or annotations and labels are automatically added to the pod for Elastic Container Instance features to take effect.

You can use one of the following methods to edit the eci-profile configuration file:

- Run the kubectl edit command.

```
kubectl edit configmap eci-profile -n kube-system
```

- Use the ACK console.
  i. On the Clusters page of the Container Service - Kubernetes console, find the cluster that you want to configure and click the cluster name to go to the cluster details page.
  ii. In the left-side navigation pane, choose **Configurations > ConfigMaps**.
  iii. Select the **kube-system** namespace.
  iv. Find and edit the eci-profile configuration file.

Example of the eci-profile configuration file:

```
apiVersion: v1
kind: ConfigMap
metadata:
 name: eci-profile
 namespace: kube-system
data:
 vpcId: "vpc-xxx"
 securityGroupId: "sg-xxx"
 vswitchIds: "vsw-111,vsw-222"
 enableClusterIp: "true"
 enableHybridMode: "false"
 enablePrivateZone: "false"
```

```
selectors: |
 [
   {
     "name":"default-selector-1",
     "objectSelector":{
       "matchLabels":{
         "alibabacloud.com/eci":"true"
       }
     }
   },
   {
     "name":"default-selector-2",
     "objectSelector":{
       "matchLabels":{
         "eci":"true"
       }
     }
   },
   {
     "name":"default-selector-3",
     "namespaceSelector":{
       "matchLabels":{
         "alibabacloud.com/eci":"true"
       }
     }
   },
   {
     "name":"default-selector-4",
     "namespaceSelector":{
       "matchLabels":{
         "eci":"true"
       }
     }
   },
   {
     "name":"default-selector-5",
     "namespaceSelector":{
       "matchLabels":{
         "virtual-node-affinity-injection":"enabled"
       }
     },
     "effect":{
     "annotations":{
       "k8s.aliyun.com/eci-image-cache": "true"
     },
     "labels":{
       "created-by-eci":"true"
     }
    }
   }
 ]
```

In the preceding example, data contains parameters such as vpcId, vswitchIds, and selectors.

Selectors contain the configurations of Elastic Container Instance Scheduler and Elastic Container Instance Effect. In each selector, you must declare the name of the selector and can declare the following fields based on your business requirements:

- namespaceSelector: the namespace labels to match.
- objectSelector: the pod labels to match.
- effect: the annotations and labels to be dynamically added.

After you configure selectors, you can run the following command to check whether the selectors take effect. If the returned YAML file contains the configured selectors, the selectors are configured. If the returned YAML file does not contain the configured selectors, check whether the format of the selectors is correct.

```
kubectl get mutatingwebhookconfigurations -o yaml vk-webhook
```

vpcId specifies the ID of the VPC. vswitchIds specifies the IDs of vSwitches. You can modify the parameters based on your business requirements. The following table describes the parameters that you can modify.

Parameter	Example	Description
enableClusterIp	"true"	Specifies whether to support Cluster IP.
enableHybridMode	"false"	Specifies whether to enable the hybrid cloud mode.
enablePrivateZone	"false"	Specifies whether to use PrivateZone for domain name resolution.
resourceGroupId	rg-aek2z3elfs4qghy	Specifies the ID of the resource group.
securityGroupId	sg-2ze0b9o8pjjzts4h****	Specifies the ID of the security group.
vSwitchIds	vsw-2zeet2ksvw7f14ryz****	Specifies the IDs of the vSwitches. Separate multiple IDs with commas (,).
vpcId	vpc-2zeghwzptn5zii0w7****	Specifies the ID of the VPC.

Example on how to configure Elastic Container Instance Scheduler

In the selectors of the eci-profile configuration file, you can declare the namespace or pod labels as arrays. The pods that have matching labels are automatically scheduled to elastic container instances.

The following sample code shows how to configure Elastic Container Instance Scheduler:

> **Notice**
>
> Remove annotations when you configure your Elastic Container Instance Scheduler.

```
selectors: |
 [
   {
    "name":"demo", #Required. The name of the selector.
    "namespaceSelector":{ #Optional. The namespace labels to be matched. You must specify namespaceSelector, objectSelector, or both.
       "matchLabels":{ #The labels to be matched. If you specify multiple labels, the specified labels have logical AND relations.
          "department":"bigdata"
       }
    },
    "objectSelector":{ #Optional. The pod labels to be matched. You must specify namespaceSelector, objectSelector, or both.
       "matchLabels":{ #The labels to be matched. If you specify multiple labels, the specified labels have logical AND relations.
          "type":"offline-task"
       }
    }
   }
 ]
```

> **Notice**
>
> In a selector, you must configure namespaceSelector, objectSelector, or both. If you configure both namespaceSelector and objectSelector, only pods that have all the labels specified in both namespaceSelector and objectSelector can be automatically scheduled to the elastic container instance.

In the preceding example, a selector named demo is declared to implement the following feature:

If the namespace of a pod has the department=bigdata label and the pod has the type=offline-task label, the pod is automatically scheduled to the elastic container instance.

Example on how to configure Elastic Container Instance Effect

In the selectors of the eci-profile configuration file, you can declare the namespace or pod labels as arrays and specify the annotations and labels to be dynamically added. If a pod has the declared labels, the specified annotations and labels are automatically added to the pod.

The following sample code shows how to configure Elastic Container Instance Effect:

> **Notice**
>
> Remove annotations when you configure your Elastic Container Instance Effect.

```
selectors: |
 [
  {
    "name":"demo", #Required. The name of the selector.
    "namespaceSelector":{ #Optional. The namespace labels to be matched.
      "matchLabels":{ #The labels to be matched. If you specify multiple labels, the specified labels have logical AND relations.
        "department":"bigdata"
      }
    },
    "objectSelector":{ #Optional. The pod labels to be matched.
      "matchLabels":{ #The labels to be matched. If you specify multiple labels, the specified labels have logical AND relations.
        "type":"offline-task"
      }
    },
    "effect":{ #The annotations and labels to be dynamically added.
      "annotations":{
        "k8s.aliyun.com/eci-image-cache": "true"
      },
      "labels":{
        "created-by-eci":"true"
      }
    }
  }
 ]
```

> **Notice**
>
> - In a selector, you can configure namespaceSelector and objectSelector based on your business requirements. If none of namespaceSelector and objectSelector is configured, the effect settings take effect on all the pods that are scheduled to the elastic container instance.
> - If you configure multiple selectors, the selectors are matched in sequence. After pods are matched, the annotations and labels specified in the effect settings are automatically added to the pods. These annotations and labels do not overwrite existing annotations and labels of the pods. If duplicate annotations or labels exist, the annotations or labels that have higher priorities are used. The existing annotations and labels of the pods have a higher priority than the annotations and labels specified in the effect settings of matched selectors. The priorities of annotations or labels in the effect settings of the selectors descend in the order in which the selectors are matched.

In the preceding example, a selector named demo is declared to implement the following feature:

If the namespace of a pod has the department=bigdata label and the pod has the type=offline-task label, the pod is automatically scheduled to the elastic container instance. At the same time, the image cache feature is enabled, and the created-by-eci=true label is added to the pod.

Example on how to configure hot update

In data of the eci-profile configuration file, you can modify vSwitchIds to update the vSwitches that can be used to create elastic container instances. The vSwtichIds parameter can be specified to implement the multi-zone feature. The following sample code shows how to configure hot update:

```
data:
  enableClusterIp: "true"
  enableHybridMode: "false"
  enablePrivateZone: "false"
  resourceGroupId: ""
  securityGroupId: sg-2ze0b9o8pjjzts4h****
  selectors: ""
  vSwitchIds: vsw-2zeet2ksvw7f14ryz****,vsw-2ze94pjtfuj9vaymf****  #Configure multiple vSwitches to impl
ement the multi-zone feature.
  vpcId: vpc-2zeghwzptn5zii0w7****
```

2.4. Install the virtual-kubelet-autoscaler add-on in an ACK cluster

virtual-kubelet-autoscaler is the Kubernetes Autoscaler add-on provided by Alibaba Cloud. If pods fail to be scheduled to matched nodes, virtual-kubelet-autoscaler takes over and re-schedules the failed pods to elastic container instances.

Prerequisites

A Container Service for Kubernetes (ACK) cluster is created, and virtual nodes are deployed.

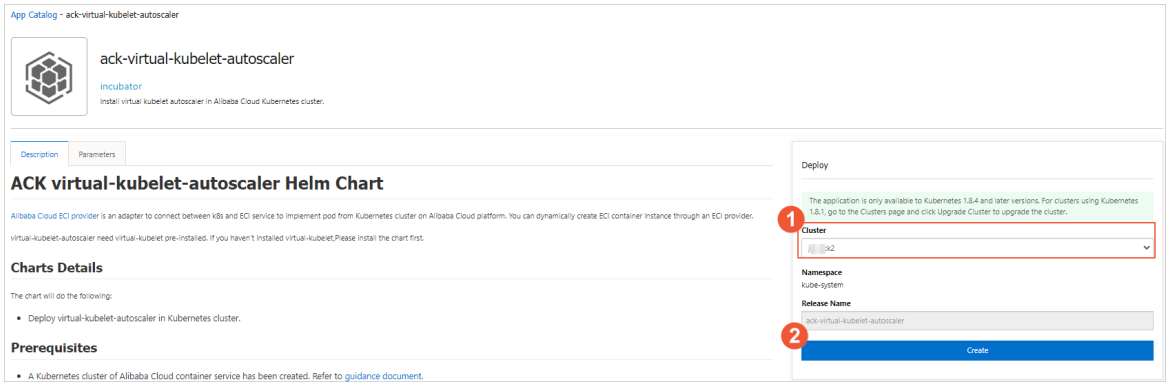
- For more information about how to create an ACK cluster, see Create a managed Kubernetes cluster.
- For more information about how to deploy virtual nodes, see Deploy ack-virtual-node in an ACK cluster.

Procedure

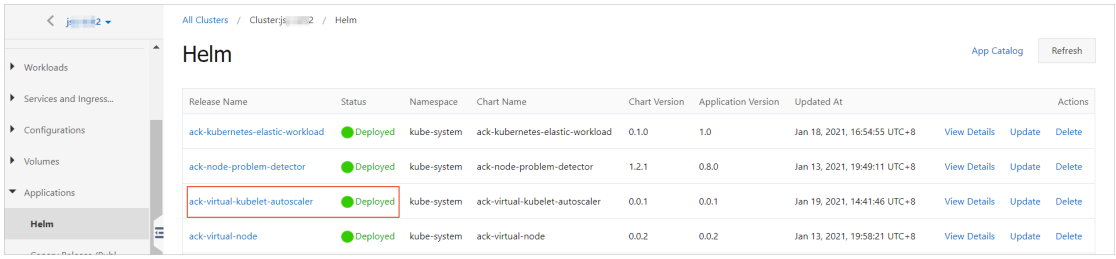
1. Log on to the ACK console.
2. In the left-side navigation pane, choose **Marketplace > App Catalog**.
3. On the **Alibaba Cloud Apps** tab, click the **ack-virtual-kubelet-autoscaler** application.

   A large volume of applications are displayed on the **Alibaba Cloud Apps** tab. You can use the search box in the upper-right corner to search for applications.

4. On the **ack-virtual-kubelet-autoscaler** page, select the created cluster from the drop-down list to deploy the application and click **Create** in the right-side **Deploy** section.



5. Check the deployment result.
   i. In the left-side navigation pane, click **Clusters**.
   ii. Click the ID of the cluster that you want to check.
   iii. In the left-side navigation pane, choose **Applications > Helm**.
   iv. Check whether the state of ack-virtual-kubelet-autoscaler is **Deployed**.



After the add-on is installed, it can schedule pods to elastic container instances in the virtual nodes when the nodes in ACK clusters are insufficient. For more information, see Schedule pods to a virtual node through the virtual-kubelet-autoscaler add-on.

2.5. Deploy and use ack-kubernetes-elastic-workload in an ACK cluster

This topic describes how to deploy and use ack-kubernetes-elastic-workload in a Container Service for Kubernetes (ACK) cluster.

Prerequisites

An ACK cluster is created, and a virtual node is deployed in the cluster.

- For more information about how to create an ACK cluster, see Create a managed Kubernetes cluster.
- For more information about how to deploy a virtual node, see Deploy ack-virtual-node in an ACK cluster.

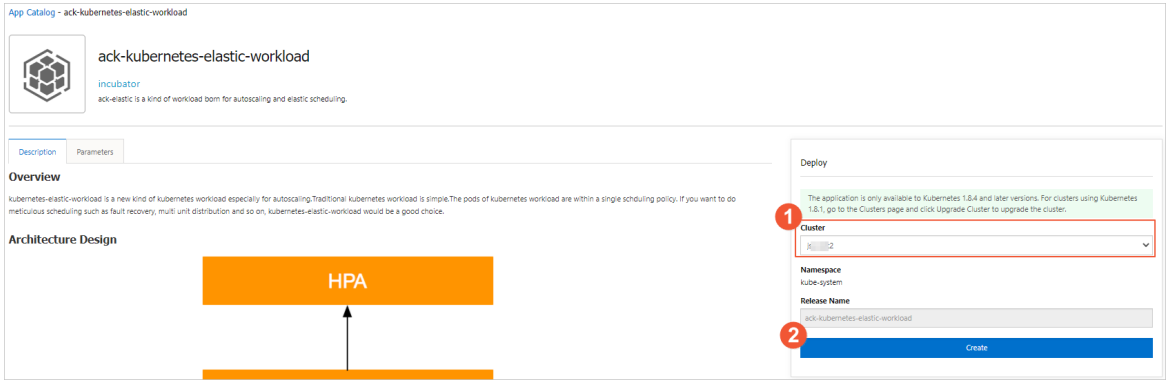
Background information

Kubernetes provides elasticity at the scheduling layer for pods and at the resource layer for nodes. Kubernetes can use Horizontal Pod Autoscaler (HPA), Cron Horizontal Pod Autoscaler(CronHPA), and Vertical Pod Autoscaling (VPA) models to scale pods and use Cluster Autoscaler and Virtual Kubelet to scale nodes. The scheduling and resource layers are decoupled by pods. This mechanism ensures that responsibilities are segregated between these layers. However, after the layers are decoupled, simple scheduling policies must be separately configured for each layer and can be used in combination. This makes it impossible to implement finer-grained scheduling policies. In Kubernetes, pod is the minimum lifecycle management unit. The pods managed by traditional Kubernetes load controllers such as Deployments and StatefulSets share scheduling policies. Therefore, to control how to allocate a workload across different resources in a fine-grained manner, you can use ack-kubernetes-elastic-workload.

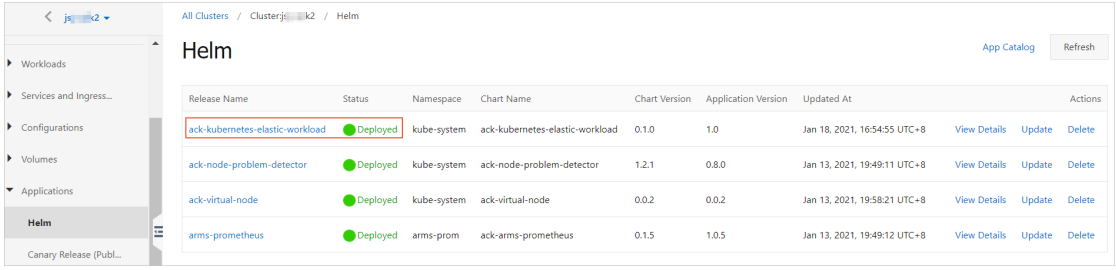
Deploy ack-kubernetes-elastic-workload

1. Log on to the ACK console.
2. In the left-side navigation pane, choose **Marketplace > App Catalog**.
3. On the **Alibaba Cloud Apps** tab, find and click **ack-kubernetes-elastic-workload**.

   A large number of applications are displayed on the **Alibaba Cloud Apps** page. You can enter a keyword in the search box in the upper-right corner to search for ack-kubernetes-elastic-workload.
4. In the **Deploy** section of the **App Catalog - ack-kubernetes-elastic-workload** page, select a cluster to deploy ack-kubernetes-elastic-workload and click **Create**.



5. Check whether ack-kubernetes-elastic-workload is deployed.
   i. In the left-side navigation pane, click **Clusters**.
   ii. Click the ID of the cluster that you selected in the previous step.
   iii. In the left-side navigation pane, choose **Applications > Helm**.
   iv. Check whether ack-kubernetes-elastic-workload is in the **Deployed** state.



Use an elastic workload

For example, capacity is planned for an application and that the application is expected to have up to four replicas to run on Elastic Compute Service (ECS) instances. Two replicas are retained during off-peak hours. If the number of replicas is greater than four, the extra replicas are scheduled to the virtual node. This prevents other applications on the ECS instances from being affected.

Kubernetes schedules workloads and manages the lifecycle of workloads. To implement the preceding scenario, you must perform the following operations:

- Control how the scheduling policy changes when the number of replicas reaches a specific value.
- Prioritize specific pods for lifecycle management.

This section describes how to use ack-kubernetes-elastic-workload to perform the preceding operations for an elastic workload.

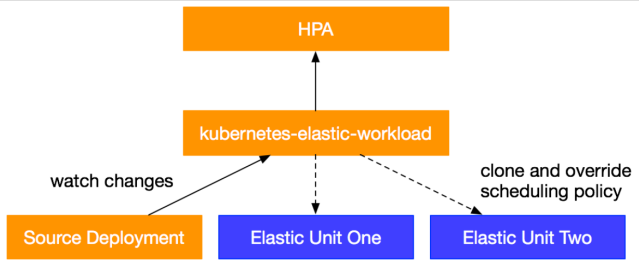
1. Create an application by using a Deployment.

```
apiVersion: apps/v1
kind: Deployment
metadata:
 name: nginx-deployment-basic
 labels:
  app: nginx
spec:
 replicas: 2
 selector:
  matchLabels:
   app: nginx
 template:
  metadata:
   labels:
    app: nginx
  spec:
   containers:
   - name: nginx
    image: nginx:1.7.9
    ports:
    - containerPort: 80
    resources:
     requests:   # If you want to use HPA for the application, you must specify requests.
      cpu: "500m"
      memory: "1024Mi"
```

2. Define an elastic workload.

```
apiVersion: autoscaling.alibabacloud.com/v1beta1
kind: ElasticWorkload
metadata:
 name: elasticworkload-sample
spec:
 sourceTarget:
  name: nginx-deployment-basic
  kind: Deployment
  apiVersion: apps/v1
  min: min: 2      #The minimum number of replicas.
  max: max: 4      #The maximum number of replicas.
 replicas: 6
 elasticUnit:
 - name: virtual-kubelet
  labels:
   alibabacloud.com/eci: "true"
```

ack-kubernetes-elastic-workload is used in a similar manner as an HPA. ack-kubernetes-elastic-workload is deployed as an external add-on and does not affect your existing business.



You can use ack-kubernetes-elastic-workload to create elastic workloads. An elastic workload listens to a source workload, and can clone and generate replicas for elastic units based on the predefined scheduling policies. If the number of replicas in an elastic workload reaches the threshold, the numbers of replicas scheduled on the source workload and on the elastic unit are dynamically adjusted.

Typically, an elastic workload consists of the following two parts:

- sourceTarget: defines the type of the source workload and the range for the number of replicas.
- elasticUnit: an array that defines the scheduling policy for the elastic unit. If you need to define scheduling policies for multiple elastic units, specify related parameters in the order shown in the template.

In this example:

- As defined in sourceTarget, the number of replicas for the source workload ranges from two to four. In this case, if the elastic workload has two to four replicas, replicas are scheduled to the source workload. If the number of replicas is greater than four, the extra replicas are scheduled to the elastic unit (Virtual Node virtual-kubelet).
- In elasticUnit, the elastic unit is defined as the virtual-kubelet virtual node, which corresponds to the `labels: alibabacloud.com/eci=true` scheduling policy.

3. Check the deployment result.
   - View the status of the elastic workload.

```
kubectl describe ew elasticworkload-sample
```

A command output similar to the following one is returned. The value of Desired Replicas in the Status section indicates the number of replicas scheduled to the elastic unit:

```
Name:         elasticworkload-sample
Namespace:    default
Labels:       <none>
Annotations:  <none>
API Version:  autoscaling.alibabacloud.com/v1beta1
Kind:         ElasticWorkload
Metadata:
  Creation Timestamp:  2021-05-21T01:53:58Z
  Generation:          4
  Managed Fields:
    API Version:  autoscaling.alibabacloud.com/v1beta1
    Fields Type:  FieldsV1
    fieldsV1:
      f:spec:
        .:
        f:elasticUnit:
        f:replicas:
        f:sourceTarget:
          .:
          f:apiVersion:
          f:kind:
          f:max:
          f:min:
          f:name:
    Manager:      Apache-HttpClient
    Operation:    Update
    Time:         2021-05-21T01:53:58Z
    API Version:  autoscaling.alibabacloud.com/v1beta1
    Fields Type:  FieldsV1
    fieldsV1:
      f:status:
        .:
        f:elasticUnitsStatus:
        f:replicas:
        f:selector:
        f:sourceTarget:
          .:
          f:apiVersion:
          f:desiredReplicas:
          f:kind:
          f:name:
          f:updateTimestamp:
    Manager:      manager
    Operation:    Update
    Time:         2021-05-21T01:56:45Z
  Resource Version:  8727
  Self Link:         /apis/autoscaling.alibabacloud.com/v1beta1/namespaces/default/elasticworkloads/elasticworkload-sample
  UID:               c4a508aa-2702-4d17-ac25-e6a207c0761a
```

```
Spec:
 Elastic Unit:
  Labels:
   alibabacloud.com/eci: true
  Name:              virtual-kubelet
 Replicas:           6
 Source Target:
  API Version: apps/v1
  Kind:       Deployment
  Max:        4
  Min:        2
  Name:       nginx-deployment-basic
Status:
 Elastic Units Status:
  Desired Replicas: 2
  Name:         nginx-deployment-basic-unit-virtual-kubelet
  Update Timestamp: 2021-05-21T01:56:45Z
 Replicas:        6
 Selector:        app=nginx
 Source Target:
  API Version:     apps/v1
  Desired Replicas: 4
  Kind:         Deployment
  Name:         nginx-deployment-basic
  Update Timestamp: 2021-05-21T01:56:45Z
Events:
 Type   Reason          Age           From        Message
 ----   ------          ----          ----        -------
 Normal SourceUpdate    12m           ElasticWorkload Source Target scale from 2 to 4
 Normal UnitCreation    12m           ElasticWorkload ElasticWorkloadUnit nginx-deployment-basi
c-unit-virtual-kubelet created
 Normal ElasticWorkloadUpdate 9m27s (x9 over 12m) ElasticWorkload ElasticWorkload update
 Normal UnitUpdate      9m27s (x8 over 12m) ElasticWorkload ElasticWorkloadUnit virtual-kubelet
has been updated
```

- View the status of pods.

```
kubectl get pod -o wide
```

A command output similar to the following one is returned. The output indicates that the elastic workload has cloned Deployments and pods, and that pod replicas in these Deployments are dynamically scheduled based on the predefined scheduling policy.

```
NAME                                                READY   STATUS    RESTARTS   AGE   IP             NODE                NOMINA
TED NODE   READINESS GATES
nginx-deployment-basic-5bf87f5f59-22jnw             1/1     Running   0          16m   10.34.0.131    cn-beiji
ng.172.16.0.1      <none>      <none>
nginx-deployment-basic-5bf87f5f59-gfp24             1/1     Running   0          13m   10.34.0.133    cn-beiji
ng.172.16.0.1      <none>      <none>
nginx-deployment-basic-5bf87f5f59-pw2zx             1/1     Running   0          13m   10.34.0.134    cn-beiji
ng.172.16.0.1      <none>      <none>
nginx-deployment-basic-5bf87f5f59-qvh7m             1/1     Running   0          16m   10.34.0.132    cn-beij
ing.172.16.0.1     <none>      <none>
nginx-deployment-basic-unit-virtual-kubelet-65fb6f4cd7-48ssb   1/1     Running   0          13m   172.16.22.
157   virtual-kubelet-cn-beijing-e   <none>      <none>
nginx-deployment-basic-unit-virtual-kubelet-65fb6f4cd7-gjqhm   1/1     Running   0          13m   172.16.22
.158   virtual-kubelet-cn-beijing-e   <none>      <none>
```

In addition, HPAs can be used for elastic workloads. Elastic workloads can dynamically adjust the number of replicas scheduled to each elastic unit based on the status of HPAs. For example, if the number of replicas in a Deployment changes from six to four for an elastic workload, the elastic workload preferentially reduces the replicas on elastic units. Sample code:

```
apiVersion: autoscaling/v2beta2
kind: HorizontalPodAutoscaler
metadata:
 name: elastic-workload-demo
 namespace: default
spec:
 scaleTargetRef:
  apiVersion: autoscaling.alibabacloud.com/v1beta1
  kind: ElasticWorkload
  name: elasticworkload-sample
 minReplicas: 2
 maxReplicas: 10
 metrics:
 - type: Resource
  resource:
   name: cpu
   target:
    type: Utilization
    averageUtilization: 50
```

In conclusion, elastic workloads can generate Deployments by cloning and overriding scheduling policies. This allows you to manage scheduling policies. Elastic workloads can also adjust the number of replicas scheduled to source workloads and adjust the number of replicas scheduled to elastic units within the specified ranges. This allows you to prioritize specific pods for lifecycle management.

2.6. ECI Pod Annotation

When you use a virtual node to schedule pods of a Kubernetes cluster to Elastic Container Instance, you can add annotations to the pods. This way, you can make full use of the Elastic Container Instance features. Make sure that the annotations that you want to add comply with the Kubernetes syntax. This topic describes the annotations supported by Elastic Container Instance and provides examples on how to configure the annotations.

The following table describes the annotations supported by Elastic Container Instance.

> **Note**
>
> The annotations described in the following table are suitable only for the pods that are scheduled to Elastic Container Instance by using virtual nodes. These pods are run on elastic container instances. The annotations cannot be added to the pods that are scheduled to Elastic Compute Service (ECS).

Annotation	Example	Description	References
k8s.aliyun.com/eci-security-group	sg-bp1dktddjsg5nktv****	The ID of the security group.	Configure a security group
k8s.aliyun.com/eci-vswitch	vsw-bp1xpiowfm5vo8o3c****	The IDs of vSwitches. You can specify multiple vSwitches for multiple zones.	Specify multiple zones to create an elastic container instance
k8s.aliyun.com/eci-schedule-strategy	VSwitchOrdered	The multi-zone scheduling policy. Default value: NoSpot. Valid values: • VSwitchOrdered: Resources in the specified zones are scheduled in the order in which the vSwitches are specified. • VSwitchRandom: Resources in the specified zones are scheduled in a random manner.	
k8s.aliyun.com/eci-ram-role-name	AliyunECIContainerGroupRole	The Resource Access Management (RAM) role that grants Elastic Container Instance permissions to access other Alibaba Cloud services.	None. The following section describes the details.
k8s.aliyun.com/eci-use-specs	2-4Gi,4-8Gi,ecs.c6.xlarge	The elastic container instance types. You can specify multiple elastic container instance types. An elastic container instance type can be a combination of vCPUs and memory or an ECS instance type.	Specify multiple instance types to create an elastic container instance

Annotation	Example	Description	References
k8s.aliyun.com/eci-spot-strategy	SpotAsPriceGo	The bidding policy for the preemptible instance. Default value: NoSpot. Valid values: • SpotAsPriceGo: The system places bids based on the spot price. • SpotWithPriceLimit: You must specify the maximum price that you want to pay for the preemptible instance per hour.	Create a preemptible instance
k8s.aliyun.com/eci-spot-price-limit	0.5	The maximum price of the preemptible instance per hour. This parameter is valid only when k8s.aliyun.com/eci-spot-strategy is set to SpotWithPriceLimit.	
k8s.aliyun.com/eci-cpu-option-core	2	The number of physical CPU cores.	Customize CPU options
k8s.aliyun.com/eci-cpu-option-ht	1	The number of threads per core.	
k8s.aliyun.com/eci-reschedule-enable	"true"	Specifies whether to enable rescheduling for elastic container instances.	None. See the sections that follow.
k8s.aliyun.com/pod-fail-on-create-err	"true"	Specifies whether to put the elastic container instances that cannot be created into the Failed state.	None. The following section describes the details.
k8s.aliyun.com/eci-image-snapshot-id	imc-2zebxkiifuyzzlhl****	The ID of the image cache. **Note** You can specify image caches or enable automatic matching for image caches. We recommend that you enable automatic matching for image caches.	

Annotation	Example	Description	References
k8s.aliyun.com/eci-image-cache	"true"	Specifies whether to enable automatic matching for image caches. **Note** You can specify image caches or enable automatic matching for image caches. We recommend that you enable automatic matching for image caches.	Use an image cache CRD to accelerate pod creation
k8s.aliyun.com/acr-instance-id	cri-j36zhodptmyq****	The ID of the Container Registry Enterprise Edition instance.	Configure settings without the need to use a password to pull images from Container Registry Enterprise Edition instances
k8s.aliyun.com/eci-eip-instanceid	eip-bp1q5n8cq4p7f6dzu****	The ID of the elastic IP address (EIP).	Enable Internet access
k8s.aliyun.com/eci-with-eip	"true"	Specifies whether to automatically create and associate an EIP.	
k8s.aliyun.com/eip-bandwidth	5	The bandwidth value for the EIP.	
k8s.aliyun.com/eip-common-bandwidth-package-id	cbwp-2zeukbj916scmj51m****	The ID of the EIP bandwidth plan.	
k8s.aliyun.com/eip-isp	BGP	The line type for the EIP. This parameter is suitable only for pay-as-you-go EIPs. Valid values: • BGP: BGP (Multi-ISP) line • BGP_PRO: BGP (Multi-ISP) Pro line	

Annotation	Example	Description	References
k8s.aliyun.com/eip-internet-charge-type	PayByBandwidth	The metering method of the EIP. Valid values: • PayByBandwidth • PayByTraffic	
k8s.aliyun.com/eci-enable-ipv6	"true"	Specifies whether to allocate IPv6 addresses. **Note** Each pod can be allocated a single IPv6 address. k8s.aliyun.com/eci-enable-ipv6 and k8s.aliyun.can be used in the same manner. Specify one of these parameters to allocate an IPv6 address.	Assign an IPv6 address to an elastic container instance
k8s.aliyun.com/eci-ipv6-count	1	The number of IPv6 addresses. Set the value to 1. **Note** Each pod can be allocated a single IPv6 address. k8s.aliyun.com/eci-enable-ipv6 and k8s.aliyun.com/eci-ipv6-count can be used in the same manner. Specify one of these parameters to allocate an IPv6 address.	
kubernetes.io/ingress-bandwidth	40M	The inbound bandwidth.	Limit the bandwidth of an elastic container instance
kubernetes.io/egress-bandwidth	20M	The outbound bandwidth.	

Annotation	Example	Description	References
k8s.aliyun.com/eci-extra-ephemeral-storage	50Gi	The size of the temporary storage space.	Create a custom temporary storage space
k8s.aliyun.com/eci-core-pattern	/pod/data/dump/core	The directory in which core dump files are stored.	View core dump files
k8s.aliyun.com/eci-ntp-server	100.100.*.*	The IP address of the Network Time Protocol (NTP) server.	Configure an NTP server for pods

Configure security groups

When Virtual Kubelet starts, it uses environment variables to configure a default security group. By default, all pods that are created on each virtual node use the security group that is configured by Virtual Kubelet. You can add annotations to specify a security group for a pod based on your business requirements.

Configuration example:

```
apiVersion: apps/v1
kind: Deployment
metadata:
 name: demo
 labels:
  app: nginx
spec:
 replicas: 1
 selector:
  matchLabels:
   app: nginx
 template:
  metadata:
    annotations:
      k8s.aliyun.com/eci-security-group: "sg-bp1dktddjsg5nktv****" # Configure the security group.
    labels:
      app: nginx
  spec:
   containers:
   - name: nginx
    image: nginx:latest
   nodeName: virtual-kubelet
```

Create pods in multiple zones

You can specify multiple zones where you can create pods. Before you specify multiple zones, specify multiple vSwitches. The system then selects a zone based on resource availability to create the pod. For more information, see Specify multiple zones to create an elastic container instance.

Configuration example:

```
apiVersion: v1
kind: Pod
metadata:
 annotations:
  k8s.aliyun.com/eci-vswitch: "vsw-bp1xpiowfm5vo8o3c****,vsw-bp1rkyjgr1xwoho6k****" # Specify multiple vSwitch IDs.
  k8s.aliyun.com/eci-schedule-strategy: "VSwitchOrdered" # Configure the multi-zone scheduling policy.
 name: nginx-test
spec:
 containers:
 - name: nginx
  image: nginx:latest
```

Configure a RAM role

You can add an annotation to configure a RAM role for a pod. The role grants the pod permissions to access Alibaba Cloud services.

> **Notice**
>
> Make sure that the configured RAM role can be used by ECS.

Configuration example:

```
apiVersion: apps/v1
kind: Deployment
metadata:
 name: set-ram-role
 labels:
  app: vk
spec:
 replicas: 1
 selector:
  matchLabels:
   app: nginx
 template:
  metadata:
    annotations:
      k8s.aliyun.com/eci-ram-role-name : "AliyunECIContainerGroupRole"   #Specify a RAM role.
    labels:
      app: nginx
  spec:
   containers:
   - name: nginx
    image: nginx:latest
   nodeName: virtual-kubelet
```

Specify one or more instance types to create a pod

You can add an annotation to specify instance types that can be used to create a pod. If resources of a specified instance type are insufficient, the system scans other specified instance types to find an instance type that provides sufficient resources to create the pod. For more information, see Specify multiple instance types to create an elastic container instance.

> **Note**
>
> An elastic container instance type can be a combination of vCPUs and memory or an ECS instance type. You can specify special ECS instance types such as GPU-accelerated ECS instances types, ECS instance types that provide high clock speeds, and ECS instance types that use local disks based on your business requirements. For more information, see Specify an ECS instance type to create an elastic container instance.

Configuration example:

```
apiVersion: apps/v1
kind: Deployment
metadata:
 name: vk-cos-use
 labels:
  app: cos
spec:
 replicas: 1
 selector:
  matchLabels:
   app: cos
 template:
  metadata:
    annotations:
      "k8s.aliyun.com/eci-use-specs": "2-4Gi,4-8Gi,ecs.c6.xlarge" # You can specify multiple specifications. The specifications can be vCPUs and memory, or specific ECS instance types.
    labels:
      app: cos
  spec:
   containers:
   - name: u1
    image: "registry-vpc.cn-beijing.aliyuncs.com/lxx/cos-4g"
   nodeName: virtual-kubelet
```

Create a preemptible elastic container instance

You can run stateless applications and jobs on preemptible instances to reduce costs. You can add annotations to create a preemptible instance. For more information, see Create a preemptible instance.

Configuration example:

```
apiVersion: apps/v1 # for versions before 1.8.0 use apps/v1beta1
kind: Deployment
metadata:
 name: nginx-deployment-basic
 labels:
  app: nginx
spec:
 replicas: 2
 selector:
  matchLabels:
   app: nginx
 template:
  metadata:
   labels:
    app: nginx
   annotations:
    k8s.aliyun.com/eci-use-specs : "ecs.c5.large"  # Specify an ECS instance type.
    k8s.aliyun.com/eci-spot-strategy: "SpotWithPriceLimit"  # Use the custom strategy to specify the maximum price.
    k8s.aliyun.com/eci-spot-price-limit: "0.250"   # Specify the maximum price per hour for the preemptible instance.
  spec:
  #  nodeSelector:
  #    env: test-team
   containers:
   - name: nginx
    image: nginx:1.7.9 # replace it with your exactly <image_name:tags>
    ports:
    - containerPort: 80
```

Customize CPU options

The CPU options of an elastic container instance include the number of physical CPU cores and the number of threads per core. An elastic container instance may support custom CPU options. This is determined by the system based on how the instance is created. For more information, see Customize CPU options.

Configuration example:

```
apiVersion: v1
kind: Pod
metadata:
 annotations:
  k8s.aliyun.com/eci-use-specs : "ecs.c6.2xlarge"   # Specify an ECS instance type that supports custom CPU options.
  k8s.aliyun.com/eci-cpu-option-core: 2           # Set the number of physical CPU cores to 2.
  k8s.aliyun.com/eci-cpu-option-ht: 1             # Set the number of threads per core to 1. This value indicates that the HT is disabled.
 name: nginx-test
spec:
 containers:
 - name: nginx
  image: nginx:latest
 restartpolicy: Always
```

Configure rescheduling for elastic container instances

Pods may fail to be scheduled to virtual nodes. You can add an annotation to enable rescheduling for pods. This ensures that the system continues to retry to schedule pods instead of returning failures even if the asynchronous scheduling fails.

Configuration example:

```
apiVersion: apps/v1
kind: Deployment
metadata:
 name: set-eci
 labels:
  app: vk
spec:
 replicas: 1
 selector:
  matchLabels:
   app: nginx
 template:
  metadata:
    annotations:
      k8s.aliyun.com/eci-reschedule-enable: "true"   # Enable rescheduling for elastic container instances.
    labels:
      app: nginx
  spec:
   containers:
   - name: nginx
    image: nginx:latest
   nodeName: virtual-kubelet
```

Change the state of the pods that cannot be created to Failed.

By default, if an error occurs when a pod is being created, the system retries for up to the specified maximum number of retries. If the pod cannot be created after the maximum number of retries, the pod enters the Pending state. For some jobs, you may want the pod to enter the Failed state. In these cases, you can add an annotation to change the state of a pod that cannot be created to Failed:

Configuration example:

```
apiVersion: apps/v1
kind: Deployment
metadata:
 name: set-pod-fail-on-create-err
 labels:
  app: vk
spec:
 replicas: 1
 selector:
  matchLabels:
   app: nginx
 template:
  metadata:
    annotations:
      k8s.aliyun.com/pod-fail-on-create-err: "true"  # Set the status to Failed if the pod fails to be created.
    labels:
      app: nginx
  spec:
   containers:
   - name: nginx
     image: nginx:latest
   nodeName: virtual-kubelet
```

Use an image cache as a custom resource definition (CRD) to accelerate pod creation

You can use image caches to accelerate pod creation. You can add annotations to specify image caches or enable automatic matching for image caches. For more information, see Use an image cache CRD to accelerate pod creation.

Configuration example:

- Specify an image cache

```
apiVersion: v1
kind: Pod
metadata:
 annotations:
  k8s.aliyun.com/eci-image-snapshot-id: imc-2ze5tm5gehgtiiga****  # Specify an image cache.
 name: nginx-imagecache-id
spec:
 containers:
 - image: nginx:1.7.9
  imagePullPolicy: IfNotPresent
  name: nginx
  resources:
   limits:
    cpu: 300m
    memory: 200Mi
   requests:
    cpu: 200m
    memory: 100Mi
 nodeName: virtual-kubelet
```

- Automatic matching

```
apiVersion: v1
kind: Pod
metadata:
 annotations:
  k8s.aliyun.com/eci-image-cache: "true"   # Specify whether to enable automatic matching for image cac
hes.
 name: nginx-auto-match
spec:
 containers:
 - image: nginx:1.7.9
  imagePullPolicy: IfNotPresent
  name: nginx
  resources:
   limits:
    cpu: 300m
    memory: 200Mi
   requests:
    cpu: 200m
    memory: 100Mi
 nodeName: virtual-kubelet
```

Specify a Container Registry Enterprise Edition instance

Container Registry allows you to pull images without the need to enter a password. You can add annotations to specify a Container Registry Enterprise Edition instance and pull images from an image repository in the instance. For more information, see Configure password-free settings to pull images from Container Registry Enterprise Edition instances.

Configuration example:

```
apiVersion: v1
kind: Pod
metadata:
 annotations:
  k8s.aliyun.com/acr-instance-id: cri-j36zhodptmyq****    # Specify the ID of a Container Registry Enterprise Edition instance.
 name: cri-test
spec:
 containers:
 - image: test****-registry.cn-beijing.cr.aliyuncs.com/eci_test/nginx:1.0  # Pull images over the Internet.
  imagePullPolicy: Always
  name: nginx
 restartPolicy: Never
```

Associate an EIP with a pod

If you want to connect a pod over the Internet, you can associate an EIP with the pod. For more information, see Bind an EIP to a pod.

Configuration example:

- Automatically create and associate an EIP

```
apiVersion: v1
kind: Pod
metadata:
 name: nginx
 annotations:
  k8s.aliyun.com/eci-with-eip: "true"   # Enable automatic EIP creation.
  k8s.aliyun.com/eip-bandwidth: "10"    # Specify the bandwidth. Default value: 5. Unit: Mbit/s.
spec:
 containers:
 - image: registry-vpc.cn-hangzhou.aliyuncs.com/jovi/nginx:alpine
  imagePullPolicy: Always
  name: nginx
  ports:
  - containerPort: 80
   name: http
   protocol: TCP
 restartPolicy: OnFailure
```

- Associate an existing EIP

```
apiVersion: v1
kind: Pod
metadata:
 name: nginx
 annotations:
  k8s.aliyun.com/eci-eip-instanceid: "eip-bp1q5n8cq4p7f6dzu****"  # Specify the EIP.
spec:
 containers:
 - image: registry-vpc.cn-hangzhou.aliyuncs.com/jovi/nginx:alpine
  imagePullPolicy: Always
  name: nginx
  ports:
  - containerPort: 80
   name: http
   protocol: TCP
 restartPolicy: OnFailure
```

Configure IPv6

Compared with IPv4 addresses, the number of IPv6 addresses is sufficient to connect more devices to the Internet. You can add annotations to allocate IPv6 addresses to pods. For more information, see Assign an IPv6 address to an elastic container instance.

Configuration example:

```
apiVersion: v1
kind: Pod
metadata:
 name: nginx
 annotations:
  k8s.aliyun.com/eci-enable-ipv6: "true"   # Enable automatic IPv6 assignation.
spec:
 containers:
 - name: nginx
  image: nginx
 nodeName: virtual-kubelet
```

Configure inbound and outbound bandwidths

You can configure inbound and outbound bandwidths for elastic container instances. You can add annotations to limit the inbound and outbound bandwidths of pods. For more information, see Limit the bandwidth of an elastic container instance.

Configuration example:

```
apiVersion: v1
kind: Pod
metadata:
 name: eci-qos
 annotations:
  kubernetes.io/ingress-bandwidth: 40M   # Specify the inbound bandwidth.
  kubernetes.io/egress-bandwidth: 10M   # Specify the outbound bandwidth.
spec:
 containers:
 - name: nginx
  image: nginx:latest
  command: ["bash","-c","sleep 100000"]
```

Specify the size of the temporary storage space

Each elastic container instance provides 20 GiB storage. If this storage space is insufficient, you can create a temporary storage space. You can add an annotation to configure the size of the temporary storage space. For more information, see Create a custom temporary storage space.

Configuration example:

```
apiVersion: v1
kind: Pod
metadata:
 name: test
 annotations:
  k8s.aliyun.com/eci-extra-ephemeral-storage: "50Gi"# Specify the size of temporary storage space.
spec:
 containers:
 - name: nginx
  image: nginx:latest
  imagePullPolicy: IfNotPresent
 restartPolicy: Always
 nodeName: virtual-kubelet
```

Configure a directory to store core dump files

If a program unexpectedly terminates or exits, a core dump file is generated. By default, this file is named core.pid and stored in the directory that is being used. You can add an annotation to configure a directory to store core dump files. For more information, see View core dump files.

Configuration example:

```
apiVersion: v1
kind: Pod
metadata:
 name: test
 annotations:
  k8s.aliyun.com/eci-core-pattern: "pod/data/dump/core"  # Specify the directory in which the core dump files are stored.
spec:
 containers:
 - image: nginx:latest
  name: test-container
  volumeMounts:
  - mountPath: /pod/data/dump/
    name: default-volume
 volumes:
 - name: nfs
  nfs:
   server: 143b24****-gfn3.cn-beijing.nas.aliyuncs.com
   path: /dump/
   readOnly: false
```

Configure the NTP service

You can add the k8s.aliyun.com/eci-ntp-server annotation to configure the NTP service for pods. For more information, see Configure an NTP server for pods.

Configuration example:

```
apiVersion: apps/v1
kind: Deployment
metadata:
 name: set-ngnix-ntp
 labels:
  app: vk
spec:
 replicas: 1
 selector:
  matchLabels:
   app: nginx
 template:
  metadata:
    annotations:
      k8s.aliyun.com/eci-ntp-server: 100.100.5.*,100.100.5.*  # Specify the IP addresses of your NTP servers.
    labels:
      app: nginx
  spec:
   containers:
   - name: nginx
    image: nginx:latest
   nodeName: virtual-kubelet
```

2.7. Custom events of Elastic Container Instance

This topic describes custom events of Elastic Container Instance and how to handle these events.

Events that are related to container groups

Name	Type	Message	Description
StockExhaust	Warning	The stock of the specified zones will be used up. %s	The resources of the elastic container instances in the current zone are about to be exhausted. We recommend that you select another zone.
NoStock	Warning	Create ECI failed because the specified instance is out of stock. %s	The resource inventory of the elastic container instances in the current zone is insufficient. You can select multiple zones and multiple instance types when you create elastic container instances. This increases the creation success rate. For more information, see Specify multiple zones to create an elastic container instance and Specify multiple instance types to create an elastic container instance.
StockClose	Warning	Create ECI failed because current zone closed or limited without living instance. %s	The elastic container instance fails to be created because elastic container instances are not supported in the current zone. Select another zone.
FailedScheduling	Warning	Unknown error occurred.	The elastic container instance fails to be created because an unknown error occurs. Submit a ticket to resolve the issue.
FailedScheduling	Warning	Schedule eci failed:%s	The elastic container instance fails to be scheduled. Try again. If the issue persists, submit a ticket.
UnknownError	Warning	The ECI service is under heavy load while creating container group resources, please wait and try again later	Elastic Container Instance is overloaded. Try again later.

Name	Type	Message	Description
SpotToBeReleased	Warning	Spot ECI will be released in %s minutes	Your preemptible elastic container instance is about to expire. Adjust your business settings accordingly to prevent business interruptions.
DiskCapacityQuotaFull	Warning	Your disk capacity quota is exceeded	The disk capacity reaches the quota limit. Submit a ticket to apply for a quota increase.
AutoInstanceTypeMatch	Normal	The most-matched instanceType for current eci instance is %s	Only the instance types that meet the requirements are selected to create elastic container instances.
UnknownError	Warning	An unknown error occurred for %s	An unknown error occurs. Submit a ticket to resolve the issue.
CpuOptionsNotSupported	Warning	The cpu options is not supported for your instanceType[%s] in current region	Your elastic container instance in the current zone does not support CPU options. For more information, see Customize CPU options.
CpuOptionsNotValid	Warning	The cpu options is not valid for your instanceType[%s]	Your elastic container instance does not support CPU options. For more information, see Customize CPU options.
SystemFailureReboot	Warning	The Specified ContainerGroup is rebooting	The elastic container instance is restarting.
Throttling	Warning	The request was denied due to system flow control, please wait and try again later	The request is rejected due to traffic throttling. Try again later.
MultiZoneRecommendations	Normal	%s	We recommend that you use the multi-zone feature. For more information, see Specify multiple zones to create an elastic container instance.

Events that are related to image caches

Name	Type	Message	Description
ImageCacheNotFound	Warning	The specified image cache %s does not exist	The specified image cache does not exist. We recommend that you query image caches and select an existing image cache.
ImageCacheMissed	Warning	Missed image cache	No image cache is matched. You must create an image cache first. For more information, see Manage image caches.
ImageCacheAutoCreated	Normal	Image cache %s is auto created	No image cache is matched. The system is creating an image cache.
ImageCacheAutoEliminated	Warning	Image cache %s is auto eliminated	The number of image caches reaches the quota limit when the system attempts to create an image cache. The system automatically deletes image caches based on the least recently used (LRU) policy. You can also submit a ticket to apply for a quota increase.
ImageCacheCapacityUtilization	Normal	Total capacity utilization of current CacheDisk is %sGB / %sGB	This event name indicates the utilization of space after the current image cache is created.
ImageCacheAutoCreateFailed	Warning	Image cache auto create failed for %s	The system fails to create the image cache. Try again. You can also manually create an image cache.
mageCacheFlashCreatedFailed	Warning	Flash image cache is failed, standard image cache will be created	The instance image cache fails to be created based on a local snapshot. The system is creating a standard image cache.
ImageCacheFlashFinished	Normal	Flash image cache is created successfully, standard image cache will be created	The instant image cache is created based on a local snapshot. The system is creating a standard image cache.
ImageCacheFlashUsedFailed	Warning	Flash image cache can only be used in zones with ESSD, but current zone is %s	If you want to create an instant image cache based on a local snapshot, you must use an enhanced SSD (ESSD). You can create an instant image cache only in zones in which ESSDs are supported.

Name	Type	Message	Description
ImageCacheFlashReplacedFailed	Warning	Standard image cache is failed, flash image cache %s will not be deleted. But flash image can only be used in zones with ESSD	The instant image cache is created based on a local snapshot. However, the standard image cache fails to be created. The system retains the local snapshot that is used to create the instant image cache.
ImageCacheFlashReplacedSuccess	Warning	Flash image cache [%s] has been replaced successfully	The local snapshot that is used to create the instant image cache is deleted.
ImageCacheCreate	Normal	Snapshot [%s] of image cache has been created, waiting for the completion of it	The system is creating the image cache. Wait until the image cache is created before you use the image cache.
ImageCacheFinished	Normal	Image cache [%s] has been made successfully	The image cache is created. You can use the image cache.
ImageCacheUpdateFinished	Normal	Image cache [%s] has been updated successfully	The image cache is updated. You can use the image cache.
ImageCacheUpdateFailed	Warning	Image cache [%s] update failed, but the old resource is still available	The image cache fails to be updated. You can use the original image cache. Try to update the image cache again.
ImageCacheUnknownError	Warning	An unknown error occurred while handling the image cache for %s	An unknown error occurs in the image cache. Submit a ticket to resolve the issue.
SuccessfulHitImageCache	Normal	Successfully hit image cache %s	An image cache is matched.
ImageCacheNotComplete	Normal	The production for specified image cache %s is not yet complete	The image cache is not created. You can use the image cache if the image cache is in the Ready state.
ImageCacheEliminated	Warning	Current imageCache is eliminated	The specified image cache is deleted. Specify another image cache.

Events that are related to other resources

Resource module	Name	Type	Message	Description
RamRole	Unauthorized	Warning	No permission granted to eci service account for %s	Your Alibaba Cloud account does not have the permissions to operate Elastic Container Instance resources. Make sure that your Alibaba Cloud account is assigned a service-linked role of Elastic Container Instance. If you use a Resource Access Management (RAM) user, make sure that the RAM user has the permissions to use the RAM role. For more information, see Grant permissions to a RAM user.
vSwitch	ResourceInsufficient	Warning	The maximum number of IP address in the VSwitch %s is exceeded	The available IP addresses in the vSwitch are insufficient. Select another vSwitch.
SecurityGroup	QuotaFull	Warning	The maximum number of instances in the security group %s is exceeded	The number of instances in the security group has reached the quota limit. Select another security group. You can also submit a ticket to apply for a quota increase.
EIP	CreateEipFailed	Warning	%s	The elastic IP address (EIP) fails to be created. Try again.
SLS	AliyunSlsQuotaExceed	Warning	%s	The number of Log Service resources exceeds the quota limit. Submit a ticket to apply for a quota increase.
SLS	AliyunSlsError	Warning	%s	A Log Service-related error occurs.
SLS	AliyunSlsProjectInvalid	Warning	%s	The specified Log Service project is invalid.