ALIBABA CLOUD

Alibaba Cloud

Elastic Container Instance

Network

Document Version: 20211223

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.Enable Internet access

To enable Internet access for your elastic container instance, you must configure a NAT gateway or an elastic IP address (EIP) for the instance and pay network usage fees. This topic describes how to associate an EIP with an elastic container instance and how to attach a NAT gateway to the virtual private cloud (VPC) where an elastic container instance resides.

Background information

The following table describes two methods used to enable Internet access for an elastic container instance.

Method	Description	Fee
Associate an EIP with the elastic container instance	EIPs are public IP addresses that can be individually purchased and managed. You can enable Internet access for an elastic container instance by associating an EIP with the instance.	EIPs support the subscription and pay-as-you-go billing methods and the pay-by-bandwidth and pay-by-data-transfer metering methods. When you associate an EIP with an elastic container instance, you are not charged a configuration fee but may be charged an association fee. For more information, see Billing overview.
Attach a NAT gateway to the VPC where the elastic container instance resides	NAT gateways are Internet gateways that can be individually purchased. After you associate an EIP with a NAT gateway, the NAT gateway can provide Internet services for all elastic container instances within the associated VPC.	NAT gateways support the pay-as-you-go billing method. A NAT gateway can provide Internet services only after it is associated with an EIP. You must pay for NAT gateways and their associated EIPs. For more information, see Billing overview.

Use appropriate methods to enable Internet access for elastic container instances based on your business needs.

- Scenario 1: Enable Internet access to NGINX deployed on an elastic container instance.

  If you want to deploy the NGINX service on an elastic container instance, you must associate an EIP with the instance when you create the instance. When NGINX starts, the elastic container instance exposes port 80 to the associated EIP. You can then use the EIP and the port number to access NGINX.

- Scenario 2: Allow multiple elastic container instances to pull images from Docker Hub over the Internet.

  By default, Elastic Container Instance does not provide external links for pulling public images over the Internet. If one or more elastic container instances in a VPC need to pull images from Docker Hub, you must attach a NAT gateway to the VPC to provide Internet access for the instances. Otherwise, the images cannot be pulled.

> **Note**
>
> When you configure Internet access for elastic container instances, make sure that rules are added to the security groups of the instances to allow traffic on specified ports and to or from specified IP addresses. For more information, see Add security group rules.

Method 1: Associate an EIP with an elastic container instance

You can associate an EIP with an elastic container instance when you create the instance. Use one of the following methods to associate an EIP with an elastic container instance:

> **Note**
>
> Each EIP can be associated with a single elastic container instance at a time and provide Internet services only for its associated elastic container instance. If multiple elastic container instances need to access the Internet, you must associate an EIP with each of these instances or attach NAT gateways to the VPCs where the instances reside.

Use Kubernetes

You can add annotations to metadata of the pod to associate an existing EIP or create and associate an EIP. Add the annotations described in the following table.

Annotation	Description
k8s.aliyun.com/eci-eip-instanceid	The existing EIP.
k8s.aliyun.com/eci-with-eip	Specifies whether to create and associate an EIP.
k8s.aliyun.com/eip-bandwidth	The maximum bandwidth value for the EIP. Unit: Mbit/s. Default value: 5.
k8s.aliyun.com/eip-common-bandwidth-package-id	The EIP bandwidth plan.
k8s.aliyun.com/eip-isp	The line type of the EIP. Valid values: • BPG: BGP (Multi-ISP) • BGP_PRO: BGP (Multi-ISP) Pro
k8s.aliyun.com/eip-internet-charge-type	The metering method of the EIP. Valid values: • PayByBandwidth: pay-by-bandwidth • PayByTraffic: pay-by-data-transfer

- Example 1: Associate an existing EIP

```
apiVersion: v1
kind: Pod
metadata:
  name: nginx
  annotations:
    k8s.aliyun.com/eci-eip-instanceid: "eip-bp1q5n8cq4p7f6dzu****"    #Associate an exist
ing 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
```

- Example 2: Create and associate an EIP and specify a bandwidth value for the EIP

```
apiVersion: v1
kind: Pod
metadata:
  name: nginx
  annotations:
    k8s.aliyun.com/eci-with-eip: "true"   #Create and associate an EIP.
    k8s.aliyun.com/eip-bandwidth: "10"   #Specify a bandwidth value for 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
```

- Example 3: Create and associate an EIP and then associate an EIP bandwidth plan

```
apiVersion: v1
kind: Pod
metadata:
  name: nginx
  annotations:
    k8s.aliyun.com/eci-with-eip: "true"   #Create and associate an EIP.
    k8s.aliyun.com/eip-common-bandwith-package-id: "cbwp-2zeukbj916scmj51m****"  #Associa
te an EIP bandwidth plan.
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
```

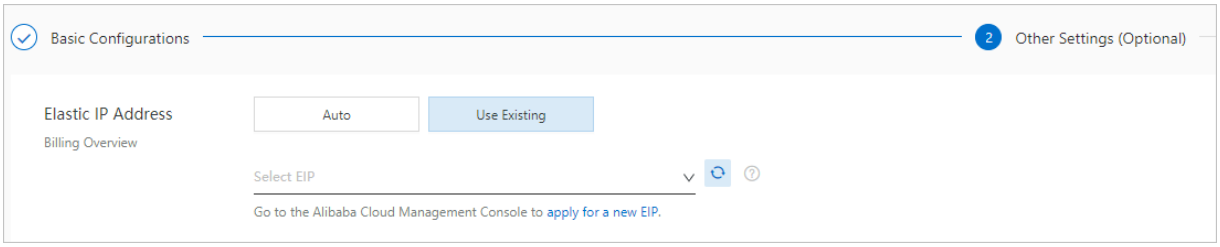
Call an API operation

When you call the CreateContainerGroup operation to create an elastic container instance, you can use the EipInstanceId parameter to associate an existing EIP or use the AutoCreateEip and EipBandwidth parameters to create and associate an EIP. The following table describes the parameters. For more information, see CreateContainerGroup.

Parameter	Type	Example	Description
EipInstanceId	String	eip-uf66jeqopgqa9hdn****	The EIP to be associated with the elastic container instance.
AutoCreateEip	Boolean	true	Specifies whether to create an EIP and associate it with the elastic container instance.
EipBandwidth	Integer	5	The maximum bandwidth value for the EIP. Unit: Mbit/s. Default value: 5. You can specify this parameter when you set AutoCreateEip to true.

Use the Elastic Container Instance console

When you create an elastic container instance in the Elastic Container Instance console, you can associate an EIP with the instance in the **Other Settings** step. In the Other Settings step, you can associate an existing EIP or create and associate an EIP, as shown in the following figure.



Method 2: Attach a NAT gateway to the VPC where an elastic container instance resides

In the VPC console, you can attach a NAT gateway to a VPC and associate an EIP with the NAT gateway to implement the following features:

- Source NAT (SNAT): allows elastic container instances within the VPC to access the Internet when these instances are not assigned public IP addresses.
- Destination NAT (DNAT): maps the EIP to the IP addresses of elastic container instances within the VPC so that the instances can provide Internet-facing services.

Perform the following steps:

1. Log on to the VPC console.
2. In the upper-left corner of the top navigation bar, select a region.
3. On the **NAT Gateway** page, create a NAT gateway.
   i. Click **Create NAT Gateway**.
   ii. Configure the parameters for the NAT gateway.

   Select the region, zone, VPC, and vSwitch of the elastic container instance. For more information, see Purchase a NAT gateway.
   iii. Confirm the configurations and fees and click **Buy Now**.
4. On the **Elastic IP Addresses** page, create an EIP.
   i. Click **Create EIP**.
   ii. Configure the parameters for the EIP.

   Select the region where the elastic container instance is located. For more information, see Apply for new EIPs
   iii. Confirm the configurations and fees and click **Buy Now**.
5. Associate the EIP with the NAT gateway.
   i. On the **NAT Gateway** page, find the created NAT gateway and click **Associate Now** in the Elastic IP Address column.
   ii. In the Associate EIP dialog box, select the created EIP and click **OK**.
6. To allow your elastic container instance to access the Internet, you must create an SNAT entry for the NAT gateway.
   i. On the **NAT Gateway** page, find the NAT gateway and click **Configure SNAT** in the Actions column.
   ii. Click **Create SNAT Entry**.

iii. Configure the parameters for the SNAT entry.

Take note of the parameters described in the following table. For more information, see Configure SNAT to access the Internet.

Parameter	Description
SNAT Entry	Click **Specify VSwitch**.
Select vSwitch	Select the vSwitch to which the elastic container instance is connected. You can specify multiple vSwitches. After the SNAT entry is created, all the elastic container instances that are connected to the specified vSwitches can use SNAT to access the Internet.
Select Public IP Address	Select **Use One IP Address** and then select the EIP that is associated with the NAT gateway. This EIP is used to communicate with the Internet.

iv. Click **OK**.

> **Note**
>
> If your elastic container instance has an associated EIP, the instance uses this EIP instead of the SNAT feature of the NAT gateway to access the Internet.

7. To allow your elastic container instance to provide Internet-facing services, you must create a DNAT entry for the NAT gateway.

i. On the **NAT Gateway** page, find the NAT gateway and click **Configure DNAT** in the Actions column.

ii. Click **Create DNAT Entry**.

iii. Configure the parameters for the DNAT entry.

Take note of the parameters described in the following table. For more information, see Configure DNAT to provide Internet-facing services.

Parameter	Description
Select Public IP Address	Select the EIP that is associated with the NAT gateway. This EIP is used to communicate with the Internet.
Select Private IP Address	Select the elastic container instance that needs to communicate with the Internet by using the DNAT entry. You can specify the elastic network interface (ENI) bound to the instance or enter the private IP address of the instance.
Port Settings	Select a DNAT mapping method: ■ Any Port: specifies IP address mapping. The NAT gateway forwards the requests destined for the associated EIP to the selected elastic container instance. ■ Specific Port: specifies port mapping. The NAT gateway forwards the requests from a specific protocol and port destined for the associated EIP to the corresponding port on the selected elastic container instance.

iv. Click **OK**.

2.Add elastic container instances to an SLB instance

Server Load Balancer (SLB) is a service that forwards network traffic to backend servers to increase the throughput of your applications. You can use SLB to prevent service interruptions that are caused by single points of failure (SPOFs) and improve the availability of applications. This topic describes how to add elastic container instances as backend servers to an SLB instance. This topic also describes how to configure listeners to forward Internet traffic to the elastic container instances.

Background information

An SLB service consists of an SLB instance, listeners, and backend servers. The following section describes how to configure elastic container instances and add the elastic container instances to an SLB instance:

1. Create multiple elastic container instances

   Before you build an SLB service, you must specify a region and a network based on your business requirements. Then, create elastic container instances in the region to deploy your applications.

2. Create an SLB instance

   Before you build an SLB service, you must create an SLB instance. Each SLB instance is a service entity that receives network traffic and forwards the traffic to backend servers.

   The SLB service is categorized into Classic Load Balancer (CLB) and Application Load Balancer (ALB). You can select an SLB type based on your business requirements. For information about the differences between CLB and ALB, see SLB instance family.

3. Add elastic container instances as backend servers to the SLB instance

   Backend servers are a group of servers that receive frontend requests. After you add elastic container instances as backend servers to an SLB instance, the elastic container instances can receive client requests that are forwarded by the SLB instance. For ALB instances, you must create a server group before you can add elastic container instances to ALB. For CLB instances, you can directly add elastic container instances to the default server group.

4. Configure listeners

   A listener is a device that checks client requests from clients and forwards healthy client requests to backend servers. You must configure listeners for your SLB instance. A listener includes a protocol, ports, and scheduling algorithms.

In the following section, CLB is used as an example to describe how to add elastic container instances to an SLB instance. You can follow the same procedure to add elastic container instances to an ALB instance. For more information about how to add elastic container instances to an ALB instance, see Quick start.

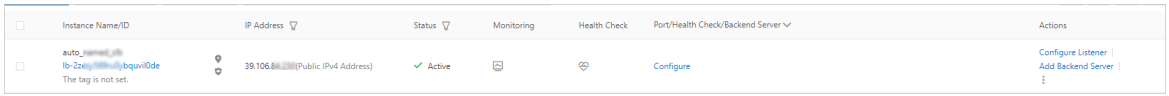
Preparations

1. Create multiple elastic container instances.

   In this example, two elastic container instances that run on the NGINX web server are created. When you create the elastic container instances, enable the logging feature. For more information, see Use an NGINX image to create an elastic container instance.

2. Create a CLB instance.

   Create a CLB instance that allows access from the Internet. For more information, see Create a CLB instance.



Procedure

If you add elastic container instances as backend servers to a CLB instance and configure listeners, the CLB instance can forward requests from clients to the elastic container instances.

> **Note**
>
> If an elastic container instance is in a final state such as the Succeeded or Failed state, you cannot add the instance as a backend server to a CLB instance.

After you bind elastic network interfaces (ENIs) to your elastic container instances, you can call a CLB API operation to add the ENIs to your CLB instance. You can also call a CLB API operation to configure listeners. Perform the following steps:

1. Call the DescribeContainerGroups operation that is provided by Elastic Container Instance to query the information about the elastic container instance.

   Obtain the following information about the elastic container instance from the response parameters:

   - The ID of the ENI instance. The EniInstanceId parameter indicates the ID of the ENI instance.
   - The internal IP address of the elastic container instance. The IntranetIp parameter indicates the internal IP address of the elastic container instance.

2. Call the AddBackendServers operation to add a backend server.

   The following table describes the main request parameters. For more information, see AddBackendServers.

Parameter	Type	Example	Description
LoadBalancerId	String	lb-2ze7o5h52g02kkzz******	The ID of the CLB instance.

Parameter	Type	Example	Description
BackendServers	String	[{ "ServerId": "eni-6wejdtelaz2bv526****", "Weight": "100", "Type": "eni", "ServerIp": "172.16.12.**", "Port":"80","Description":"test" },{ "ServerId": "eni-6wejdtelaz2bv321****", "Weight": "100", "Type": "eni", "ServerIp": "172.16.12.**", "Port":"80","Description":"test" }]	The backend servers that you want to add to the CLB instance. Configure the following parameters: ○ ServerId: The ID of the backend server. Enter the ID of the ENI instance that is bound to the elastic container instance. ○ Weight: The weight of the backend server. Valid values: 0 to 100. Default value: 100. If the value is set to 0, no requests are forwarded to the backend server. ○ Description: The description of the backend server. The description must be 1 to 80 characters in length. ○ Type: The type of the backend server. Set this parameter to eni. ○ ServerIp: The IP address of the backend server. Enter the internal IP address of the elastic container instance.

3. Call the following operations to create listeners:
   - CreateLoadBalancerTCPListener: creates a TCP listener.
   - CreateLoadBalancerUDPListener: creates a UDP listener.
   - CreateLoadBalancerHTTPListener: creates an HTTP listener.
   - CreateLoadBalancerHTTPSListener: creates an HTTPS listener.

   In this example, a TCP listener is created. The following table describes the main request parameters that are used to create the TCP listener. For more information, see CreateLoadBalancerTCPListener.

Parameter	Type	Example	Description

Parameter	Type	Example	Description
Bandwidth	Integer	-1	The maximum bandwidth of the listener. Valid values: ○ -1: For a pay-by-traffic Internet-facing CLB instance, set this value to -1. A value of -1 specifies that the bandwidth is unlimited. ○ 1 to 5120: For a pay-by-bandwidth Internet-facing CLB instance, you can specify the maximum bandwidth for each listener. The sum of the maximum bandwidth values of all listeners cannot exceed the maximum bandwidth of the CLB instance. Unit: Mbit/s.
BackendServers	Integer	80	The backend port that is used by the CLB instance. Valid values: 1 to 65535.
LoadBalancerId	String	lb-2ze7o5h52g02kkzz******	The ID of the CLB instance.
ListenerPort	Integer	80	The frontend port that is used by the CLB instance. Valid values: 1 to 65535.

4. Call the StartLoadBalancerListener operation to enable the listener.

Verify that the elastic container instances are added to the CLB instance

1. Run the following command multiple times to send requests to the public IP address of the CLB instance:

```
curl 39.106.**.** 80
```

2. View the logs of the elastic container instances. The logs show that the CLB instance forwards the requests to different elastic container instances.

3.Use the service discovery feature based on Alibaba Cloud DNS PrivateZone in ASK clusters

Serverless Kubernetes (ASK) clusters support service discovery for intranet Services, headless Services, and ClusterIP type Services.

Prerequisites

- Alibaba Cloud DNS PrivateZone is activated in the Alibaba Cloud DNS console.
- Create an ASK cluster.
- You are connected to the ASK cluster. For more information, see Use kubectl to connect to an ASK cluster.

Context

Alibaba Cloud DNS PrivateZone is a private domain name resolution and management service based on Virtual Private Cloud (VPC). You can map a private domain name to an IP address in one or more custom VPCs. Your private domain names are not accessible in other network environments.

Procedure

1. Create a Deployment and a Service.

   You can copy the following content into a YAML file and run the `kubectl create -f nginx-service-ack.yaml` command to create a Deployment and a Service.

```
apiVersion: v1
kind: Service
metadata:
  name: nginx-headless-service
spec:
  ports:
  - port: 80
    protocol: TCP
  selector:
    app: nginx
  clusterIP: None
---
apiVersion: v1
kind: Service
metadata:
  name: nginx-clusterip-service
spec:
  ports:
  - port: 80
    protocol: TCP
  selector:
    app: nginx
  type: ClusterIP
---
```

```
apiVersion: v1
kind: Service
metadata:
  name: nginx-intranet-service
  annotations:
    service.beta.kubernetes.io/alicloud-loadbalancer-address-type: intranet
spec:
  ports:
  - port: 80
    protocol: TCP
  selector:
    app: nginx
  type: LoadBalancer
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
  labels:
    app: nginx
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image:  nginx:alpine
        ports:
        - containerPort: 80
```

2. Run the following command to view the state of the created application:

```
kubectl get svc,pod,deployment
```

3. Log on to the Alibaba Cloud DNS console.
4. In the left-side navigation pane, click **PrivateZone** to go to the **All Zones** tab.
5. On the All Zones tab, find the zone that you want to manage and click **Configure** in the Actions column. The resolution settings page appears.

> **Note** Each record is in the `$svc.$ns` format and corresponds to an IP address. The following resolution rules apply:
>
> - A LoadBalancer type Service corresponds to only one resolution record in Alibaba Cloud DNS PrivateZone. The record corresponds to the IP address of the related Server Load Balancer (SLB) instance.
> - A ClusterIP type Service corresponds to only one resolution record in Alibaba Cloud DNS PrivateZone. The record corresponds to the IP address of the cluster.
> - A headless Service corresponds to multiple resolution records in Alibaba Cloud DNS PrivateZone. These records correspond to the IP addresses of the backend pods.

You can access a Service by using the private domain name in the VPC.

- Long domain name: You can use a long domain name of the `$svc.$ns.svc.cluster.local.$clusterId` format to access Services that are synchronized from other clusters to Alibaba Cloud DNS PrivateZone.
- Short domain name: You can use a short domain name of the `$svc` format to access Services in the same namespace. You can use a short domain name of the `$svc.$ns` format to access Services in other namespaces.

For more information, see serverless-k8s-examples.

4.Deploy applications that provide services by using Ingresses

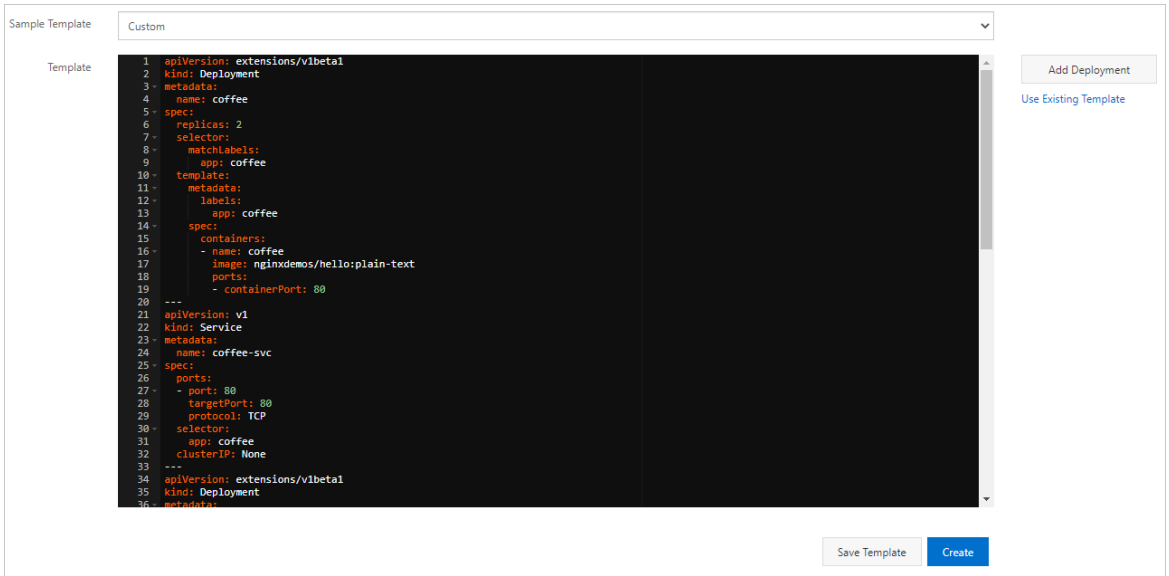
This topic describes how to deploy applications that provide services by using an Ingress on a virtual node of a Container Service for Kubernetes (ACK) cluster. This allows you to provide the applications with scalable and unlimited computing capacities without the need to create new nodes in the cluster. This also ensures the elasticity of the applications to withstand traffic fluctuations.

Prerequisites

- A virtual node is deployed in your ACK cluster. For more information, see Deploy the virtual node controller and use it to create Elastic Container Instance-based pods.
- The *virtual-node-affinity-injection: enabled* label is added to the vk namespace. For more information, see Create a pod in a namespace with specified labels.

Procedure

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



You can use the following YAML template to create applications and an Ingress that is used to enable access to the applications:

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: coffee
spec:
```

```
  replicas: 2
  selector:
    matchLabels:
      app: coffee
  template:
    metadata:
      labels:
        app: coffee
    spec:
      containers:
      - name: coffee
        image: nginxdemos/hello:plain-text
        ports:
        - containerPort: 80
---
apiVersion: v1
kind: Service
metadata:
  name: coffee-svc
spec:
  ports:
  - port: 80
    targetPort: 80
    protocol: TCP
  selector:
    app: coffee
  clusterIP: None
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: tea
spec:
  replicas: 3
  selector:
    matchLabels:
      app: tea
  template:
    metadata:
      labels:
        app: tea
    spec:
      containers:
      - name: tea
        image: nginxdemos/hello:plain-text
        ports:
        - containerPort: 80
---
apiVersion: v1
kind: Service
metadata:
  name: tea-svc
  labels:
spec:
```

```
  ports:
  - port: 80
    targetPort: 80
    protocol: TCP
  selector:
    app: tea
  clusterIP: None
---
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: cafe-ingress
spec:
  rules:
  - host: cafe.example.com
    http:
      paths:
      - path: /tea
        backend:
          serviceName: tea-svc
          servicePort: 80
      - path: /coffee
        backend:
          serviceName: coffee-svc
          servicePort: 80
```

Verify the result

- In the left-side navigation pane of the cluster details page, choose **Workloads > Deployments**. You can find the newly created coffee and tea applications.
- In the left-side navigation pane of the cluster details page, choose **Workloads > Pods**. You can verify that the pods of the newly created applications run on virtual-kubelet nodes.
- On the details page of the cluster, choose **Network > Ingresses**. You can find the newly created Ingress.
- Run the following command to query the Ingress. Then, test access to the Ingress.

  ```
  kubectl  get ing
  ```

  Expected output:

  ```
  NAME           HOSTS              ADDRESS          PORTS   AGE
  cafe-ingress   cafe.example.com   114.55.252.185   80      6m18s
  ```

  Run the following command to access `"Host:cafe.example.com" <EXTERNAL_IP>/tea` to test whether the tea application can be accessed:

  ```
  curl -H "Host:cafe.example.com" <EXTERNAL_IP>/tea
  ```

  Expected output:

```
Server address: 192.168.xx.xx:80
Server name: tea-658d56f6cc-cxxxx
Date: 25/Sep/2020:12:36:50 +0000
URI: /tea
Request ID: b01d5bab9ae07abb8bc385377193xxxx
```

Run the following command to access `"Host:cafe.example.com" <EXTERNAL_IP>/coffee` to test whether the coffee application can be accessed:

```
curl -H "Host:cafe.example.com" <EXTERNAL_IP>/coffee
```

Expected output:

```
Server address: 192.168.xx.xx:80
Server name: coffee-8c8ff9b4f-hxxxx
Date: 25/Sep/2020:12:36:47 +0000
URI: /coffee
Request ID: 722fe41a65a7fb552613c56e0a9axxxx
```

5.Assign an IPv6 address to an elastic container instance

Elastic container instances support IPv4 and IPv6 addresses. Compared with IPv4, IPv6 offers more IP addresses. This topic describes how to assign an IPv6 address to an elastic container instance.

Prerequisites

IPv6 CIDR blocks are enabled for the virtual private cloud (VPC) and the vSwitch of the elastic container instance. For more information, see Enable an IPv6 CIDR block for a VPC.

Background information

IPv4 addresses are widely used, but the limited number of IPv4 addresses restricts the development of the Internet. Compared with IPv4 addresses, IPv6 addresses are more sufficient and allow more types of devices to access the Internet. For more information, see IPv6.

The following limits apply when you assign IPv6 addresses to elastic container instances:

- Only a single IPv6 address can be assigned to each elastic container instance.
- All elastic container instances that were created by specifying vCPU and memory specifications support IPv6 addresses. However, among elastic container instances that were created by specifying Elastic Compute Service (ECS) instance types, only the instances created based on the instance types of the following instance families support IPv6 addresses:
  - General-purpose instance families: g6e, g6, g5, and sn2ne
  - Compute-optimized instance families: c6e, c6a, c6, c5, and sn1ne
  - Memory-optimized instance families: r6e, r6, r5, and se1ne
  - Instance families with high clock speeds: hfc6 and hfg6
  - Compute-optimized GPU-accelerated instance families: gn6i, gn6v, and gn5i
  - Big data instance family with enhanced network performance: d1ne
  - Instance family with local SSDs: i2
  - Burstable instance families: t6 and t5
  - Shared instance family: s6

  For more information, see Instance families.
- By default, IPv6 addresses can be used to communicate only within VPCs. If you want to use an IPv6 address to communicate over the Internet, you must enable public bandwidth for the IPv6 address. For more information, see Enable Internet connectivity for an IPv6 address.

Procedure

When you create an elastic container instance, you can add an annotation to metadata in the pod configuration file to assign an IPv6 address. The `k8s.aliyun.com/eci-enable-ipv6` annotation is added. Sample code:

```
apiVersion: v1
kind: Pod
metadata:
  name: nginx
  annotations:
    k8s.aliyun.com/eci-enable-ipv6: "true" # Assign an IPv6 address.
spec:
  containers:
  - name: nginx
    image: nginx
```

6.Change the image of a pod without changing the IP address

When you update a pod to troubleshoot issues or to improve functions of a pod, you may need to modify the image of the pod. During this process, the IP address of the pod cannot be changed to ensure business continuity. This topic describes how to use the kubectl command to modify the container image of a pod and keep the IP address of the pod unchanged.

Prerequisites

kubectl is installed and kubectl can interact with your kubernetes cluster.

Procedure

The following example describes how to use the kubectl command to change the container image from `nginx:1.7.9` to `nginx:1.9.6` and keep the IP address of the pod that runs NGINX unchanged.

1. Create a pod.

   ```
   kubectl create -f nginx.yaml
   ```

   The following code provides an example of the content of nginx.yaml. The container image used is `nginx:1.7.9` .

   ```
   apiVersion: v1
   kind: Pod
   metadata:
     name: nginx
     namespace: default
   spec:
     nodeName: virtual-kubelet
     containers:
     - image: nginx:1.7.9
       imagePullPolicy: Always
       name: nginx
   ```

2. Query the information of the pod.

   i. Query the IP address of the pod.

   ```
   kubectl get pod/nginx -n default -o wide
   ```

   The following example output shows that the IP address of the pod is 172.16.22.193.

   ```
   NAME    READY   STATUS    RESTARTS   AGE    IP              NODE
   NOMINATED NODE   READINESS GATES
   nginx   1/1     Running   0          5m5s   172.16.22.193   virtual-kubelet-cn-beij
   ing-e   <none>           <none>
   ```

ii. Query the image tag of the pod.

```
kubectl get pod/nginx -n default -o=custom-columns='IMAGE:spec.containers[*].image'
```

The following example output shows that the container image of the pod is `nginx:1.7.9`.

```
IMAGE
nginx:1.7.9
```

3. Use the following methods to change the container image:
   - kubectl patch

     Run the kubectl command to modify the tag of the container image.

     ```
     kubectl patch pod nginx -p '{"spec":{"containers":[{"name": "nginx","image": "nginx:1
     .9.6"}]}}'
     ```

   - kubectl edit

     Run the kubectl edit command to edit the pod and modify the tag of the container image.

     ```
     kubectl edit pod/nginx -o yaml
     ```

   - kubectl apply

     Open the nginx.yaml configuration file of the pod, modify the tag of the container image, and then run the kubectl apply command to redeploy the pod.

     ```
     kubectl apply -f nginx.yaml
     ```

4. View the information of the pod after you change the image.

   i. Query the IP address of the pod.

   ```
   kubectl get pod/nginx -n default -o wide
   ```

   The following example output shows that the IP address of the pod is 172.16.22.193, which is the same as the IP address of the pod before the image was changed.

   ```
   NAME    READY   STATUS    RESTARTS   AGE   IP              NODE
   NOMINATED NODE   READINESS GATES
   nginx   1/1     Running   1          19m   172.16.22.193   virtual-kubelet-cn-beiji
   ng-e   <none>           <none>
   ```

   ii. Query the image tag of the pod.

   ```
   kubectl get pod/nginx -n default -o=custom-columns='IMAGE:spec.containers[*].image'
   ```

   The following example output shows that the container image of the pod is changed to `nginx:1.9.6`.

   ```
   IMAGE
   nginx:1.9.6
   ```

7.Configure a security group

Security groups act as virtual firewalls to provide Stateful Packet Inspection (SPI) and packet filtering capabilities and define security domains in the cloud. You can add security group rules to control inbound and outbound traffic for elastic container instances within security groups.

Security group overview

Security group definition

A security group is a logically isolated group of instances within the same region that are mutually trusted and share the same security requirements. The rules of a security group control access to or from the Internet or internal network for the elastic container instances within the security group.

> Note
>
> - Each security group can manage multiple elastic container instances within the same region.
> - Each elastic container instance must belong to a single security group.

Security group types

Security groups are classified into basic security groups and advanced security groups. By default, the following rules are added when a security group is created:

- Inbound rules that allow access on ports 80 (HTTP), 443 (HTTPS), 22 (SSH), and 3389 (RDP) and an inbound rule that allows Internet Control Message Protocol (ICMP) access on all ports. These rules can be modified.
- An outbound rule that allows all access on all ports.

The following table describes the differences in features of basic and advanced security groups.

Feature	Basic security group	Advanced security group
Access control policy when the security group contains no rules	• Inbound: denies all access requests. • Outbound: allows all access requests.	• Inbound: denies all access requests. • Outbound: denies all access requests.
Maximum number of private IP addresses	2,000	65,536
Mutual access between instances within the same security group	By default, instances within the same security group can access each other over the internal network.	By default, instances within the same security group are isolated from each other over the internal network. You must manually add security group rules to allow mutual access between the instances.

Feature	Basic security group	Advanced security group
Control on access to or from other security groups	Rules can be added to control access to or from other security groups.	Rules cannot be added to control access to or from other security groups.

> **Notice**
>
> If your business requires a large number of elastic container instances and high O&M efficiency, we recommend that you use advanced security groups. Compared with basic security groups, advanced security groups can accommodate more elastic container instances and make it easier to configure security group rules.

Security group rules

Rules can be added to security groups to control inbound and outbound traffic. A security group rule is defined by attributes such as the rule direction, action, protocol type, port range, and authorization object. Take note of the following items about security group rules:

- The total number of inbound and outbound rules in each security group cannot exceed 200.
- Follow the principle of least privilege when you add security group rules. Examples:
  - Specify single ports such as port 80 in the format of 80/80, instead of a port range such as ports 1 through 80 in the format of 1/80.
  - 0.0.0.0/0 indicates all IP addresses. Do not configure it as the authorization object unless necessary.

For more information, see Overview.

Specify a security group

When you create an elastic container instance, you must specify a security group for the instance.

> **Notice**
>
> The security groups of elastic container instances cannot be changed. To use an elastic container instance within a different security group, create an identical elastic container instance in that security group.

Kubernetes mode

When you use Elastic Container Instance based on Virtual Kubelet in Kubernetes scenarios, all elastic container instances within a cluster are added to the default security group configured by Virtual Kubelet. You can move an elastic container instance to a specified security group based on your needs.

- **Cluster**

  You can run the kubectl edit command to modify the eci-profile configuration file of a cluster and change the default security group ID in the data section for the elastic container instances within the cluster.

> **Note**
>
> Virtual Kubelet of v2.0.0.90-15deb126e-aliyun or later allows modifications to eci-profile for hot updates. If your Virtual Kubelet version is earlier than v2.0.0.90-15deb126e-aliyun, we recommend that you upgrade your Virtual Kubelet.

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

Modify the securityGroupId field in the data section. Sample code:

```
data:
  enableClusterIp: "true"
  enableHybridMode: "false"
  enablePrivateZone: "false"
  resourceGroupId: ""
  securityGroupId: sg-2ze0b9o8pjjzts4h**** #Specify a security group ID.
  selectors: ""
  vSwitchIds: vsw-2zeet2ksvw7f14ryz****,vsw-2ze94pjtfuj9vaymf****
  vpcId: vpc-2zeghwzptn5zii0w7****
```

- **Elastic container instance**

  You can add annotations to metadata in the pod configuration file to specify a security group for an elastic container instance. Sample code:

```
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****"      #Specify a
security group ID.

        labels:
            app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx:latest
```

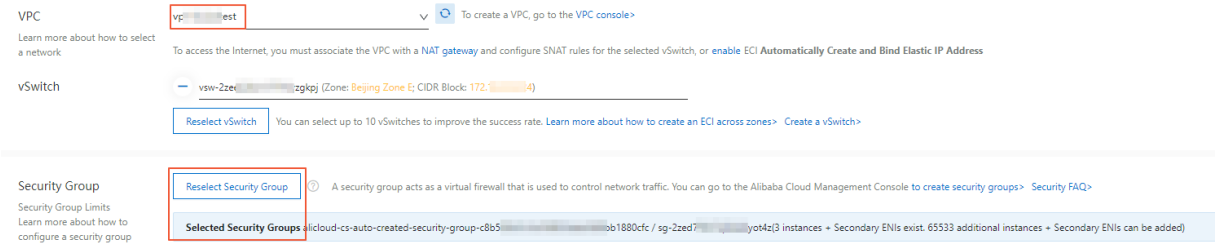
API mode

When you call the CreateContainerGroup operation to create an elastic container instance, you can use the SecurityGroupId parameter to specify a security group. The following table describes the parameter. For more information, see CreateContainerGroup.

Parameter	Type	Example	Description
SecurityGroupId	String	sg-uf66jeqopgqa9hdn****	The ID of the security group.

Console mode

When you create an elastic container instance on the instance buy page in the Elastic Container Instance console, you can specify a security group for the instance.



Add security group rules

You can add rules to a security group to control inbound and outbound traffic for the elastic container instances within the security group.

- If your elastic container instance needs to communicate with a network outside the security group to which the instance belongs, you can add a security group rule to allow the instance access to the network.
- When attacks are detected from request sources during the operation of elastic container instances, you can add security group rules to block the malicious requests.

For more information about how to add security group rules, see Add security group rules.

8.Limit the bandwidth of an elastic container instance

Elastic Container Instance allows you to configure the network bandwidth values for inbound and outbound traffic. This topic describes how to limit the inbound and outbound bandwidth of an elastic container instance.

Kubernetes mode

When you use Kubernetes to create an elastic container instance, you can add annotations to a pod to specify the maximum inbound and outbound bandwidth values. Take note of the following items:

- kubernetes.io/ingress-bandwidth: the inbound bandwidth.
- kubernetes.io/egress-bandwidth: the outbound bandwidth.

Supported units include GB, G, MB, M, KB, K, and B. If the unit is not specified, B is used by default, which indicates bytes.

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

API mode

When you use the CreateContainerGroup operation to create an elastic container instance, you can use the CreateContainerGroup and IngressBandwidth parameters to specify the maximum inbound and outbound bandwidth values. The following table describes the parameters. For more information, see CreateContainerGroup.

Parameter	Type	Required	Example	Description
IngressBandwidth	Long	No	102400	The inbound bandwidth. Unit: bytes.
EgressBandwidth	Long	No	102400	The outbound bandwidth. Unit: bytes.