ALIBABA CLOUD

Alibaba Cloud

Auto Scaling

Best practices

Document Version: 20210909

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.Build a scalable web application

This topic describes how to build a scalable web application by using Auto Scaling that can automatically respond to increases and decreases in business activities. This allows you to handle daily business and traffic spikes during major activities.

Prerequisites

- An Alibaba Cloud account is created. To create an Alibaba Cloud account, go to the account registration page.
- A custom image is created for an ECS instance. For more information, see Create a custom image from an instance.

Scenarios

An e-commerce platform launches promotions during holidays, member days, and shopping festivals to attract users. To handle the traffic spikes during promotions, the operations and maintenance (O&M) personnel estimate the compute resources required for new promotional activities based on historical data. If unexpected traffic spikes occur during peak hours, the O&M personnel must manually create ECS instances. This is time-consuming and may affect the availability of your application.

You can adopt the solutions provided in this topic if your application has the following characteristics:

- Deployed in a cluster that has at least one server.
- Has traffic spikes for a short duration. For example, the traffic spikes last no more than nine hours each day, and no more than 20 days each month.

Solutions

Auto Scaling automatically scales compute resources based on increases and decreases in business activities without the need for prediction and manual intervention. This ensures the availability of your application. Especially during big promotions such as Double 11, Auto Scaling can deliver up to thousands of ECS instances within minutes, and respond to traffic spikes automatically and timely to ensure service availability.

You can adopt the following solutions:

- Purchase subscription ECS instances to meet daily business requirements.
- Use Auto Scaling to monitor load changes and automatically create ECS instances in response to unexpected traffic spikes.

Benefits

Auto Scaling enables you to respond to traffic spikes and offers the following benefits:

- Zero backup resource cost

  Auto Scaling automatically creates and releases ECS instances based on your requirements. You do not need to maintain backup resources. You only need to reserve compute resources for daily business traffic.

- Zero maintenance cost

You can configure the scaling policy in advance. When the load increases, Auto Scaling automatically creates and adds ECS instances to the whitelist of the ApsaraDB for RDS instance and SLB backend server group. When the load decreases, Auto Scaling automatically removes ECS instances from the SLB backend server group and the whitelist of the ApsaraDB for RDS instance, and then releases the instances. The whole process is automatically triggered and completed without the need for manual intervention.

- Flexibility and intelligence

  Auto Scaling provides a variety of scaling modes. You can select a combination of multiple scaling modes based on business changes to implement the optimal match for your business. For example, if your web application that requires a large and steady volume of traffic experiences a temporary traffic spike, you can use the dynamic mode based on CloudMonitor metrics. This allows you to monitor average CPU utilization and automatically respond to traffic changes in a timely manner.

Procedure

Evaluate business modules based on your business architecture and perform the following operations to implement automatic scaling for specified business modules:

- Step 1: Use a custom image to create subscription ECS instances
- Step 2: Create and enable a scaling group
- Step 3: Add subscription ECS instances and configure the automatic scaling policy

Step 1: Use a custom image to create subscription ECS instances

Create and add the specified number of subscription ECS instances to a scaling group in response to daily traffic requirements of business modules. Perform the following operations:

1. Log on to the ECS console.
2. In the left-side navigation pane, choose **Instances & Images > Images**.
3. In the top navigation bar, select a region.
4. Find the custom image of the web application and click **Create Instance** in the **Actions** column.
5. Configure the parameters to create the instance.
   - Set **Billing Method** to **Subscription**.
   - Information in the **Region** and **Image** sections is automatically filled.

   Configure other parameters based on your needs. For more information, see Create an instance by using the wizard.

Step 2: Create and enable a scaling group

Create a scaling group for business modules that require elastic scaling. Select a custom image for the scaling configuration to ensure that automatically created ECS instances meet web application requirements. Perform the following operations:

1. Log on to the Auto Scaling console.
2. In the top navigation bar, select a region.
3. Create a scaling group:
   - Set **Source Type** to **Create from Scratch**.
   - Set **Minimum Number of Instances** to **0**.
   - Set **Network Type** to **VPC**.

- Set **Multi-zone Scaling Policy** to **Balanced Distribution Policy**.
- Set **Instance Reclaim Mode** to **Release Mode**.
- Bind the SLB and ApsaraDB for RDS instances used by your current business modules.

Configure other parameters based on your needs. For more information, see Create a scaling group.

4. Click **View Scaling Group Details**.
5. Go to the **Instance Configuration Source** page to create a scaling configuration.

   Set **Image** to the custom image of the web application.

   Configure other parameters based on your needs. For more information, see Create a scaling configuration.
6. Enable the scaling configuration and scaling group.

Step 3: Add subscription ECS instances and configure the automatic scaling policy

Add subscription ECS instances to a scaling group and create a target tracking rule to implement automatic scaling based on traffic changes in response to traffic spikes. Perform the following operations:

1. Go to the **ECS Instances** page, and add existing subscription ECS instances to the scaling group.
2. Switch the subscription ECS instances to the Protected state to ensure service availability during daily business.
3. Go to the **Basic Information** page, and modify the minimum and maximum numbers of instances in the scaling group based on business needs.
4. Go to the **Scaling Rules** page, and create a target tracking rule.
   - Set **Rule Type** to **Target Tracking Scaling Rule**.
   - Set **Metric Name** to **Average CPU Usage**.
   - Set **Target Value** to **50%**.

   Configure other parameters based on your needs. For more information, see Create a scaling rule.

Result

The state of subscription ECS instances is switched to Protected to ensure service availability during daily business. The ECS instances in the **Protected** state cannot be removed from the scaling group and their weights in SLB are not affected.

The scaling group automatically keeps the average CPU utilization of ECS instances at about 50%. When the average CPU utilization exceeds 50%, Auto Scaling automatically creates ECS instances to balance loads. When the average CPU utilization drops below 50%, Auto Scaling automatically releases ECS instances to reduce costs. The number of ECS instances remains greater than or equal to the specified minimum number of instances, and less than or equal to the maximum number of instances to meet business requirements and keep costs within expectation.

2.Use Auto Scaling to reduce costs

This topic describes how to use Auto Scaling to purchase pay-as-you-go and preemptible ECS instances to reduce costs during predictable business peaks.

Prerequisites

- An Alibaba Cloud account is created. To create an Alibaba Cloud account, go to the account registration page.
- A custom image is created for an ECS instance. For more information, see Create a custom image from an instance.

Scenarios

An online education platform experiences traffic peaks from 17:00 to 22:00 every day. However, during other times of the day, the business traffic is significantly lower. To ensure that the platform can deliver reliable services during peak hours, the number of compute resources is scaled based on the peak traffic loads. During off-peak hours, these resources are idle, which results in a large amount of wasted cost. Furthermore, when the platform experiences unexpected traffic spikes, ECS instances must be manually created to ensure service availability.

You can adopt the solutions provided in this topic if your application has the following characteristics:

- Deployed in a cluster that has at least one server.
- Has predictable traffic patterns. For example, traffic peaks occur from 17:00 to 22:00 each day and the compute resources are idle during other times of the day.

Solutions

Auto Scaling uses a combination of pay-as-you-go and preemptible instances to meet peak traffic requirements at lower costs.

You can adopt the following solutions:

- Purchase subscription ECS instances to maintain a baseline compute capability for off-peak hours.
- Specify multiple instance types and use a combination of pay-as-you-go and preemptible instances to scale compute capabilities for peak hours. Auto Scaling creates ECS instances based on unit prices of vCPUs in ascending order. Instances that use lowest-priced vCPUs are preferentially created.

Benefits

Auto Scaling enables you to reduce costs and offers the following benefits:

- Zero backup resource cost

  Auto Scaling automatically creates and releases ECS instances based on your requirements. You do not need to maintain backup resources. You only need to reserve compute resources for off-peak hours.

- Zero maintenance cost

  You can configure the scaling policy in advance. When the load increases, Auto Scaling automatically creates and adds ECS instances to the whitelist of the ApsaraDB for RDS instance and SLB backend server group. The whole process is automatically triggered and completed without the need for manual intervention.

- High cost-effectiveness

Auto Scaling supports the combination of pay-as-you-go and preemptible instances. You can purchase ECS instances at up to 90% discount. If the preemptible instances are insufficient, pay-as-you-go instances are created to ensure service availability. The cost optimization policy also supports supplemental preemptible instances. After this feature is enabled, Auto Scaling automatically creates preemptible instances at lowest price five minutes before existing preemptible instances are released.

Procedure

Evaluate business modules based on your business architecture and perform the following operations to reduce costs for required business modules:

- Step 1: Use a custom image to create subscription ECS instances
- Step 2: Create and enable a scaling group
- Step 3: Add subscription ECS instances and configure the automatic scaling policy

Step 1: Use a custom image to create subscription ECS instances

Create and add the specified number of subscription ECS instances to a scaling group in response to off-peak traffic requirements of business modules. Perform the following operations:

1. Log on to the ECS console.
2. In the left-side navigation pane, choose **Instances & Images > Images**.
3. In the top navigation bar, select a region.
4. Find the custom image of the application and click **Create Instance** in the **Actions** column.
5. Configure the parameters to create an instance.
   - Set **Billing Method** to **Subscription**.
   - Information in the **Region** and **Image** sections is automatically filled.

   Configure other parameters based on your needs. For more information, see Create an instance by using the wizard.

Step 2: Create and enable a scaling group

Create a scaling group for business modules that require lower costs. Select a custom image for the scaling configuration to ensure that automatically created ECS instances meet application requirements. Perform the following operations:

1. Log on to the Auto Scaling console.
2. In the top navigation bar, select a region.
3. Create a scaling group.
   - Set **Source Type** to **Create from Scratch**.
   - Set **Minimum Number of Instances** to **0**.
   - Set **Network Type** to **VPC**.
   - Set **Multi-zone Scaling Policy** to **Cost Optimization Policy**.
     - Set **Minimum Pay-as-you-go Instances** to **0**.
     - Set **Percentage of Pay-as-you-go Instances** to **30%**.
     - Set **Lowest Cost Instance Types** to **3**.
     - Enable the supplemental preemptible instances mode.

- Set **Reclaim Mode** to **Release Mode**.
- Bind the SLB and ApsaraDB for RDS instances used by your current business modules.

Configure other parameters based on your needs. For more information, see Create a scaling group.

4. Click **View Scaling Group Details**.
5. Go to the **Instance Configuration Source** page to create a scaling configuration.
   - Set **Billing Method** to **Preemptible Instance**.
   - Select at least three instance types.
   - Set **Image** to your custom image.

   Configure other parameters based on your needs. For more information, see Create a scaling configuration.
6. Enable the scaling configuration and scaling group.

Step 3: Add subscription ECS instances and configure the automatic scaling policy

Add subscription ECS instances to a scaling group and create a step scaling rule to implement automatic and smooth scaling based on business changes. You can significantly reduce costs by using a combination of subscription and preemptible instances. Perform the following operations:

1. Go to the **ECS Instances** page, and add existing subscription ECS instances to the scaling group.
2. Switch the subscription ECS instances to the Protected state to ensure service availability during off-peak hours.
3. Go to the **Basic Information** page, and modify the minimum and maximum numbers of instances in the scaling group based on business needs.
4. Go to the **Scaling Rules** page, and create a step scaling rule.
   - Set **Rule Type** to **Step Scaling Rule**.
   - Set **Monitoring Type** to **System Monitoring**.
   - Set **Run At** to the time when the average CPU utilization is greater than 50% for three consecutive times.
   - Set **Operation** based on the following rules:
     - Add five instances when the average CPU utilization is greater than or equal to 60% and less than 70%.
     - Add 10 instances when the average CPU utilization is greater than or equal to 70%.

   Configure other parameters based on your needs. For more information, see Create a scaling rule.

Result

The state of subscription ECS instances is switched to Protected to ensure service availability during off-peak hours. The ECS instances in the Protected state cannot be removed from the scaling group, and their weights in SLB are not affected.

During peak hours, Auto Scaling automatically creates a specific number of ECS instances based on the average CPU utilization to implement smooth scaling. Due to the cost optimization policy and supplemental preemptible instances mode, you can purchase ECS instances at lower costs.

3.Deploy a high-availability compute cluster

This topic describes how to use Auto Scaling to evenly distribute ECS instances across zones and deploy a high-availability compute cluster at lower costs by using preemptible ECS instances.

Prerequisites

- An Alibaba Cloud account is created. To create an Alibaba Cloud account, go to the account registration page.
- A custom image is created for an ECS instance. For more information, see Create a custom image from an instance.

Scenarios

An online advertising provider uses machine learning to implement targeted advertising. During peak hours, the provider requires a large number of compute resources. This results in higher costs and may face scalability problems, such as insufficient resources, insufficient time to manually create ECS instances, and service disruption. All these problems pose risks to the business.

You can adopt the solutions provided in this topic if your application is applicable to the following scenarios:

- Distributed big data computing
- Artificial intelligence training

Solutions

Auto Scaling can provision a compute cluster in a short amount of time. The balanced distribution policy allows you to automatically distribute compute nodes across multiple zones. Auto Scaling also performs health checks on ECS instances to ensure the high availability of the compute cluster.

You can adopt the following solutions:

- Use Auto Scaling to distribute compute nodes across multiple zones and specify multiple instance types.
- Purchase preemptible ECS instances to reduce costs.

Benefits

Auto Scaling enables you to deploy a high-availability compute cluster and offers the following benefits:

- Zero maintenance cost

  You can configure the scaling policy in advance. When the load increases, the scaling group automatically creates and adds ECS instances to the whitelist of the ApsaraDB for RDS instance. When the load decreases, the scaling group automatically removes ECS instances from the whitelist of the ApsaraDB for RDS instance, and then releases the instances. The whole process is automatically triggered and completed without the need for manual intervention.

- High cost-effectiveness

  Auto Scaling supports preemptible ECS instances. You can purchase preemptible instances at up to 90% discount.

- High availability

  Auto Scaling uses the balanced distribution policy to automatically distribute and deploy compute nodes across zones. This ensures service availability and reduces the risk that resources in a zone may be insufficient. Auto Scaling automatically performs health checks to ensure the availability of ECS instances in a scaling group.

Procedure

Evaluate business modules based on your business architecture and create scaling groups for the business modules that require high-availability clusters. Select a custom image for the scaling configuration to ensure that the automatically created ECS instances meet application requirements.

1. Log on to the Auto Scaling console.
2. In the top navigation bar, select a region.
3. Create a scaling group.
   - Set **Source Type** to **Create from Scratch**.
   - Set **Minimum Number of Instances** to **100**.
   - Set **Network Type** to **VPC**.
   - Select vSwitches across multiple zones.
   - Set **Multi-zone Scaling Policy** to **Balanced Distribution Policy**.
   - Bind the ApsaraDB for RDS instances used by your current business modules.

   Configure other parameters based on your needs. For more information, see Create a scaling group.
4. Click **View Scaling Group Details**.
5. Go to the **Instance Configuration Source** page to create a scaling configuration.
   - Set **Billing Method** to **Preemptible Instance**.
   - Set **Image** to your custom image.

   Configure other parameters based on your needs. For more information, see Create a scaling configuration.
6. Enable the scaling configuration and scaling group.

Result

After the scaling group is enabled, the scaling group automatically distributes 100 ECS instances evenly across the selected zones. This can reduce impacts on the application when a zone has insufficient resources. The scaling group automatically creates new preemptible instances after the previous preemptible instances are reclaimed. Additionally, the scaling group automatically removes unhealthy ECS instances and creates new ECS instances. This ensures the high availability of clusters and also reduces costs.

4.Associate ECS instances in a scaling group with ApsaraDB instances

This topic describes how to associate ECS instances in a scaling group with ApsaraDB instances. You can add the ECS instances and the ApsaraDB instances to the same security group, associate the scaling group with the ApsaraDB instances, and add the ECS instances to the whitelists of the ApsaraDB instances.

Context

ECS instances in a scaling group may be automatically released. Therefore, we recommend that you save your application data to ApsaraDB instances. The console is used in this example to demonstrate how to associate ECS instances in a scaling group with ApsaraDB instances.

Method 1: (Recommended) Add an ECS instance and an ApsaraDB instance to the same security group

When a scaling group and an ApsaraDB instance are of the VPC type and added to the same security group, ECS instances in the scaling group can directly access the ApsaraDB instance.

> **Note** This method applies to all ApsaraDB services such as ApsaraDB RDS and ApsaraDB for MongoDB.

In this example, ApsaraDB RDS for MySQL is used. You can perform the following operations:

- Create a scaling group and an ApsaraDB RDS for MySQL instance:
  i. Create a VPC-type scaling group. For more information, see Create a scaling group.
  ii. Create and enable a scaling configuration whose security group belongs to the same VPC as the scaling group. For more information, see Create a scaling configuration.
  iii. Enable the scaling group. For more information, see Enable a scaling group.
  iv. Create and use an ApsaraDB RDS for MySQL instance whose network type and security group are the same as those of the scaling group. For more information, see Create an ApsaraDB RDS for MySQL instance and Configure a security group for an ApsaraDB RDS for MySQL instance.

  > **Note** For more information, see General workflow to use RDS for MySQL.

- Modify the scaling group and the ApsaraDB RDS for MySQL instance. Perform the following operations to configure the network type and security group of the ApsaraDB RDS for MySQL instance based on the configurations of the scaling group:
  i. View the network type of the scaling group and the security group specified in the scaling configuration. For more information, see View scaling groups.

  > **Note** The network type of a scaling group cannot be changed after the scaling group is created. If the network type of a scaling group is classic network, you must create another scaling group. For more information, see Create a scaling group.

ii. Check whether the network type of the ApsaraDB RDS for MySQL instance is the same as that of the scaling group. If not, change the network type of the ApsaraDB RDS for MySQL instance. For more information, see Change the network type of an ApsaraDB RDS for MySQL instance.

iii. Check whether the security group of the ApsaraDB RDS for MySQL instance is the same as that of the scaling group. If not, change the security group of the ApsaraDB RDS for MySQL instance. For more information, see Configure a security group for an ApsaraDB RDS for MySQL instance.

Method 2: Associate a scaling group with an ApsaraDB instance

When you create or modify a scaling group, you can associate it with an ApsaraDB RDS instance. After the scaling group is associated with an ApsaraDB RDS instance, all ECS instances in the scaling group can directly access the ApsaraDB RDS instance, regardless of the network type of the scaling group and the ApsaraDB RDS instance.

> **Note** This method applies only to ApsaraDB RDS.

- Create a scaling group. For more information, see Create a scaling group.
- Modify the scaling group. For more information, see Modify a scaling group.

> **Note** If ECS instances exist in a scaling group, you can use the following method to add the instances to the whitelists of the associated ApsaraDB RDS instances when you modify the scaling group:
>
> - Console: In the **Edit Scaling Group** dialog box, select **Add or remove instances in the scaling group to or from whitelists of RDS instances when you associate or disassociate RDS instances.**
> - AttachDBInstances: Set ForceAttach to *true*.

Method 3: Use lifecycle hooks and OOS templates to add ECS instances to the whitelists of ApsaraDB instances

You can use lifecycle hooks in conjunction with Operation Orchestration Service (OOS) templates to put ECS instances in a scaling group into the wait state. Then, you can add the ECS instances to the whitelists of ApsaraDB instances associated with the scaling group. After the ECS instances are added to the whitelists of the ApsaraDB instances, the ECS instances can directly access the ApsaraDB instances, regardless of the network type of the scaling group and the ApsaraDB instances.

> **Note** This method applies only to PolarDB, ApsaraDB for MongoDB, AnalyticDB for PostgreSQL, and AnalyticDB for MySQL.

- Automatically add or remove ECS instances to or from the whitelist of a PolarDB cluster
- Automatically add or remove ECS instances to or from the whitelist of a MongoDB instance
- Automatically add or remove ECS instances to or from the whitelist of an AnalyticDB for MySQL cluster

5.Use lifecycle hooks to ensure service availability

After a scaling group is associated with a Server Load Balancer (SLB) instance, all the Elastic Compute Service (ECS) instances in the scaling group are automatically added to the backend server group of the SLB instance to process the client requests distributed from the SLB instance. In this topic, we recommend that you use lifecycle hooks. During a scale-in or scale-out event, a lifecycle hook can be used to put ECS instances that are being added to or removed from a scaling group into the wait state for the specified period of time to ensure service availability.

Prerequisites

A scaling group is associated with an SLB instance. For more information, see Use SLB in Auto Scaling.

Scenario 1: Scale-out events

During a scale-out event, created ECS instances are added to a scaling group if you do not create a lifecycle hook for the scaling group. The created ECS instances are also added to the backend server group of the SLB instance that is associated with the scaling group and start to provide services. Applications on ECS instances require some time to start before they can provide services for clients. If the applications are not started when they receive requests from the clients, the applications cannot provide services.

We recommend that you create lifecycle hooks for scaling groups. Before ECS instances in a scaling group are added to the backend server group of an SLB instance that is associated with the scaling group, a lifecycle hook is used to put the ECS instances into the wait state. After applications on the ECS instances are started and the lifecycle hook times out, the ECS instances are added to the backend server group of the SLB instance to provide services. For more information, see Create a lifecycle hook. Take note of the following configurations:

- Set **Applicable Scaling Activity Type** to **Scale-out Event**.
- We recommend that you set **Timeout Period** to a period of time during which the applications on the ECS instances can be started.

  > **Note** If you want to terminate the timeout period in advance, you can call the CompleteLifecycleAction operation. For more information, see CompleteLifecycleAction.

- Set **Execution Policy** to **Continue**.

After the lifecycle hook is created, created ECS instances stay in the **Pending** state until the lifecycle hook times out. While the ECS instances are in the wait state, applications on the ECS instances are started. After the ECS instances are put out of the wait state, they are added to the scaling group and the backend server group of the associated SLB instance. The ECS instances provide services when they are in the **In Service** state.

Scenario 2: Scale-in events

During a scale-in event, ECS instances are removed from a scaling group if you do not create a lifecycle hook for the scaling group. The ECS instances are also removed from the backend server group of the associated SLB instance and stop providing services. However, applications on the ECS instances may be processing requests from clients. This may cause access exceptions on the clients.

We recommend that you create lifecycle hooks for scaling groups. Before ECS instances in a scaling group are removed from the backend server group of an SLB instance that is associated with the scaling group, a lifecycle hook is used to put the ECS instances into the wait state. After the ECS instances process the received requests and the lifecycle hook times out, the ECS instances are removed from the backend server group of the associated SLB instance. For more information, see Create a lifecycle hook. Take note of the following configurations:

- Set **Applicable Scaling Activity Type** to **Scale-in Event**.
- We recommend that you set **Timeout Period** to the maximum processing time for all requests.

> **Note** If you want to terminate the timeout period in advance, you can call the CompleteLifecycleAction operation. For more information, see CompleteLifecycleAction.

After the lifecycle hook is created, the ECS instances to be removed stay in the **Suspending** state until the lifecycle hook times out. While the ECS instances are in the wait state, they process the already received requests and no longer receive new requests. After the ECS instances are put out of the wait state, they are removed from the scaling group and the backend server group of the associated SLB instance.

Related information

References

- CreateLifecycleHook

6.Update images and execute scripts

You can use the Rolling Update feature to update images of and execute scripts on ECS instances with ease. This enhances the efficiency of managing ECS instances in a scaling group.

Prerequisites

An Alibaba Cloud account is created. To create an Alibaba Cloud account, go to the account registration page.

Procedure

The following section describes the status of a scaling group:

- The scaling group is in China (Hangzhou).
- The CentOS 6.4 64-bit public image is used in the active scaling configuration of the scaling group.
- 100 ECS instances are created based on the active scaling configuration of the scaling group. All the instances are in the **In Service** state.
- No scaling activities are in progress in the scaling group.

This topic describes how to update the images of ECS instances in a scaling group to the Alibaba Cloud Linux 2 image, and how to install Apache on the ECS instances after the images are updated. Perform the following operations:

- Step 1: Create a custom image
- Step 2: Update images and execute scripts

Step 1: Create a custom image

1. Log on to the ECS console.
2. In the top navigation bar, select a region.
3. In the left-side navigation pane, choose **Instances & Images > Instances**.
4. Creates an ECS instance.
    i. In the upper-right corner of the Instances page, click **Create Instance**.

ii. Configure parameters to create an instance.

The following table describes the sample configurations of the instance. Configure other parameters and make sure that the configurations are the same as those of the active scaling configuration in the scaling group.

Step	Parameter	Example
Basic Configurations	**Billing Method**	Pay-As-You-Go
	Region	■ Region: China (Hangzhou) ■ Zone: Random
	Image	■ Type: Public Image ■ Version: Aliyun Linux 2.1903 LTS 64-bit
System Configurations (Optional)	**Instance Name**	Instance-ForCustomImage

iii. Click **Next: Preview**.

iv. Read and select *ECS Terms of Service*, and then click **Create Order**.

v. In the **Created** message, click **Console** to view the creation progress on the Instances page.

If the instance is in the **Running** state, it is created.

> **Note** Before you create a custom image, you can configure the Instance-ForCustomImage instance, such as deploying applications and copying data. This helps reduce maintenance operations after the image is updated.

5. Create a custom image to be used to update images of ECS instances in the scaling group.

i. Find the Instance-ForCustomImage instance and choose **More > Disk and Image > Create Custom Image** in the **Actions** column.

ii. Configure parameters for the custom image.

The following table describes the sample configurations of the image. Configure other parameters.

Parameter	Example
Custom Image Name	Image-AliyunLinux
Custom Image Description	The image used to update images of ECS instances in the scaling group

iii. Click **Create**.

6. Create a custom image to be used to roll back images of ECS instances in the scaling group.

i. Find an ECS instance that belongs to the scaling group and choose **More > Disk and Image > Create Custom Image** in the **Actions** column.

ii. Configure parameters for the custom image.

The following table describes the sample configurations of the image. Configure other parameters.

Parameter	Example
Custom Image Name	Image-CentOSBck
Custom Image Description	The image used to roll back images of ECS instances in the scaling group when errors occur to the rolling update task

iii. Click **Create**.

7. In the left-side navigation pane, choose **Instances & Images > Images** to view the creation progress of the Image-AliyunLinux and Image-CentOSBck images on the Images page.

If the progress is 100%, the image is created.

Step 2: Update images and execute scripts

1. Log on to the Auto Scaling console.
2. In the top navigation bar, select a region.
3. In the left-side navigation pane, click **Scaling Groups**.
4. Find the scaling group and use one of the following methods to open the details page of the scaling group:
   - Click the ID of the scaling group in the **Scaling Group Name/ID** column.
   - In the **Actions** column corresponding to the scaling group, click **Details**.
5. In the left-side navigation pane, click **Rolling Update**.
6. Create and execute an image update task.
   i. In the upper-left corner of the page, click **Create Execution Task**.
   ii. Configure parameters for the rolling update task.

   The following table describes the sample configurations of the task. Configure other parameters.

Parameter	Example
Task Description	Update the image from CentOS 6.4 64-bit to Aliyun Linux 2.1903 LTS 64-bit in batches
Task Type	Image Update
Image for Update	Image-AliyunLinux
Image for Rollback	Image-CentOSBck
Execution Batch	10
Suspension Policy	Automatic

iii. Click **Create Task**.

iv. Read the execution impact and click **OK** if you confirm the information.

After you confirm the information, the rolling update task is automatically executed.

After the task is complete, images of the 100 ECS instances in the scaling group are updated to Aliyun Linux 2.1903 LTS 64-bit.

7. Create and execute a script execution task.
   i. In the upper-left corner of the page, click **Create Execution Task**.
   ii. Configure parameters for the script execution task.

   The following table describes the sample configurations of the task. Configure other configurations.

Parameter	Example
Task Description	Install Apache and view the status of Apache
Task Type	Script Execution
Script for Execution	# Install Apache. yum install -y httpd # Start Apache. systemctl start httpd # Enable Apache to run at startup. systemctl enable httpd # View the status of Apache. systemctl status httpd
Script for Rollback	# View the status of Apache. systemctl status httpd
Execution Batch	10
Suspension Policy	Automatic

   iii. Click **Create Task**.
   iv. Read the execution impact and click **OK** if you confirm the information.

   Then, the rolling update task is automatically executed.

After the task is complete, Apache is installed on the 100 ECS instances in the scaling group. The status of Apache is active.

Related information

- Create an instance by using the wizard
- Create a custom image from an instance
- Rolling update

7.Use Alibaba Cloud CLI to execute rolling update tasks

Alibaba Cloud Command Line Interface (Alibaba Cloud CLI) is a management tool based on Alibaba Cloud OpenAPI. You can use Alibaba Cloud CLI to call Alibaba Cloud APIs to flexibly manage and scale Alibaba Cloud services. This topic describes how to use Alibaba Cloud CLI to perform rolling update tasks.

Prerequisites

- Alibaba Cloud CLI is installed. For more information about how to install Alibaba Cloud CLI, see the "Installation Guide" section in What is Alibaba Cloud CLI?
- A scaling group is created and ECS instances are added to it.
- The instance configuration source of the scaling group is a scaling configuration if you want to update the images of ECS instances in the scaling group.
- Operation Orchestration Service (OOS) packages are created if you want to install OOS packages on ECS instances in the scaling group. For more information, see Manage custom software on multiple ECS instances.

Context

Rolling update tasks can be used to update the configurations of ECS instances in batches. For more information, see Rolling update.

Procedure

This topic describes how to use Alibaba Cloud CLI to update images of, execute scripts on, and install OOS packages on ECS instances in a scaling group. Perform the following operations:

- Step 1: Create a RAM user and grant it permissions
- Step 2: Configure and verify Alibaba Cloud CLI
- Step 3: Use Alibaba Cloud CLI to execute a rolling update task
- Execute rollback tasks to handle exceptions in rolling update tasks

Step 1: Create a RAM user and grant it permissions

1. Log on to the RAM console.
2. Create a RAM user.

   A RAM user named clitest is created in this example. For more information, see Create a RAM user.

   i. In the left-side navigation pane, choose **Identities > Users**.

   ii. On the Users page, click **Create User**.

iii. On the **Create User** page, configure the parameters and then click **OK**.

The following table describes the parameters.

Parameter	Example
Logon Name	clitest@sample.com
Display Name	clitest
Access Mode	**Programmatic Access**. An AccessKey pair is automatically created for the RAM user. The RAM user can call API operations or use SDKs to access Alibaba Cloud resources.

iv. On the **Basic Information** page, click **Download CSV File**.

> **Note** The AccessKey secret is displayed only when you create an AccessKey pair, and is unavailable for subsequent queries. We recommend that you save the AccessKey secret for subsequent use. If an AccessKey pair is disclosed or lost, you must create a new one.

3. Grant the RAM user permissions to manage resources.
   i. In the left-side navigation pane, choose **Identities > Users**.
   ii. Find the clitest user that you created. Click **Add Permissions** in the **Actions** column.
   iii. In the Add Permissions panel, select the policies that contain the permissions required to execute the rolling update task, and then click **OK**.

   The following table describes the parameters in the Add Permissions panel.

Parameter	Description
Authorization	Keep the **Alibaba Cloud account all resources** default value selected.
Principal	Keep the clitest@sample.com default value selected.
Select Policy	Select the following system policies: ■ AliyunECSFullAccess: Provides permissions to manage Elastic Compute Service (ECS) resources such as ECS instances. ■ AliyunESSFullAccess: Provides permissions to manage Auto Scaling resources such as scaling groups. ■ AliyunOOSFullAccess: Provides permissions to manage OOS resources such as executions. ■ AliyunOSSFullAccess: Provides permissions to manage Object Storage Service (OSS) resources such as buckets.

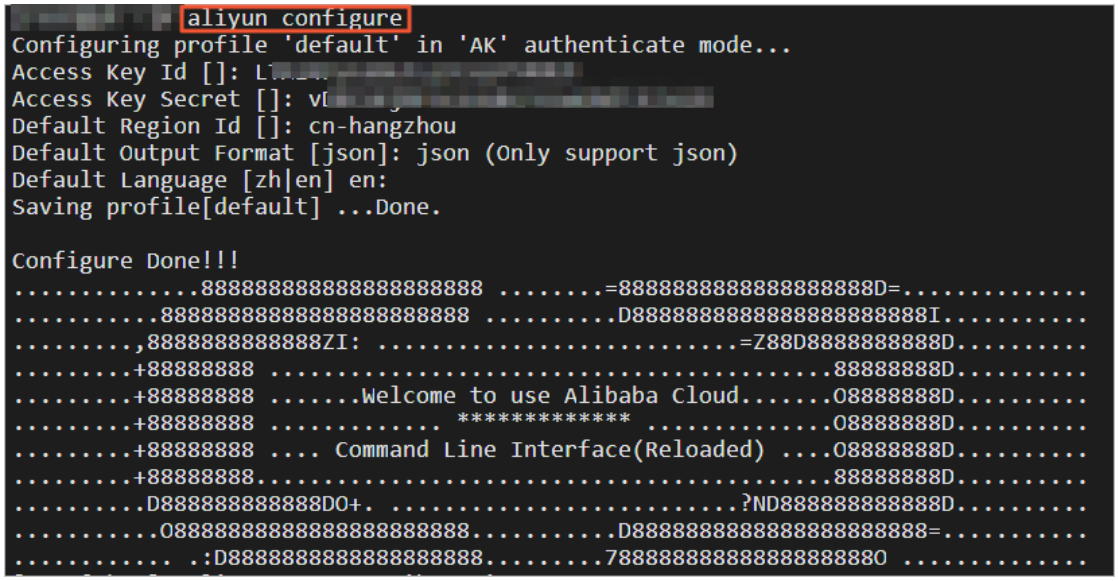
Step 2: Configure and verify Alibaba Cloud CLI

For information about the configuration parameters, see the "Configure Alibaba Cloud CLI" section in What is Alibaba Cloud CLI?

1. Open Alibaba Cloud CLI on your local computer.
2. Configure Alibaba Cloud CLI.
   i. Run the following command to open the configuration file:

   ```
   aliyun configure
   ```

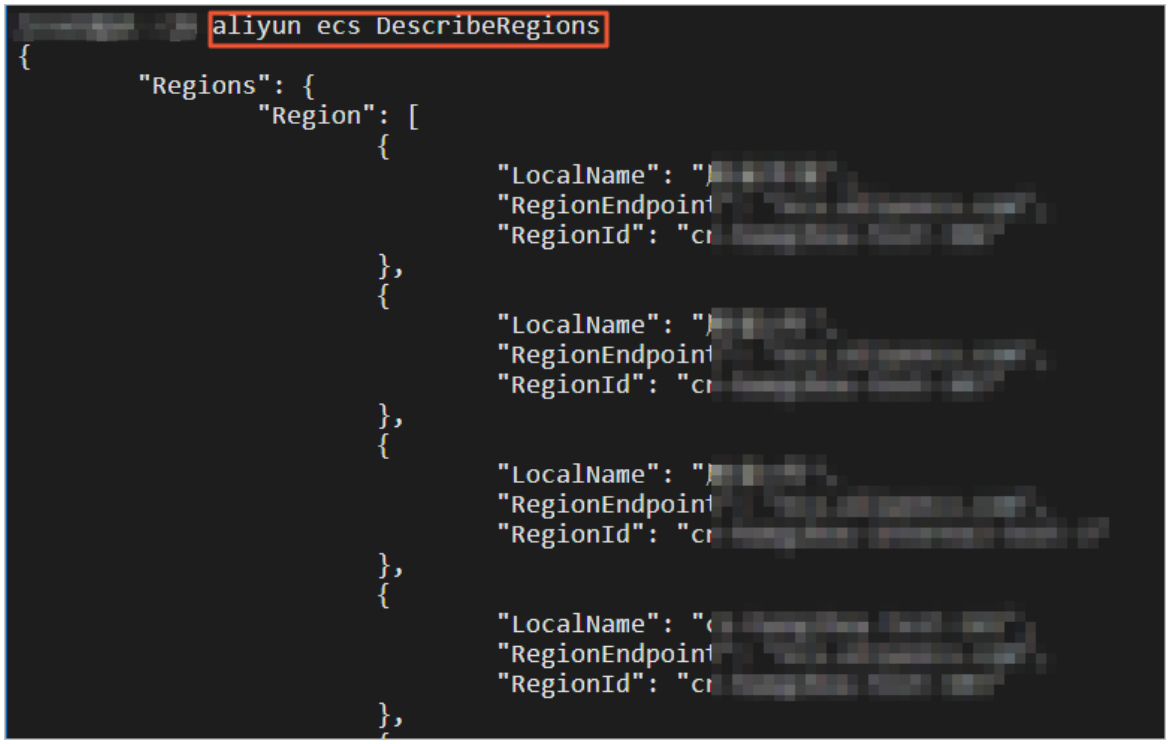
   ii. Enter the AccessKey ID and AccessKey secret.



3. Run the following command to check whether Alibaba Cloud CLI is available:

   ```
   aliyun ecs DescribeRegions
   ```

This command is used to query supported regions. If region information is returned, the command is executed. The following figure shows an example command output.



Step 3: Use Alibaba Cloud CLI to execute a rolling update task

This step provides example commands to demonstrate how to update images of, execute scripts on, and install OOS packages on ECS instances in the scaling group.

1. Run an Alibaba Cloud CLI command to execute a rolling update task.

   For information about the OOS template parameters involved in the code, see OOS template parameters.

   - In the following example, an image update command is used to update images of ECS instances in the scaling group to Alibaba Cloud Linux 2.1903 LTS 64-bit.

```
aliyun oos StartExecution --TemplateName ACS-ESS-RollingUpdateByReplaceSystemDiskInScalingGroup --Parameters "{
    \"invokeType\": \"invoke\",
    \"scalingGroupId\": \"asg-bp18p2yfxow2dloq****\",
    \"scalingConfigurationId\": \"asc-bp1bx8mzur534edp****\",
    \"imageId\": \"aliyun_2_1903_x64_20G_alibase_20200529.vhd\",
    \"sourceImageId\": \"centos_7_8_x64_20G_alibase_20200717.vhd\",
    \"OOSAssumeRole\": \"\",
    \"enterProcess\": [
     \"ScaleIn\",
     \"ScaleOut\",
     \"HealthCheck\",
     \"AlarmNotification\",
     \"ScheduledAction\"
    ],
    \"exitProcess\": [
     \"ScaleIn\",
     \"ScaleOut\",
     \"HealthCheck\",
     \"AlarmNotification\",
     \"ScheduledAction\"
    ],
    \"batchNumber\": 2,
    \"batchPauseOption\": \"Automatic\"
   }"
```

- In the following example, a script execution command is used to run the **df -h** and **ifconfig** shell commands on ECS instances in the scaling group to view the disks and network configurations of the instances.

```
aliyun oos StartExecution --TemplateName ACS-ESS-RollingUpdateByRunCommandInScalingGroup --Parameters "{
    \"invokeType\": \"invoke\",
    \"scalingGroupId\": \"asg-bp18p2yfxow2dloq****\",
    \"commandType\": \"RunShellScript\",
    \"invokeScript\": \"df -h\nifconfig\",
    \"rollbackScript\": \"df -h\nifconfig\",
    \"OOSAssumeRole\": \"\",
    \"exitProcess\": [
     \"ScaleIn\",
     \"ScaleOut\",
     \"HealthCheck\",
     \"AlarmNotification\",
     \"ScheduledAction\"
    ],
    \"enterProcess\": [
     \"ScaleIn\",
     \"ScaleOut\",
     \"HealthCheck\",
     \"AlarmNotification\",
     \"ScheduledAction\"
    ],
    \"batchNumber\": 2,
    \"batchPauseOption\": \"Automatic\"
   }"
```

- In the following example, a command for installing OOS packages is used to install the WordPress package on all ECS instances in the scaling group.

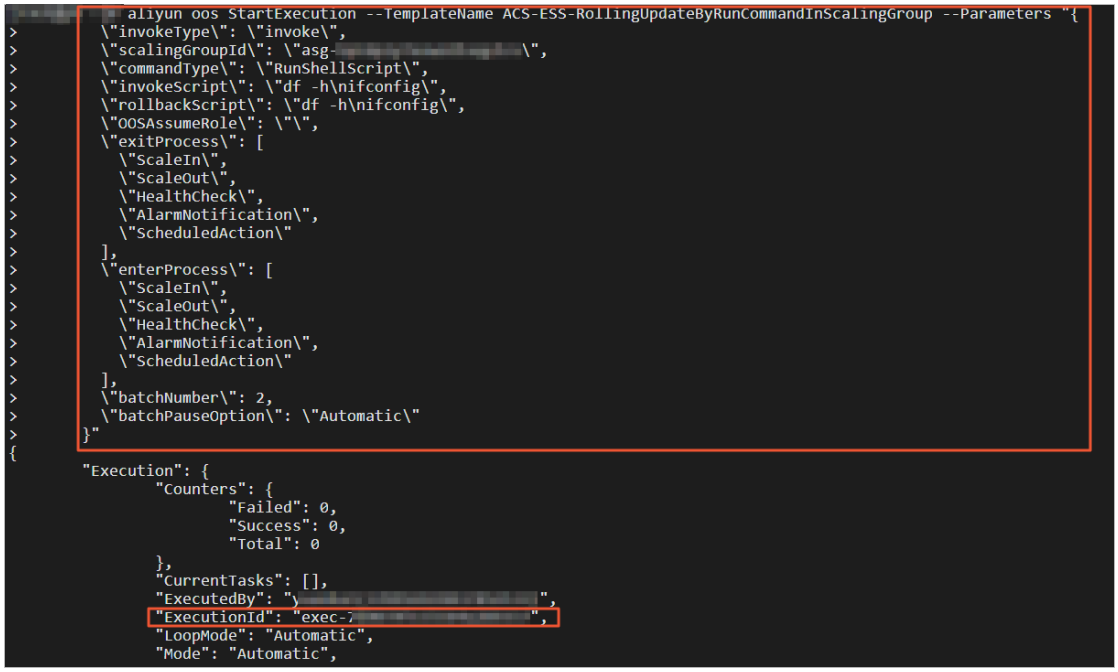
```
aliyun oos StartExecution --TemplateName ACS-ESS-RollingUpdateByConfigureOOSPackage --Param
eters "{
    \"invokeType\": \"invoke\",
    \"scalingGroupId\": \"asg-bp18p2yfxow2dloq****\",
    \"packageName\": \"wordpress\",
    \"packageVersion\": \"v4\",
    \"action\": \"install\",
    \"OOSAssumeRole\": \"\",
    \"enterProcess\": [
     \"ScaleIn\",
     \"ScaleOut\",
     \"HealthCheck\",
     \"AlarmNotification\",
     \"ScheduledAction\"
    ],
    \"exitProcess\": [
     \"ScaleIn\",
     \"ScaleOut\",
     \"HealthCheck\",
     \"AlarmNotification\",
     \"ScheduledAction\"
    ],
    \"batchNumber\": 2,
    \"batchPauseOption\": \"Automatic\"
   }"
```

2. View the execution details.

   When you run CLI commands to execute rolling update tasks, executions are automatically created in OOS. You can find the execution based on the returned execution ID to view the details about the execution, such as the execution results and output. In the following example, a script execution output is used to describe how to obtain the execution ID and view the execution details.

i. Find the execution ID of the rolling update task in the command output.

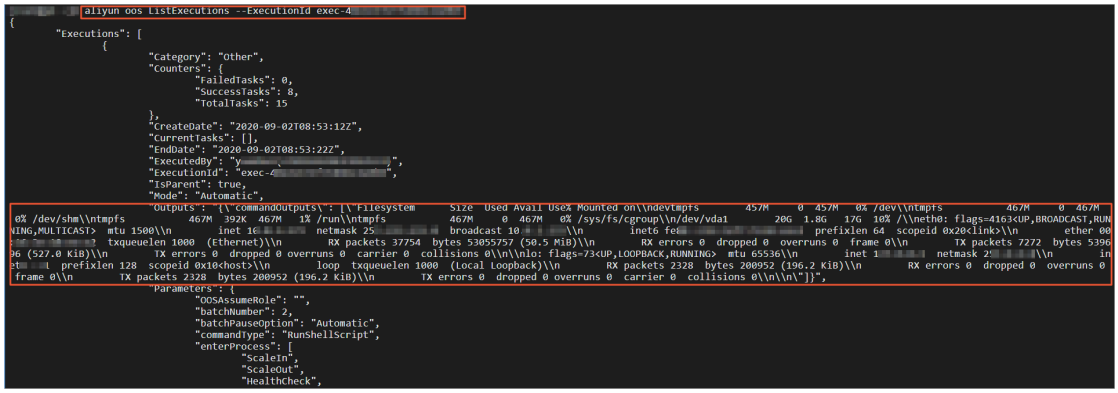
The following figure shows an example execution ID.



ii. Run the following Alibaba Cloud CLI command to view the execution details:

```
aliyun oos ListExecutions --ExecutionId exec-40e2e17ef7e04****
```

The following figure shows the example execution details.

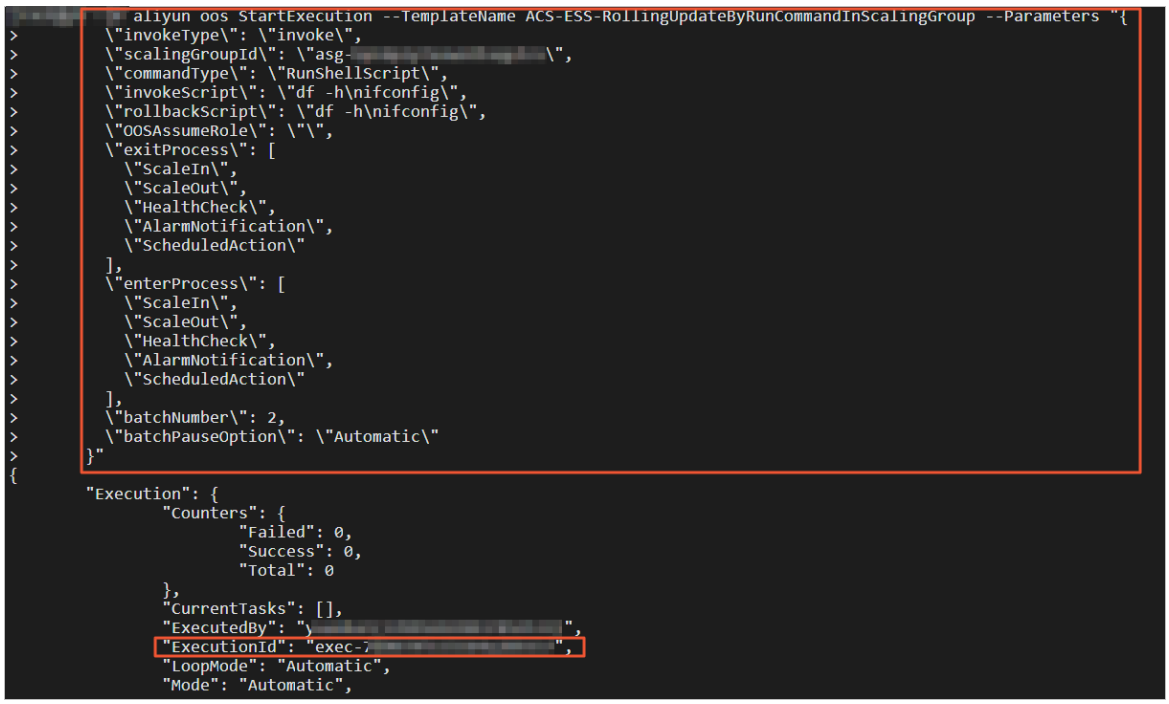


Execute rollback tasks to handle exceptions in rolling update tasks

If exceptions occur during rolling update tasks or you want to use previous configurations after rolling update tasks are executed, you can execute rollback tasks to restore the configurations of the ECS instances. Example commands are provided in this section to demonstrate how to roll back rolling update tasks that have been executed.

1. Find the execution ID of the rolling update task in the command output.

   The following figure shows an example execution ID.



2. Run an Alibaba Cloud CLI command to execute a rollback task.

   For information about the OOS template parameters involved in the code, see OOS template parameters.

   > **Note** When you execute a rollback task, OOS automatically finds ECS instances to be rolled back based on the rolling update task, and suspends or resumes scaling processes of the scaling group. Therefore, you may not need to specify some parameters.

   - In the following example, a command for rolling back image update is used to roll the images of ECS instances in the scaling group back to CentOS 7.8 64-bit.

     ```
     aliyun oos StartExecution --TemplateName ACS-ESS-RollingUpdateByReplaceSystemDiskInScalingGroup --Parameters "{
         \"invokeType\": \"rollback\",
         \"scalingGroupId\": \"asg-bp18p2yfxow2dloq****\",
         \"scalingConfigurationId\": \"asc-bp1bx8mzur534edp****\",
         \"sourceImageId\": \"centos_7_8_x64_20G_alibase_20200717.vhd\",
         \"sourceExecutionId\": \"exec-83dba59be77d430****\",
         \"OOSAssumeRole\": \"\",
         \"batchNumber\": 2,
         \"batchPauseOption\": \"Automatic\"
       }"
     ```

   - In the following example, a command for rolling back script execution is used to execute the rollback script in the ECS instances. The **df -h** and **ifconfig** shell commands are still used here. You can change the script based on your configurations.

```
aliyun oos StartExecution --TemplateName ACS-ESS-RollingUpdateByRunCommandInScalingGroup --Parameters "{
    \"invokeType\": \"rollback\",
    \"commandType\": \"RunShellScript\",
    \"rollbackScript\": \"df -h\nifconfig\",
    \"scalingGroupId\": \"asg-bp18p2yfxow2dloq****\",
    \"sourceExecutionId\": \"exec-40e2e17ef7e046****\",
    \"OOSAssumeRole\": \"\",
    \"batchNumber\": 2,
    \"batchPauseOption\": \"Automatic\"
   }"
```

- In the following example, a command for rolling back OOS package installation is used to install the previous version of the WordPress package that is created in OOS for the ECS instances in the scaling group.

```
aliyun oos StartExecution --TemplateName ACS-ESS-RollingUpdateByConfigureOOSPackage --Parameters "{
    \"invokeType\": \"rollback\",
    \"scalingGroupId\": \"asg-bp18p2yfxow2dloq****\",
    \"packageVersion\": \"v3\",
    \"packageName\": \"wordpress\",
    \"sourceExecutionId\": \"exec-f4e61f2f21fe490****\",
    \"OOSAssumeRole\": \"\",
    \"batchNumber\": 2,
    \"batchPauseOption\": \"Automatic\"
   }"
```

3. View the execution details.

   When you run CLI commands to execute a rollback task, executions are also automatically created in OOS. You can also use the methods in Step 3 to view the details of an execution, such as the execution results and output.

OOS template parameters

The following tables list the parameters of the OOS public templates used in the preceding examples.

Parameters in the ACS-ESS-RollingUpdateByReplaceSystemDiskInScalingGroup template

Parameter	Description
invokeType	The type of the task. Valid values: • invoke: rolling update task • rollback: rollback task
scalingGroupId	The ID of the scaling group in which the task is to be executed.
scalingConfigurationId	The ID of the active scaling configuration in the scaling group.
imageId	The ID of the image used to replace the original image.
sourceImageId	The ID of the image used for the rollback task.

Parameter	Description
OOSAssumeRole	The RAM role used to execute the task. Default value: OOSServiceRole.
enterProcess	The scaling process that is suspended before the task is executed.
exitProcess	The scaling process that is resumed when the task is complete.
batchNumber	The number of batches in which the ECS instances in the scaling group are divided. The task is executed in these batches. Each batch contains at least one ECS instance.
batchPauseOption	Specifies whether and how to suspend the task. Valid values: • Automatic: The task is not suspended but is complete one time. • FirstBatchPause: The task is suspended when the first batch of executions are complete. • EveryBatchPause: The task is suspended when each batch of executions are complete.
sourceExecutionId	The execution ID of the source rolling update task when a rollback task is executed.

Note You can log on to the OOS console to view more parameter descriptions. For example, to view the parameter descriptions for the China (Hangzhou) region, see ACS-ESS-RollingUpdateByReplaceSystemDiskInScalingGroup.

Parameters in the ACS-ESS-RollingUpdateByRunCommandInScalingGroup template

Parameter	Description
invokeType	The type of the task. Valid values: • invoke: rolling update task • rollback: rollback task
scalingGroupId	The ID of the scaling group in which the task is to be executed.
commandType	The type of the script to be executed. The value of RunShellScript indicates a shell script.
invokeScript	The script that is executed on ECS instances during a rolling update task.
rollbackScript	The script that is executed on ECS instances during a rollback task.
OOSAssumeRole	The RAM role used to execute the task. Default value: OOSServiceRole.
enterProcess	The scaling process that is suspended before the task is executed.
exitProcess	The scaling process that is resumed when the task is complete.

Parameter	Description
batchNumber	The number of batches in which the ECS instances in the scaling group are divided. The task is executed in these batches. Each batch contains at least one ECS instance.
batchPauseOption	Specifies whether and how to suspend the task. Valid values: • Automatic: The task is not suspended but is complete one time. • FirstBatchPause: The task is suspended when the first batch of executions are complete. • EveryBatchPause: The task is suspended when each batch of executions are complete.
sourceExecutionId	The execution ID of the source rolling update task when a rollback task is executed.

Note You can log on to the OOS console to view more parameter descriptions. For example, to view the parameter descriptions for the China (Hangzhou) region, see ACS-ESS-RollingUpdateByRunCommandInScalingGroup.

Parameters in the ACS-ESS-RollingUpdateByConfigureOOSPackage template

Parameter	Description
invokeType	The type of the task. Valid values: • invoke: rolling update task • rollback: rollback task
scalingGroupId	The ID of the scaling group in which the task is to be executed.
packageName	The name of the software package.
packageVersion	The version of the software package.
action	The action to be taken on the software package. Valid values: • install • uninstall
OOSAssumeRole	The RAM role used to execute the task. Default value: OOSServiceRole.
enterProcess	The scaling process that is suspended before the task is executed.
exitProcess	The scaling process that is resumed when the task is complete.
batchNumber	The number of batches in which the ECS instances in the scaling group are divided. The task is executed in these batches. Each batch contains at least one ECS instance.

Parameter	Description
batchPauseOption	Specifies whether and how to suspend the task. Valid values: • Automatic: The task is not suspended but is complete one time. • FirstBatchPause: The task is suspended when the first batch of executions are complete. • EveryBatchPause: The task is suspended when each batch of executions are complete.
sourceExecutionId	The execution ID of the source rolling update task when a rollback task is executed.

Note You can log on to the OOS console to view more parameter descriptions. For example, to view the parameter descriptions for the China (Hangzhou) region, see ACS-ESS-RollingUpdateByConfigureOOSPackage.

8.Use Alibaba Cloud SDK for Python to execute rolling update tasks

Alibaba Cloud SDK for Python allows you to access Alibaba Cloud services without the need to perform complex coding. This topic describes how to use Alibaba Cloud SDK for Python to call API operations provided by Operation Orchestration Service (OOS) to execute rolling update tasks on Linux computers.

Prerequisites

- A scaling group is created and ECS instances are added to it.
- Python is installed on your local computer.
- A RAM user is created and an AccessKey pair is obtained for the RAM user.

Context

Rolling update tasks can be used to update the configurations of ECS instances in batches. You can use rolling update tasks to update the images of, execute scripts on, and install OOS packages on the running ECS instances in batches in a scaling group.

Procedure

To use Alibaba Cloud SDK for Python to execute a script on ECS instances on a local computer, perform the following operations:

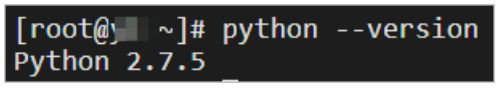
- Step 1: Install Alibaba Cloud SDK for Python
- Step 2: Execute a rolling update task
- Execute rollback tasks to handle exceptions in rolling update tasks

Step 1: Install Alibaba Cloud SDK for Python

1. Run the following command to check the version of Python:

   ```
   python --version
   ```

   If the version of Python is returned, Python is installed. The following figure shows an example command output.

2. Run the following command to install the SDK core library:

   ```
   pip install aliyun-python-sdk-core
   ```

3. Run the following command to install the OOS SDK:

   ```
   pip install aliyun-python-sdk-oos
   ```

Step 2: Execute a rolling update task

Sample code is provided in this step to demonstrate how to execute a script on ECS instances.

1. Create a Python script and enter the code used to execute the rolling update task.

For information about the OOS template parameters involved in the code, see OOS template parameters. The following content shows the sample code:

```
# -*- coding: UTF-8 -*-
from aliyunsdkcore.client import AcsClient
from aliyunsdkcore.acs_exception.exceptions import ClientException
from aliyunsdkcore.acs_exception.exceptions import ServerException
from aliyunsdkoos.request.v20190601 import StartExecutionRequest
import json
# Create an AcsClient instance.
client = AcsClient('<accessKeyId>', '<accessSecret>', 'cn-hangzhou')
# Create a request and configure the JSON data format.
request = StartExecutionRequest.StartExecutionRequest()
request.set_accept_format('json')
# Replace the template name based on the selected update method.
request.set_TemplateName("ACS-ESS-RollingUpdateByRunCommandInScalingGroup")
# Replace parameters based on the selected template.
parameters = {"invokeType": "invoke",
        "scalingGroupId": "asg-bp18p2yfxow2dloq****",
        "commandType": "RunShellScript",
        "invokeScript": "df -h\nifconfig",
        "rollbackScript": "df -h\nifconfig",
        "OOSAssumeRole": "",
        "exitProcess": [
          "ScaleIn",
          "ScaleOut",
          "HealthCheck",
          "AlarmNotification",
          "ScheduledAction"
         ],
        "enterProcess": [
          "ScaleIn",
          "ScaleOut",
          "HealthCheck",
          "AlarmNotification",
          "ScheduledAction"
         ],
        "batchNumber": 2,
        "batchPauseOption": "Automatic"}
request.set_Parameters(json.dumps(parameters))
# Initiate the API request and show the response.
response = client.do_action_with_exception(request)
print(response)
```

2. Execute the Python script and check the output.

   You can find information such as the execution ID of the rolling update task in the output. The following figure shows an example output.

   > **Note** To execute a rollback task, you must enter the execution ID of the source rolling update task.

Execute rollback tasks to handle exceptions in rolling update tasks

If exceptions occur during rolling update tasks or you want to use previous configurations after rolling upgrade tasks are executed, you can execute rollback tasks to restore the configurations of the ECS instances. Sample code is provided in this section to demonstrate how to roll back rolling update tasks that have been executed.

1. Create a Python script and enter the code used to execute the rollback task.

   For information about the OOS template parameters involved in the code, see OOS template parameters. The following content shows the sample code:

```
# -*- coding: UTF-8 -*-
from aliyunsdkcore.client import AcsClient
from aliyunsdkcore.acs_exception.exceptions import ClientException
from aliyunsdkcore.acs_exception.exceptions import ServerException
from aliyunsdkoos.request.v20190601 import StartExecutionRequest
import json
# Create an AcsClient instance.
client = AcsClient('<accessKeyId>', '<accessSecret>', 'cn-hangzhou')
# Create a request and configure the JSON data format.
request = StartExecutionRequest.StartExecutionRequest()
request.set_accept_format('json')
# Replace the template name based on the selected update method.
request.set_TemplateName("ACS-ESS-RollingUpdateByRunCommandInScalingGroup")
# Specify the parameters used in the rollback task.
parameters = {"invokeType": "rollback",
        "scalingGroupId": "asg-bp18p2yfxow2dlo****",
        "commandType": "RunShellScript",
        "rollbackScript": "df -h\nifconfig",
        "OOSAssumeRole": "",
        "sourceExecutionId": "exec-8fe4a73e9ffd423****",
        "batchNumber": 2,
        "batchPauseOption": "Automatic"}
request.set_Parameters(json.dumps(parameters))
# Initiate the API request and show the response.
response = client.do_action_with_exception(request)
print(response)
```

2. Execute the Python script and check the output.

   The following figure shows an example output.

OOS template parameters

This following table lists the parameters of the ACS-ESS-RollingUpdateByRunCommandInScalingGroup public template used in the preceding examples.

Parameter	Description
invokeType	The type of the task. Valid values: • invoke: rolling update task • rollback: rollback task
scalingGroupId	The ID of the scaling group in which the task is to be executed.
commandType	The type of script to be executed. The value of RunShellScript indicates a shell script.
invokeScript	The script that is executed on the ECS instances during a rolling update task.
rollbackScript	The script that is executed on the ECS instances during a rollback task.
OOSAssumeRole	The RAM role used to execute the task. Default value: OOSServiceRole.
enterProcess	The scaling process that is suspended before the task is executed.
exitProcess	The scaling process that is resumed when the task is complete.
batchNumber	The number of batches in which the ECS instances in the scaling group are divided. The task is executed in these batches. Each batch contains at least one ECS instance.
batchPauseOption	Specifies whether and how to suspend the task. Valid values: • Automatic: The task is not suspended but is complete one time. • FirstBatchPause: The task is suspended when the first batch of executions are complete. • EveryBatchPause: The task is suspended when each batch of executions are complete.
sourceExecutionId	The execution ID of the source rolling update task when a rollback task is executed.

Related information

- Use Alibaba Cloud CLI to execute rolling update tasks

9.Automatically deploy applications on ECS instances created by Auto Scaling

This topic uses CentOS as an example to describe how to use a Shell script to automatically deploy applications on ECS instances created by Auto Scaling.

Context

CentOS can be booted into the following runlevels:

- Runlevel 0: the halt runlevel.
- Runlevel 1: causes the system to start up in a single user mode under which only the root user can log on.
- Runlevel 2: boots the system into the multi-user mode with text-based console logon capability. This runlevel does not start the network.
- Runlevel 3: boots the system into the multi-user mode with text-based console logon capability. This runlevel starts the network.
- Runlevel 4: undefined runlevel. This runlevel can be configured to provide a custom boot state.
- Runlevel 5: boots the system into the multi-user mode. This runlevel starts the graphical desktop environment at the end of the boot process.
- Runlevel 6: reboots the system.

You can use a script to automatically install or update applications or run specific code on ECS instances created by Auto Scaling. To do so, add the script to a custom image and configure a command to run the script when the operating system boots. Then, select the custom image in a scaling configuration. After an ECS instance is created based on the scaling configuration, the script is automatically run on the ECS instance.

CentOS 6 and earlier versions use System V init as the initialization system, whereas CentOS 7 uses Systemd as the initialization system. System V init and Systemd are quite different in the ways that they operate. This topic describes how to configure a script in CentOS 6 and earlier versions and in CentOS 7 respectively.

CentOS 6 and earlier versions

This section describes how to configure a script in CentOS 6 and earlier versions.

1. Create a Shell script for testing.

```
#! /bin/sh
# chkconfig: 6 10 90
# description: Test Service
echo "hello world!"
```

The CHKCONFIG command in the preceding script sets the runlevel and priorities for running the script when the operating system boots and shuts down. The value 6 indicates runlevel 6, which means that the script is run when the operating system reboots. For more information about runlevels, see the background information in this topic. The value 10 indicates the priority for running the script when the operating system boots. The value 90 indicates the priority for running the script when the operating system shuts down. A priority ranges from 0 to 100, where a higher value indicates a lower priority.

> **Note** To make sure that the ECS instance is released only after the script is run on the ECS instance, change runlevel 6 to runlevel 0 in the preceding script.

2. Place the test script in the */etc/rc.d/init.d/* directory and run the `chkconfig --level 6 test on` command. Then, the script is run each time the operating system reboots.

> **Note** To make sure that the ECS instance is released only after the script is run on the ECS instance, change runlevel 6 to runlevel 0 in the preceding script. Then, the script is run each time the operating system shuts down.

For example, you can use the following sample script to automatically install PHPWind. You still need to enter the password for logging on to the database. Modify the script as required in actual use.

```
cd /tmp
echo "phpwind"
yum install -y \
unzip \
wget \
httpd \
php \
php-fpm \
php-mysql \
php-mbstring \
php-xml \
php-gd \
php-pear \
php-devel
chkconfig php-fpm on \
&& chkconfig httpd on
wget http://pwfiles.oss-cn-hangzhou.aliyuncs.com/com/soft/phpwind_v9.0_utf8.zip \
&& unzip -d pw phpwind_v9.0_utf8.zip \
&& mv pw/phpwind_v9.0_utf8/upload/* /var/www/html \
&& wget http://ess.oss-cn-hangzhou.aliyuncs.com/ossupload_utf8.zip -O ossupload_utf8.zip \
&& unzip -d ossupload ossupload_utf8.zip \
&& /bin/cp -rf ossupload/ossupload_utf8/* /var/www/html/src/extensions/ \
&& chown -R apache:apache /var/www/html
service httpd start && service php-fpm start
echo "Install CloudMonitor"
wget http://update2.aegis.aliyun.com/download/quartz_install.sh
chmod +x quartz_install.sh
bash quartz_install.sh
echo "CloudMonitor installed"
```

CentOS 7

This section describes how to configure a script in CentOS 7. CentOS 7 uses Systemd as the initialization system. After you configure a script by following the steps in this section, the script can be run when the system is shut down.

1. Create the script to run.
2. Create the *run-script-when-shutdown.service* file in the */etc/systemd/system* directory.

   Add the following content to the file. Change the value of the ExecStop variable to the absolute path of the script to run.

   ```
   [Unit]
   Description=service to run script when shutdown
   After=syslog.target network.target
   [Service]
   Type=simple
   ExecStart=/bin/true
   ExecStop=/path/to/script/to/run
   RemainAfterExit=yes
   [Install]
   WantedBy=default.target
   ```

3. Run the following commands to start the run-script-when-shutdown service:

   ```
   systemctl enable run-script-when-shutdown
   systemctl start run-script-when-shutdown
   ```

   > **Note**
   >
   > - You can configure the *run-script-when-shutdown* service to specify the script to run. This allows you to flexibly change the script to run.
   > - If the run-script-when-shutdown service is no longer needed, run the `systemctl disable run-script-when-shutdown` command to disable the service.

10.Use user data to automatically configure ECS instances

To provide more efficient and flexible scaling services, Auto Scaling allows you to configure user data in scaling configurations to customize ECS instances. You can pass in user data to perform automated configuration tasks on ECS instances, such as installing applications on ECS instances. This allows you to scale applications in a more secure and efficient manner.

Prerequisites

An Alibaba Cloud account is created. To create an Alibaba Cloud account, go to the account registration page.

To verify the effect of user data, you must log on to ECS instances. We recommend that you use Secure Shell (SSH) key pairs to log on to Linux instances. For more information, see Create an SSH key pair and Connect to a Linux instance by using an SSH key pair.

Context

An example is used in this topic to describe how to use user data in Auto Scaling. You can customize user data based on your business requirements.

For more information about user data, see Overview of ECS instance user data. Both Windows and Linux instances support user data. You can use user data for the following scenarios:

- Configure a script that is run when an ECS instance starts. In this way, you can customize the startup behavior of the ECS instance.
- Pass data to an ECS instance. You can reference the data on the ECS instance.

Compared with using open source IT infrastructure management tools such as Terraform, the method of using user data that is natively supported by Auto Scaling to manage the infrastructure is more efficient and secure. You only need to configure a Base64-encoded custom script and pass the script to a scaling configuration as user data. ECS instances created based on the scaling configuration can run the script upon startup to automatically deploy applications. In this way, you can scale applications. When you use user data, take note of the following items:

- The network type of the scaling group must be Virtual Private Cloud (VPC).
- The user data must be Base64-encoded.
- We recommend that you do not configure confidential information such as passwords and private keys in user data because user data is passed to instances in plaintext. If you must pass confidential information, we recommend that you encrypt the confidential information based on Base64 and decrypt the information on the instance.

If you call an API operation to create a scaling configuration, you can use the UserData parameter to configure user data. For more information, see CreateScalingConfiguration.

In addition to user data, you can use SSH key pairs, Resource Access Management (RAM) roles, and tags to customize ECS instances efficiently and flexibly. For more information, see Configure parameters in a scaling configuration to implement automatic deployment.

Procedure

Perform the following steps to configure user data in a scaling configuration:

1. Step 1: Prepare user data
2. Step 2: Create and enable a scaling group
3. Step 3: Verify the user data

Step 1: Prepare user data

You can configure a custom shell script in user data and enable the script to run when ECS instances start. When you customize a shell script, take note of the following items:

- Format: The first line must start with `#!`, such as `#! /bin/sh`.
- Limit: The script size cannot exceed 16 KB before the script is encoded in Base64.
- Frequency: The script is run only when instances are started for the first time.

1. Customize a shell script to configure the Domain Name System (DNS), Yellowdog Updater, Modified (YUM), and Network Time Protocol (NTP) services when an ECS instance starts.

   The following section shows the shell script:

   ```
   #! /bin/sh
   # Modify DNS
   echo "nameserver 8.8.8.8" | tee /etc/resolv.conf
   # Modify yum repo and update
   rm -rf /etc/yum.repos.d/*
   touch myrepo.repo
   echo "[base]" | tee /etc/yum.repos.d/myrepo.repo
   echo "name=myrepo" | tee -a /etc/yum.repos.d/myrepo.repo
   echo "baseurl=http://mirror.centos.org/centos" | tee -a /etc/yum.repos.d/myrepo.repo
   echo "gpgcheck=0" | tee -a /etc/yum.repos.d/myrepo.repo
   echo "enabled=1" | tee -a /etc/yum.repos.d/myrepo.repo
   yum update -y
   # Modify NTP Server
   echo "server ntp1.aliyun.com" | tee /etc/ntp.conf
   systemctl restart ntpd.service
   ```

2. Encode the shell script in Base64.

   The following section shows the Base64-encoded shell script:

   ```
   IyEvYmluL3NoCiMgTW9kaWZ5IEROUwplY2hvICJuYW1lc2VydmVyIDguOC44LjgiIHwgdGVlIC9ldGMvcmVz
   b2x2LmNvbmYKIyBNb2RpZnkgeXVtIHJlcG8gYW5kIHVwZGF0ZQpybSAtcmYgL2V0Yy95dW0ucmVwb3MuZ
   C8qCnRvdWNoIG15cmVwby5yZXBvCmVjaG8gIltiYXNlXSIgfCB0ZWUgL2V0Yy95dW0ucmVwb3MuZC9teXJl
   cG8ucmVwbwplY2hvICJuYW1lPW15cmVwbyIgfCB0ZWUgLWEgL2V0Yy95dW0ucmVwb3MuZC9teXJlcG8uc
   mVwbwplY2hvICJiYXNldXJsPWh0dHA6Ly9taXJyb3IuY2VudG9zLm9yZy9jZW50b3MiIHwgdGVlIC1hIC9ldG
   MveXVtLnJlcG9zLmQvbXlyZXBvLnJlcG8KZWNobyAiZ3BnY2hlY2s9MCIgfCB0ZWUgLWEgL2V0Yy95dW0uc
   mVwb3MuZC9teXJlcG8ucmVwbwplY2hvICJlbmFibGVkPTEiIHwgdGVlIC1hIC9ldGMveXVtLnJlcG9zLmQvb
   XlyZXBvLnJlcG8KeXVtIHVwZGF0ZSAteQojIE1vZGlmeSBOVFAgU2VydmVyCmVjaG8gInNlcnZlciBudHAxLm
   FsaXl1bi5jb20iIHwgdGVlIC9ldGMvbnRwLmNvbmYKc3lzdGVtY3RsIHJlc3RhcnQgbnRwZC5zZXJ2aWNl
   ```

Step 2: Create and enable a scaling group

1. Create a scaling group and view details of the scaling group after it is created.

   For more information, see Create a scaling group. Take note of the following items:

   - Minimum Number of Instances: Set this parameter to 1. An ECS instance is automatically created after the scaling group is enabled.

- Instance Template Source: Set this parameter to **Create from Scratch**.
- Network Type: Set this parameter to **VPC** and specify a VPC and a VSwitch.

2. Create and enable a scaling configuration after it is created.

   For more information, see Create a scaling configuration. Take note of the following items:

   - In the **Basic Configurations** step, set Image to Ubuntu 16.04 64-bit.
   - In the **System Configurations (Optional)** step, pass in the user data that were created in Step 1 and select an existing SSH key pair.

3. Enable the scaling group.

   For more information, see Enable a scaling group.

Step 3: Verify the user data

The minimum number of instances in the scaling group is set to 1. Therefore, an ECS instance is automatically created after the scaling group is enabled.

1. View the scaling activity.

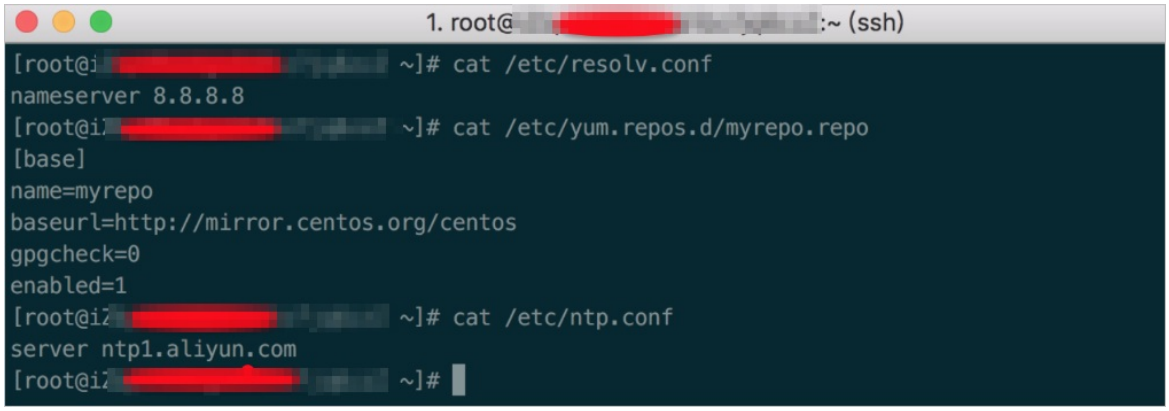
   For more information, see View the details of a scaling activity.

2. Log on to the ECS instance.

   For more information, see Connect to a Linux instance by using an SSH key pair.

3. Check the status of services on the instance.

   The following figure shows that the DNS, YUM, and NTP services are enabled on the ECS instance. This indicates that the user data configured in the scaling configuration is in effect.



11.Configure parameters in a scaling configuration to implement automatic deployment

Auto Scaling automatically adds and removes ECS instances based on your business requirements. To provide more flexible scaling services, Auto Scaling allows you to configure the following settings in a scaling configuration to customize ECS instances: tags, Secure Shell (SSH) key pairs, Resource Access Management (RAM) roles, and user data. This topic describes tags, SSH key pairs, RAM roles, and user data, and how to configure them in a scaling configuration.

Prerequisites

An Alibaba Cloud account is created. To create an Alibaba Cloud account, go to the account registration page.

Context

Auto Scaling can automatically scale ECS instances during peak or off-peak traffic hours, and can also automatically deploy applications on ECS instances. Auto Scaling allows you to configure various parameters in a scaling configuration to customize ECS instances efficiently and flexibly based on your business requirements.

- Tags

  For more information about tags, see Overview. Tags can be used to identify resources and user groups. Enterprises and individuals can use tags to categorize their ECS resources to simplify search and aggregation of resources. When you create a scaling configuration, you can select tags to be bound to the ECS instances that are created based on the scaling configuration.

  If you call an API operation to create a scaling configuration, you can use the Tags parameter to specify tags. For more information, see CreateScalingConfiguration.

- SSH key pairs

  For more information, see Overview. Alibaba Cloud supports only 2048-bit RSA key pairs. SSH key pairs apply only to Linux instances. After an SSH key pair is created, Alibaba Cloud stores the public key and offers you the private key.

  Compared with logons to ECS instances by using passwords, logons to ECS instances by using SSH key pairs are more efficient and secure. You can specify an SSH key pair when you create a scaling configuration. After Auto Scaling creates an ECS instance based on the scaling configuration, the instance stores the public key of the specified SSH key pair. You can use the private key to log on to the ECS instance from your local device. Note that:

  If you call an API operation to create a scaling configuration, you can use the KeyPairName parameter to specify an SSH key pair. For more information, see CreateScalingConfiguration.

- RAM roles

  RAM is a service provided by Alibaba Cloud to manage user identities and resource access permissions. RAM allows you to create different roles and grant different permissions on Alibaba Cloud services to each role.

For more information about RAM roles, see Overview. An ECS instance can assume a RAM role to obtain the permissions granted to the RAM role. When you specify a RAM role in a scaling configuration, make sure that ECS has been selected as the trusted entity of the RAM role. Otherwise, Auto Scaling cannot create ECS instances based on the scaling configuration.

If you call an API operation to create a scaling configuration, you can use the RamRoleName parameter to specify a RAM role. For more information, see CreateScalingConfiguration.

- User data

  For more information about user data of ECS instances, see Overview of ECS instance user data. Both Windows and Linux instances support user data. You can use user data for the following scenarios:

  - Configure a script that is run when an ECS instance starts. In this way, you can customize the startup behavior of the ECS instance.
  - Pass data to an ECS instance. You can reference the data on the ECS instance.

  Compared with using open source IT infrastructure management tools such as Terraform, the method of using user data that is natively supported by Auto Scaling to manage the infrastructure is more efficient and secure. You only need to configure a Base64-encoded custom script and pass the script to a scaling configuration as user data. ECS instances created based on the scaling configuration can run the script upon startup to automatically deploy applications. In this way, you can scale applications. Take note of the following items:

  - The network type of the scaling group must be Virtual Private Cloud (VPC).
  - The user data must be Base64-encoded.
  - We recommend that you do not configure confidential information, such as passwords and private keys in user data because user data is passed to instances in plaintext. If you must pass confidential information, we recommend that you encrypt the confidential information based on Base64 and decrypt the information on the instance.

  If you call an API operation to create a scaling configuration, you can use the UserData parameter to configure user data. For more information, see CreateScalingConfiguration.

Proper use of Auto Scaling can reduce your costs on servers, service management, and operations and maintenance (O&M). To help you understand and properly use Auto Scaling, this topic demonstrates how to configure the preceding parameters in a scaling configuration for Auto Scaling to automatically scale and customize ECS instances. Specifically, this topic demonstrates how to configure tags, an SSH key pair, a RAM role, and user data containing a custom script in a scaling configuration. When an ECS instance is created based on the scaling configuration, the tags are bound to the ECS instance, and the ECS instance assumes the RAM role. You can use the SSH key pair to log on to the ECS instance. The custom script is automatically run when the ECS instance starts.

Procedure

Perform the following steps to configure custom settings, including tags, an SSH key pair, a RAM role, and user data, in a scaling configuration:

1. Step 1: Prepare custom settings
2. Step 2: Apply the preceding settings
3. Step 3: Verify the preceding settings

Step 1: Prepare custom settings

Perform the following operations to create tags, an SSH key pair, a RAM role, and user data:

1. Create tags.

   For more information, see Create or bind a tag.

2. Create an SSH key pair.

   For more information, see Create an SSH key pair.

3. Create a RAM role.

   For more information, see Create a RAM role for a trusted Alibaba Cloud service. You can also use an existing RAM role. When you specify a RAM role in a scaling configuration, make sure that ECS has been selected as the trusted entity of the RAM role. Otherwise, Auto Scaling cannot create ECS instances based on the scaling configuration. For example, the RAM role `AliyunECSImageExportDefaultRole` grants the permission of exporting images. The trust policy of the RAM role allows all ECS instances in the current account to assume this RAM role. The following section shows the policy content:

   ```
   {
     "Statement": [
       {
         "Action": "sts:AssumeRole",
         "Effect": "Allow",
         "Principal": {
           "Service": [
             "ecs.aliyuncs.com"
           ]
         }
       }
     ],
     "Version": "1"
   }
   ```

   > **Note** `ecs.aliyuncs.com` in the preceding policy indicates that all ECS instances in the current account can assume this RAM role.

4. Prepare user data.

   For more information, see Overview of ECS instance user data. In this example, a shell script is provided in user data to write the following string to the */root/output10.txt* file when an ECS instance starts for the first time: `Hello World. The time is now {Current time}`. The following section shows the script:

   ```
   #! /bin/sh
   echo "Hello World. The time is now $(date -R)!" | tee /root/output10.txt
   ```

   The following section shows the Base64-encoded string of the script:

   ```
   IyEvYmluL3NoDQplY2hvICJIZWxsbyBXb3JsZC4gIFRoZSB0aW1lIGlzIG5vdyAkKGRhdGUgLVIpISIgfCB0ZWUgL3Jvb3Qvb3V0cHV0MTAudHh0
   ```

Step 2: Apply the preceding settings

Perform the following operations to create a scaling group and a scaling configuration and apply the preceding settings in the scaling configuration:

1. Create a scaling group and view details of the scaling group after it is created.

For more information, see Create a scaling group. Take note of the following items:

- Minimum Number of Instances: Set this parameter to 1. An ECS instance is automatically created after the scaling group is enabled.
- Instance Template Source: Set this parameter to **Create from Scratch**.
- Network Type: Set this parameter to **VPC** and specify a VPC and a VSwitch.

2. Create and enable a scaling configuration after it is created.

   For more information, see Create a scaling configuration. Take note of the following items:

   - In the **Basic Configurations** step, set Image to Ubuntu 16.04 64-bit.
   - In the **System Configurations (Optional)** step, select the tags, SSH key pair, RAM role, and user data that were created in Step 1.

3. Enable the scaling group.

   For more information, see Enable a scaling group.

Step 3: Verify the preceding settings

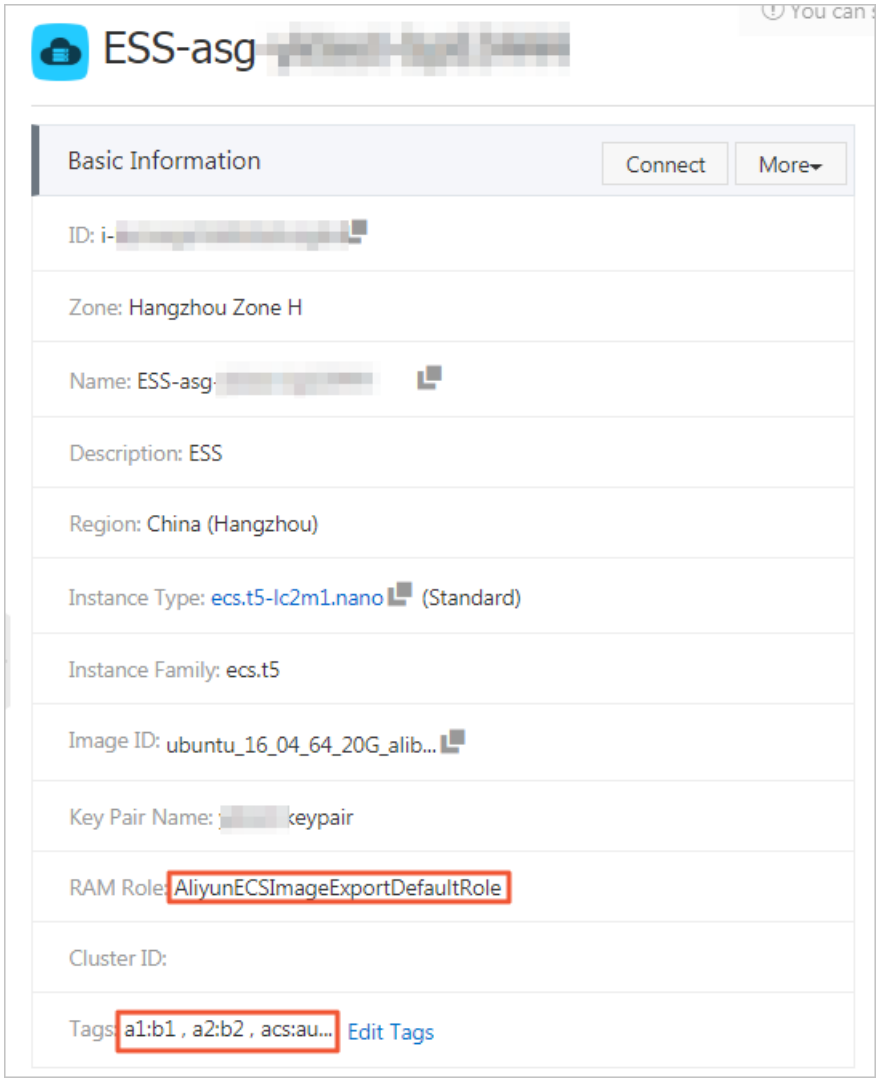
In Step 2, the minimum number of instances in the scaling group is set to 1. Therefore, an ECS instance is automatically created after the scaling group is enabled.

1. View the automatically created ECS instance.

   For more information, see View ECS instances.

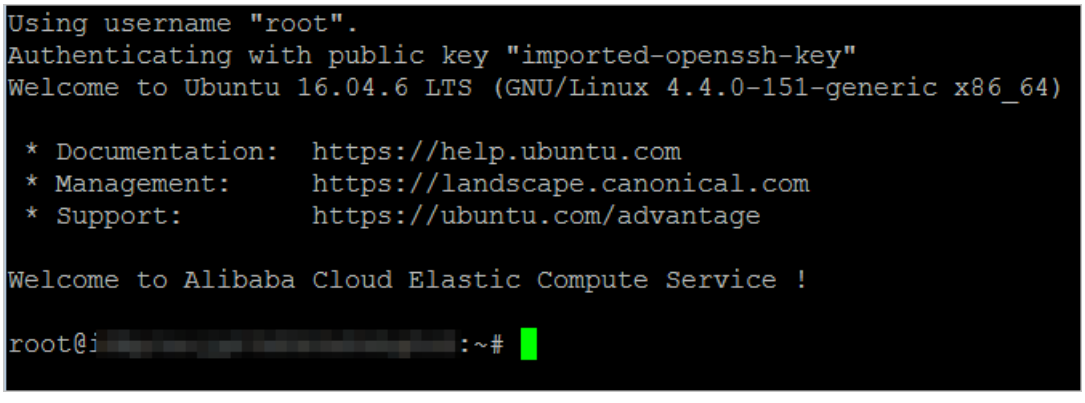
2. Click the instance ID to view details of the instance in the **ECS Instance ID/Name** column.

   The following figure shows details of the instance. You can find that the instance has assumed the RAM role and the tags have been bound to the instance.



3. Use the SSH key pair to log on to the instance.

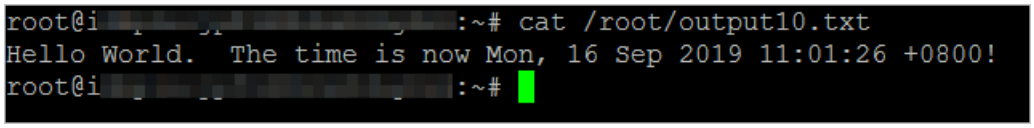
   For more information, see Connect to a Linux instance by using an SSH key pair. The following figure shows a successful logon. This indicates that the SSH key pair is in effect.



4. Run the following command to view the content of the */root/output10.txt* file:

```
cat /root/output10.txt
```

The following figure shows the file content. This indicates that the user data configured in the scaling configuration is in effect.



> **Note** A simple shell script is used in this example. You can create a script based on your requirements to customize more startup behaviors.

12.Reduce costs by configuring a cost optimization policy

This topic describes how to configure a cost optimization policy for a scaling group. A cost optimization policy can be used to create multiple types of ECS instances across different zones. This increases the success rate of creating ECS instances and reduces costs.

Prerequisites

- An Alibaba Cloud account is created. To create an Alibaba Cloud account, go to the account registration page.
- A virtual private cloud (VPC) is created. For more information, see Work with VPCs.
- Multiple vSwitches are created across different zones of the VPC. For more information, see Work with vSwitches.

Context

Auto Scaling supports creating ECS instances of multiple instance types in a scaling group. You can specify multiple instance types in a scaling configuration. Auto Scaling creates instances based on the priorities of the instance types. If resources of the instance type with the highest priority are insufficient, Auto Scaling automatically attempts to use the instance type with the next highest priority to create instances. This increases the success rate of creating ECS instances when resources of a specific instance type are insufficient. During business peaks, ECS instances of instance types that have higher specifications are required to respond to business requirements in a timely manner. In this case, Auto Scaling must focus on creating ECS instances with sufficient performance, instead of creating ECS instances of a specific instance type.

Auto Scaling supports creating ECS instances across different zones. You can select multiple vSwitches that are located in different zones when you create a scaling group. If the zone where a vSwitch resides does not have sufficient ECS instance resources, Auto Scaling automatically attempts to create instances in other zones. This increases the success rate of creating ECS instances. After you configure multiple zones, you can also configure a multi-zone scaling policy based on your actual business needs. Multi-zone scaling policies include priority policies, balanced distribution policies, and cost optimization policies.

> **Note**
> - Multi-zone scaling policies are available only for VPC-connected scaling groups.
> - The multiple-zone scaling policy of a scaling group cannot be modified.

Auto Scaling cannot create a preemptible instance when the market price of the instance exceeds your bid. This may affect your service. To avoid this issue, you can set your multi-zone scaling policy to cost optimization policy. If a preemptible instance fails to be created, Auto Scaling automatically attempts to create a pay-as-you-go instance of the same instance type. This increases the success rate of creating ECS instances and reduces costs. You can configure a cost optimization policy and multiple instance types at the same time to further increase the success rate of scaling. Auto Scaling attempts to create ECS instances in the scaling group that applies the cost optimization policy based on the unit prices of vCPUs in ascending order. Even if you set the billing method to pay-as-you-go, the cost optimization policy ensures that you can use ECS instance resources at the lowest cost.

Procedure

1. Create a scaling group.

   This step describes parameters related to the multi-zone scaling policy. For more information about other parameters of the scaling group, see Create a scaling group.

   i. Set the network type to **VPC** and select multiple vSwitches in the same VPC.

      A vSwitch belongs to only one zone. After you configure multiple vSwitches for the scaling group, the scaling group can create ECS instances in multiple zones. This helps you utilize available ECS resources in different zones based on your requirements.

   ii. Set the multi-zone scaling policy to **Cost Optimization Policy**.

   iii. Configure other parameters.

2. Create a scaling configuration.

   This step describes parameters related to the multi-zone scaling policy. For more information about other parameters of the scaling configuration, see Create a scaling configuration.

   i. Set the billing method to **Preemptible Instance**.

   ii. Select multiple instance types. You can select up to 10 instance types.

   - We recommend that you select instance types with similar performance in terms of the vCPU, memory, physical processor, clock speed, internal network bandwidth, or packet forwarding rate.
   - We recommend that you set a maximum bid for each instance type. If you use automatic bidding, Auto Scaling bids for and creates preemptible instances at the market price.
   - The configurations of I/O optimized instances vary greatly from those of non-I/O optimized instances. If you choose these two types of instances at the same time, the success rate of creating instances cannot be significantly increased.

   iii. Configure other parameters.

3. Enable the scaling group.
4. Create a scaling rule.

   This step describes parameters for creating a simple scaling rule to verify the cost optimization policy. For more information about other parameters of the scaling rule, see Create a scaling rule.

   i. Set Rule Type to **Simple Scaling Rule**.

   ii. Set Operation to **Add 1 Instances**.

   iii. Configure other parameters.

5. Execute the scaling rule.

Verification

Assume that in the preceding procedure, you have specified a vSwitch in Qingdao Zone B and a vSwitch in Qingdao Zone C for the scaling group, and specified the ecs.sn1.large and ecs.sn1.xlarge instance types for the scaling configuration. The billing method is set to Preemptible Instance. Therefore, instances of a specific instance type have two unit prices of vCPUs. One is for preemptible instances, and the other is for pay-as-you-go instances.

> **Notice** The prices listed in this topic are only for reference. Refer to the buy page of ECS instances for actual prices.

Based on the preceding settings of the instance types and billing method, you have four plans for creating instances. The following table lists the four plans by vCPU unit price in ascending order.

No.	Instance Type	Billing method	vCPU	Market price of instance (RMB per hour)	Unit price of vCPU (RMB per hour)
Plan 1	ecs.sn1.xlarge	Preemptible instance	8	0.158	0.01975
Plan 2	ecs.sn1.large	Preemptible instance	4	0.088	0.022
Plan 3	ecs.sn1.xlarge	Pay-as-you-go	8	1.393	0.174125
Plan 4	ecs.sn1.large	Pay-as-you-go	4	0.697	0.17425

Expected process for creating instances: During a scale-out event, Auto Scaling preferentially creates ECS instances based on Plan 1. If instances fails to be created in both Zone B and Zone C due to insufficient resources, Auto Scaling attempts to create ECS instances based on Plan 2, Plan 3, and Plan 4 in sequence.

Execute the scaling rule to trigger a scale-out event during which an ECS instance is created and added to the scaling group. In the Auto Scaling console, go to the **ECS Instances** page of the scaling group and click the created ECS instance to view its instance type and billing method. In this example, the instance type is **ecs.sn1.xlarge** and the billing method is **Pay-As-You-Go-Preemptible Instance**. This indicates that costs are reduced.

13.Use Alibaba Cloud ESS SDK to create a multi-zone scaling group

This topic describes how to use Alibaba Cloud ESS SDK for Java or Python to create a multi-zone scaling group.

Prerequisites

An Alibaba Cloud account is created. To create an Alibaba Cloud account, go to the account registration page.

Context

The network type of a scaling group can be Virtual Private Cloud (VPC) or classic network. When you create a VPC-connected scaling group, you must configure a VSwitch for the scaling group. After the scaling group is created, all ECS instances that are created for the scaling group use this VSwitch.

Originally, Auto Scaling allows a VPC-connected scaling group to have only one VSwitch configured. A VSwitch belongs to only one zone. If ECS instances cannot be created in the zone where the VSwitch resides due to reasons such as insufficient resources, the scaling configurations, scaling rules, and event-triggered tasks in the scaling group become invalid.

To address the preceding issue and improve the availability of scaling groups, the VSwitchIds.N parameter is added to allow you to create multi-zone scaling groups. When you create a scaling group, you can use the VSwitchIds.N parameter to configure multiple VSwitches for the scaling group. When ECS instances cannot be created in the zone where a VSwitch resides, Auto Scaling automatically switches to the zone where a different VSwitch resides. When you use this parameter, take note of the following items:

- If the VSwitchIds.N parameter is specified, the VSwitchId parameter is ignored.
- The VSwitchIds.N parameter allows you to specify up to five VSwitches within a VPC across multiple zones when you create a scaling group. Valid values of N: 1 to 5.
- VSwitches specified in the VSwitchIds. N parameter must be within the same VPC.
- In the VSwitchIds.N parameter, N indicates the priority of each VSwitch. The VSwitch with N set to 1 has the highest priority to create ECS instances. The greater the N value, the lower the priority.
- When an ECS instance cannot be created in the zone where the VSwitch with the highest priority resides, the instance will be created in the zone where the VSwitch with the second highest priority resides. We recommend that you specify multiple VSwitches across different zones in the same region to avoid failing to create ECS instances due to insufficient resources in a single zone and improve the availability of scaling groups.

Use Alibaba Cloud ESS SDK for Java to create a multi-zone scaling group

1. Install Alibaba Cloud ESS SDK for Java.

   Download the *aliyun-java-sdk-core* and *aliyun-java-sdk-ess* dependency libraries. You can visit Maven Central to search for and download the corresponding JAR packages. The JAR package version for *aliyun-java-sdk-ess* must be V2.1.3 or later and the package version for *aliyun-java-sdk-core* must be the latest.

You can also use Apache Maven to manage the dependency libraries of your Java projects by adding the following dependencies to the *pom.xml* file:

```
<dependency>
<groupId>com.aliyun</groupId>
<artifactId>aliyun-java-sdk-ess</artifactId>
<version>2.1.3</version>
</dependency>
<dependency>
<groupId>com.aliyun</groupId>
<artifactId>aliyun-java-sdk-core</artifactId>
<version>3.5.0</version>
</dependency>
```

2. Use SDK for Java to create a multi-zone scaling group.

   After Alibaba Cloud ESS SDK for Java is imported to a Java project, you can use the SDK code to create a multi-zone scaling group. The following section shows the sample code:

```
public class EssSdkDemo {
public static final String     REGION_ID       = "cn-hangzhou";
public static final String     AK              = "ak";
public static final String     AKS             = "aks";
public static final Integer    MAX_SIZE        = 10;
public static final Integer    MIN_SIZE        = 1;
public static final String     SCALING_GROUP_NAME = "TestScalingGroup";
// The list of VSwitches. The VSwitches are listed in descending order of priority. The first VSwitch has t
he highest priority.
public static final String[]    vswitchIdArray    = { "vsw-id1", "vsw-id2", "vsw-id3", "vsw-id4", "vsw-id5" };
public static final List<String> vswitchIds       = Arrays.asList(vswitchIdArray);
public static void main(String[] args) throws Exception {
  IClientProfile clientProfile = DefaultProfile.getProfile(REGION_ID, AK, AKS);
  IAcsClient client = new DefaultAcsClient(clientProfile);
  createScalingGroup(client);
}
/**
 * Create a multi-zone scaling group.
 * @param client
 * @return
 * @throws Exception
*/
public static String createScalingGroup(IAcsClient client) throws Exception {
  CreateScalingGroupRequest request = new CreateScalingGroupRequest();
  request.setRegionId("cn-beijing");
  request.setMaxSize(MAX_SIZE);
  request.setMinSize(MIN_SIZE);
  request.setScalingGroupName(SCALING_GROUP_NAME);
  request.setVSwitchIds(vswitchIds);
  CreateScalingGroupResponse response = client.getAcsResponse(request);
  return response.getScalingGroupId();
}
}
```

   In the preceding code, the VSwitches are listed in descending order of priority. The first VSwitch has the highest priority.

Use Alibaba Cloud ESS SDK for Python to create a multi-zone scaling group

1. Install Alibaba Cloud ESS SDK for Python.

   To install Alibaba Cloud ESS SDK for Python, you must download and install the *aliyun-python-sdk-ess* and *aliyun-python-sdk-core* dependency libraries. We recommend that you use pip to install Python dependency libraries. For more information, visit Installation-pip. After pip is installed, run the `pip install aliyun-python-sdk-ess==2.1.3 pip install aliyun-python-sdk-core==3.5.0` command to install the dependency libraries.

2. Use SDK for Python to create a multi-zone scaling group.

   After Alibaba Cloud ESS SDK for Python is imported to a Python project, you can use the SDK code to create a multi-zone scaling group. The following section shows the sample code:

```
# coding=utf-8
import json
import logging
from aliyunsdkcore import client
from aliyunsdkess.request.v20140828.CreateScalingGroupRequest import CreateScalingGroupRequest
logging.basicConfig(level=logging.INFO,
        format='%(asctime)s %(filename)s[line:%(lineno)d] %(levelname)s %(message)s',
        datefmt='%a, %d %b %Y %H:%M:%S')
# Replace the following ak and aks values with your own AccessKey ID and AccessKey secret:
ak = 'ak'
aks = 'aks'
scaling_group_name = 'ScalingGroupTest'
max_size = 10
min_size = 1
vswitch_ids = ["vsw-id1", "vsw-id2", "vsw-id3", "vsw-id4", "vsw-id5"]
region_id = 'cn-beijing'
clt = client.AcsClient(ak, aks, region_id)
```

```
def _create_scaling_group():
request = CreateScalingGroupRequest()
request.set_ScalingGroupName(scaling_group_name)
request.set_MaxSize(max_size)
request.set_MinSize(min_size)
request.set_VSwitchIds(vswitch_ids)
response = _send_request(request)
return response.get('ScalingGroupId')
```

```
def _send_request(request):
request.set_accept_format('json')
try:
  response_str = clt.do_action(request)
  logging.info(response_str)
  response_detail = json.loads(response_str)
  return response_detail
except Exception as e:
  logging.error(e)
if __name__ == '__main__':
scaling_group_id = _create_scaling_group()
print 'Scaling group created successfully. Scaling group ID:' + str (scaling_group_id)
```

In the preceding code, the VSwitches are listed in descending order of priority. The first VSwitch has the highest priority.

Related information

Reference

- CreateScalingGroup
- CreateScalingConfiguration
- Configure parameters in a scaling configuration to implement automatic deployment

14.Use performance metrics to measure Auto Scaling

You can set weights for different instance types based on performance metrics such as the number of vCPUs. You can specify the capacity of a single instance of a specific instance type in a scaling group. After the weights are set, Auto Scaling can measure the capacity of a scaling group by using performance metrics. This helps you determine the overall performance of a scaling group in a more precise manner.

Context

By default, Auto Scaling measures the capacity of a scaling group based on the number of ECS instances in the scaling group. If you specify only a single instance type in the active scaling configuration of a scaling group, the number of instances in the scaling group is proportional to the overall performance of the scaling group. However, if you specify multiple instance types that have different specifications in the active scaling configuration of a scaling group and create instances of multiple instance types, the number of instances in the scaling group cannot precisely reflect the overall performance of the scaling group. For example, the performance of 10 ecs.c5.xlarge (4 vCPUs and 8 GiB memory) instances is twice that of 10 ecs.c5.large (2 vCPUs and 4 GiB memory) instances.

You can specify the weights of instance types. Even if Auto Scaling creates multiple instances of different instance types in a scaling group, you can precisely measure the performance of the scaling group. For example, if you set the weights of instance types based on the number of vCPUs in a scaling group, the capacity of the scaling group indicates the total number of vCPUs of all instances in the scaling group.

Terms

Term	API parameter	Description
weight	WeightedCapacity	The weight of an instance type. You can set the weight of an instance type based on performance metrics such as the number of vCPUs. You can specify the capacity of a single instance of a specific instance type in a scaling group.
total capacity	TotalCapacity	The total capacity of all instances in a scaling group.
maximum capacity	MaxSize	The maximum value of the total capacity of a scaling group. **Note** After a scale-out event is executed in a scaling group, the total capacity of the scaling group may exceed the maximum capacity. This is because the maximum capacity may not be divisible by the weight of a single instance type. However, the extra capacity is less than the maximum weight.
minimum capacity	MinSize	The minimum value of the total capacity of a scaling group.

Term	API parameter	Description
expected capacity	DesiredCapacity	The expected value of the total capacity of a scaling group. Auto Scaling ensures that the total capacity is no less than the expected capacity. **Note** After a scale-out event is executed in a scaling group, the total capacity of the scaling group may exceed the expected capacity. This is because the expected capacity may not be divisible by the weight of a single instance type. However, the extra capacity is less than the maximum weight.

Rules for scaling

- When the total capacity of a scaling group is less than the expected capacity or the minimum capacity, scale-out events are triggered.
- When the total capacity of a scaling group is greater than or equal to the sum of the expected capacity and the maximum weight, scale-in events are triggered.

Note Auto Scaling performs automatic scaling based preferentially on the scaling policy specified for a scaling group. If you configure a cost optimization policy for a scaling group, Auto Scaling creates instances based on the weighted unit prices in ascending order. Auto Scaling releases instances based on the weighted unit prices in descending order. For information about how to calculate weighted unit prices, see Calculation of weighted unit prices.

Usage notes

- You must set weights for all instance types in a scaling group.
- If you delete an instance type specified in the active scaling configuration of a scaling group, the weights of instances of this instance type remain unchanged in the scaling group.
- After you modify the weight of an instance type in a scaling group, Auto Scaling recalculates the capacity of the scaling group based on the modified weight if instances of this instance type have been created. **Scaling activities may be triggered**.

Procedure

In this example, a scaling configuration is used as the configuration source of a scaling group to set the weights of instance types.

Note You can also use a launch template as the configuration source. You can specify the LaunchTemplateOverride.N.InstanceType and LaunchTemplateOverride.N.WeightedCapacity parameters in the CreateScalingGroup operation to set the weights of instance types. For more information, see CreateScalingGroup.

1. Create a scaling group.

   This step describes the parameters related to the multi-zone scaling policy. For information about other parameters of a scaling group, see Create a scaling group.

i. Set Network Type to **VPC** and select multiple vSwitches in the same VPC.

One vSwitch belongs to only one zone. After you configure multiple vSwitches for the scaling group, Auto Scaling can create ECS instances in multiple zones. This helps you utilize available ECS resources in different zones.

ii. Set Multi-zone Scaling Policy to **Cost Optimization Policy**.

iii. Configure other parameters for the scaling group.

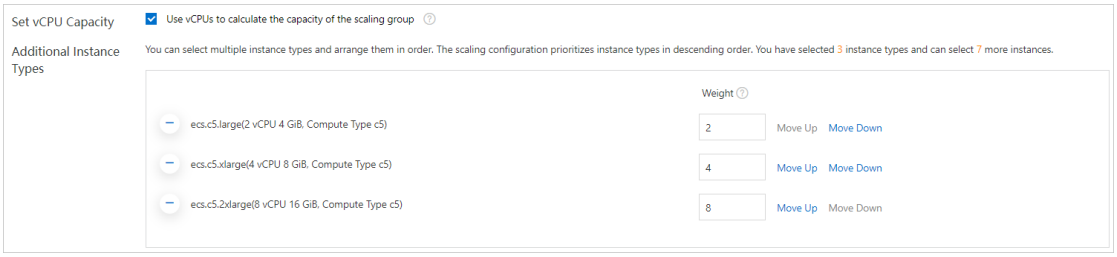
2. Create a scaling configuration.

This step describes the parameters for setting the weight of an instance type based on the number of vCPUs. For information about other parameters of a scaling configuration, see Create a scaling configuration.

i. Set Billing Method to **Pay-As-You-Go**.

ii. Select multiple instance types. You can select up to 10 instance types.

iii. Select **Set vCPU Capacity**. By default, the system sets weights for all selected instance types based on the number of vCPUs.



You can customize the weights of instance types. When you customize weights, we recommend that you take note of the following items:

- Use performance metrics related to instance types to set weights. For example, you can set weights based on the number of CPU cores or the amount of memory. You can use an instance type that has 1 vCPU and 1 GiB memory or provides the minimum performance as the capacity unit of a scaling group. The capacity of the scaling group is calculated based on the capacity unit.
- Set appropriate weight values to ensure that the current capacity of a scaling group is two to three times the maximum weight of instance types.
- Set the weights of different instance types to values that have small differences. For example, do not set the weight of an instance type that has lower specifications to 1 and the weight of an instance type that has higher specifications to 200. If the difference between the weights of instance types in a scaling group is large, the overall costs of the scaling group may increase.

For information about the priority of multiple instance types when they are used to create instances, see Calculation of weighted unit prices.

iv. Configure other parameters for the scaling configuration.

3. Enable the scaling group.

4. Create a scaling rule.

This step describes the parameters for creating a simple scaling rule to verify the cost optimization policy. For information about other parameters of a scaling rule, see Create a scaling rule.

i. Set Rule Type to **Simple Scaling Rule**.

ii. Set Operation to **Add 10 Capacity Unit**.

iii. Configure other parameters for the scaling rule.

5. Execute the scaling rule.

   In this example, the weighted unit price of the ecs.c5.2xlarge instance type is the lowest. Therefore, Auto Scaling creates two instances of the ecs.c5.2xlarge instance type for the scaling group. This adds 16 capacity units to the scaling group.

Calculation of weighted unit prices

If you set the multi-zone scaling policy to **Cost Optimization Policy** for a scaling group and set the weights of instance types, Auto Scaling attempts to create instances based on the weighted unit prices in ascending order. For more information, see Reduce costs by configuring a cost optimization policy.

The following table provides examples on how to calculate weighted unit prices of different instance types.

> **Note** The market prices of instance types in the following table are for reference only. For information about the actual market prices, see the Pricing tab on the Elastic Compute Service page.

Instance type	vCPU	Market price	Weight	Weighted unit price
ecs.c5.large	2	USD 0.073/Hour	2	USD 0.037/Hour
ecs.c5.xlarge	4	USD 0.144/Hour	4	USD 0.036/Hour
ecs.c5.2xlarge	8	USD 0.288/Hour	8	USD 0.036/Hour

15.Set rules for generating sequential and unique hostnames

This topic describes how to set rules for hostnames in a scaling configuration to generate sequential and unique hostnames for ECS instances during scale-out events. This helps you manage ECS instances in a more efficient manner.

Context

Auto Scaling can create one or more ECS instances during a single scale-out event in a scaling group based on scaling rules. Auto Scaling can also create multiple ECS instances during multiple scale-out events. You can use one of the following methods to set rules for hostnames in a scaling configuration or launch template:

- If you want to make the hostnames of ECS instances in a scaling group sequential and unique, you must set rules for the hostnames in a scaling configuration instead of in a launch template. For more information, see (Recommended) Sorting at a fixed-value increment and Dynamic sorting based on extended sequential values.

  > **Note** The hostnames of ECS instances in a scaling group are sequentially generated but not necessarily consecutive. For example, the hostnames of created ECS instances in a scaling group are ess-node-0999, ess-node-1000, and ess-node-1002. This indicates that the ess-node-1001 ECS instance fails to start normally. Auto Scaling considers this instance unhealthy, removes it from the scaling group, and then creates another instance whose hostname is ess-node-1002.

- If you want to make the hostnames of only ECS instances that are created during a single scale-out event sequential and unique, you can set the hostnames based on the specified sorting rule. For more information, see Batch configure sequential names or hostnames for multiple instances.
- If you have no naming requirements for hostnames, you can use regular hostnames without the need to set hostnames based on the preceding rules. For example, if you set the hostname to hostname for a scaling group, the hostnames of all created ECS instances in the scaling group are hostname.

In this topic, two example scenarios are used to describe how to set rules for generating sequential and unique hostnames by using the Auto Scaling console and by calling API operations.

Scenario 1: Set sequential and unique hostnames in the console

1. Create a scaling group.

   For more information, see Create a scaling group.

2. Create a scaling configuration and enable it.

   In the **System Configurations (Optional)** step, specify the naming convention in the **Host** section. In this example, enter ess-node-(AUTO_INCREMENT)[0,3]-ecshost.

   > **Note** In this example, the naming convention specifies to increment the hostname by a fixed value for each created ECS instance. For more information, see (Recommended) Sorting at a fixed-value increment.

   For more information, see Create a scaling configuration.

3. Enable the scaling group.

For more information, see Enable a scaling group.

4. Create and manually execute a scaling rule.
   i. Create a scaling rule. For more information, see Create a scaling rule.

      In this example, set Rule Type to Simple Scaling Rule and Operation to Add 3 Instances.

   ii. Manually execute the scaling rule. For more information, see Execute a scaling rule.

      After the scaling rule is executed, the hostnames of the three ECS instances are displayed as ess-node-000-ecshost, ess-node-001-ecshost, and ess-node-002-ecshost.

Scenario 2: Set sequential and unique hostnames by calling API operations

1. Call the CreateScalingGroup operation to create a scaling group.
2. Call the CreateScalingConfiguration operation to create a scaling configuration.

   Set the HostName parameter to ess-node-(AUTO_INCREMENT)[0,3]-ecshost.

   > **Note** In this example, the naming convention specifies to increment the hostname by a fixed value for each created ECS instance. For more information, see (Recommended) Sorting at a fixed-value increment.

3. Call the EnableScalingGroup operation to enable the scaling group.
4. Create and manually execute a scaling rule.
   i. Call the CreateScalingRule operation to create a scaling rule.

      In this example, a simple scaling rule is created to add three ECS instances.

   ii. Call the ExecuteScalingRule operation to execute the scaling rule.

      After the scaling rule is executed, the hostnames of the three ECS instances are displayed as ess-node-000-ecshost, ess-node-001-ecshost, and ess-node-002-ecshost.

(Recommended) Sorting at a fixed-value increment

Hostnames are in the *name_prefix(AUTO_INCREMENT)[begin_number,bits]name_suffix* format.

Hostname description

Segment	Required	Description	Example
name_prefix	Yes	The prefix of the hostname.	ess-node-
(AUTO_INCREMENT)	Yes	The fixed value used to indicate the sorting method.	(AUTO_INCREMENT)

Segment	Required	Description	Example

[begin_number,bits]	Yes	The sequential value of the hostname. After this segment is specified, the sequential values of hostnames are incremented.
		
		Note By default, the system sequentially increments the values. If a created ECS instance in a scaling group cannot be started, Auto Scaling removes the instance from the scaling group and creates another instance. Therefore, the hostnames of ECS instances in the scaling group may not be consecutively incremented.
		
		• begin_number: the start value of the sequential value. Valid values: 0 to 999999.
		  ◦ When Auto Scaling creates instances in a scaling group for the first time, the start value that you specified takes effect. If the start value is not specified, *0* is used.
		  ◦ When Auto Scaling creates instances in a scaling group not for the first time, the start value increments starting from the maximum sequential value of existing hostnames in the scaling group.
		• bits: the number of digits of the sequential value. Valid values: 1 to 6. Default value: *6*.
		
		Note We recommend that you set *bits* to at least *3*. Otherwise, the upper limit of the sequential values may be reached in a short period of time. If Auto Scaling needs to create more ECS instances after the upper limit is reached, an error is reported and the scale-out event is suspended. In this case, you must set the rules for generating hostnames again.
		
		The *[begin_number,bits]* segment cannot contain spaces. By default, when the number of digits of the specified *begin_number* value is greater than the *bits* value, *bits* is set to *6*.
			[0,6]

name_suffix	No	The suffix of the instance name or hostname.	-ecshost

Hostname examples

Example	Existing hostname with the maximum sequential value in a scaling group	Hostname (three created ECS instances)	Description
ess-node-(AUTO_INCREMENT)[0,3]-ecshost	N/A	ess-node-000-ecshost, ess-node-001-ecshost, and ess-node-002-ecshost.	The number of digits of all sequential values is the *bits* value. When Auto Scaling creates instances in a scaling group for the first time, the sequential value starts from the *begin_number* value and sequentially increments based on the number of created instances.
• ess-node-(AUTO_INCREMENT)[]-ecshost • ess-node-(AUTO_INCREMENT)[,]-ecshost	N/A	ess-node-000000-ecshost, ess-node-000001-ecshost, and ess-node-000002-ecshost.	If *begin_number* is not specified, *begin_number* is set to *0*. If *bits* is not specified, *bits* is set to *6*.
ess-node-(AUTO_INCREMENT)[99,1]-ecshost	ess-node-000099-ecshost	ess-node-000100-ecshost, ess-node-000101-ecshost, and ess-node-000102-ecshost.	• When Auto Scaling creates instances in a scaling group not for the first time, the sequential value increments starting from the maximum sequential value of existing hostnames in the scaling group. • By default, when the number of digits of the specified *begin_number* value is greater than the *bits* value, *bits* is set to *6*.
ess-node-(AUTO_INCREMENT)[0,2]-ecshost	ess-node-99-ecshost	An error is reported and the scale-out event is suspended.	• When Auto Scaling creates instances in a scaling group not for the first time, the sequential value increments starting from the maximum sequential value of existing hostnames in the scaling group. • If Auto Scaling needs to create more ECS instances after the upper limit is reached, an error is reported and the scale-out event is suspended. In this case, you must set the rules for generating hostnames again.

Example	Existing hostname with the maximum sequential value in a scaling group	Hostname (three created ECS instances)	Description
ess-node-(AUTO_INCREMENT)[0,4]	ess-node-0998-ecshost	ess-node-0999, ess-node-1000, and ess-node-1002.	• When Auto Scaling creates instances in a scaling group not for the first time, the sequential value increments starting from the maximum sequential value of existing hostnames in the scaling group. • By default, the system sequentially increments the values. If a created ECS instance in a scaling group cannot be started, Auto Scaling removes the instance from the scaling group and creates another instance. Therefore, the hostnames of ECS instances in the scaling group may not be consecutively incremented. In this example, the hostname of the ECS instance that fails to start is ess-node-1001.

Dynamic sorting based on extended sequential values

Hostnames are in the *name_prefix(ess_extend_begin,ess_extend_bits)[begin_number,bits]name_suffix* format.

Hostname description

Segment	Required	Description	Example
name_prefix	Yes	The prefix of the hostname.	ess-node-

Segment	Required	Description	Example
(ess_extend_begin,ess_extend_bits)	Yes	The extended sequential value of the hostname. When the base sequential value of existing hostnames in a scaling group is equal to the maximum sequential value, one value is added to the extended sequential value. Then, the base sequential value increments from 0 again until the upper limit is reached. • ess_extend_begin: the start value of the extended sequential value. Valid values: 0 to ZZZ. Each valid value ranges from 0 to 9, a to z, and A to Z. For example, if one value is added to 9, a is obtained. If one value is added to z, A is obtained. ◦ When Auto Scaling creates instances in a scaling group for the first time, the start value that you specified takes effect. If the start value is not specified, *0* is used. ◦ When Auto Scaling creates instances in a scaling group not for the first time, the start value increments starting from the maximum sequential value of existing hostnames in the scaling group. • ess_extend_bits: the number of digits of the extended sequential value. Valid values: 1 to 3. Default value: *3*. **Notice** If Auto Scaling needs to create more ECS instances after the upper limits of extended sequential values and base sequential values are both reached, an error is reported and the scale-out event is suspended. In this case, you must set the rules for generating hostnames again. The *(ess_extend_begin,ess_extend_bits)* segment cannot contain spaces. By default, when the number of digits of the specified *ess_extend_begin* value is greater than the *bits* value, *bits* is set to *3*.	(0,3)
		The base sequential value of the hostname. After this segment is specified, the base sequential values of hostnames are incremented to the maximum value. Then, one value is added to the extended sequential value and the base sequential value increments from 0 again until the upper limit is reached.	

Segment	Required	Description	Example
[begin_number,bits]	Yes	**Notice** By default, the system sequentially increments the values. If a created ECS instance in a scaling group cannot be started, Auto Scaling removes the instance from the scaling group and creates another instance. Therefore, the hostnames of ECS instances in the scaling group may not be consecutively incremented. • begin_number: the start value of the base sequential value. Valid values: 0 to 999999. ◦ When Auto Scaling creates instances in a scaling group for the first time, the start value that you specified takes effect. If the start value is not specified, *0* is used. ◦ When Auto Scaling creates instances in a scaling group not for the first time, the start value increments starting from the maximum sequential value of existing hostnames in the scaling group. • bits: the number of digits of the base sequential value. Valid values: 1 to 6. Default value: *6*. **Notice** • When the sum of the maximum base sequential value of existing hostnames and the number of instances to be created in a scaling group is greater than or equal to the maximum base sequential value, the hostnames of instances in the scaling group may not be consecutively incremented. To ensure that the hostnames of ECS instances in the scaling group are consecutively incremented, we recommend that you set the number of digits of base sequential values to at least 3. • If Auto Scaling needs to create more ECS instances after the upper limits of extended sequential values and base sequential values are both reached, an error is reported and the scale-out event is suspended. In this case, you must set the rules for generating hostnames again.	[0,6]
name_suffix	No	The suffix of the instance name or hostname.	-ecshost

Hostname examples

The *[begin_number,bits]* segment cannot contain spaces. By default, when the number of digits of the specified *begin_number* value is greater than the *bits* value, *bits* is set to *6*.

Example	Existing hostname with the maximum sequential value in a scaling group	Hostname (three created ECS instances)	Description
ess-node-(0,3)[0,3]-ecshost	N/A	ess-node-000000-ecshost, ess-node-000001-ecshost, and ess-node-000002-ecshost.	When Auto Scaling creates instances in a scaling group for the first time, the sequential values for hostnames of the instances are set based on the following rules: • Extended sequential value: The number of digits of all extended sequential values is the *ess_extend_bits* value. The start value of all extended sequential values is the *ess_extend_begin* value. If the base sequential value reaches the maximum value, one value is added to the extended sequential value and the base sequential value increments from 0 again. • Base sequential value: The number of digits of all base sequential values is the *bits* value. The start value of all base sequential values is the *begin_number* value. The base sequential value is incremented based on the number of ECS instances to be created. If the base sequential value reaches the maximum value, one value is added to the extended sequential value and the base sequential value increments from 0 again.
• ess-node-()[]-ecshost • ess-node-(,)[,]-ecshost	N/A	ess-node-000000000-ecshost, ess-node-000000001-ecshost, and ess-node-000000002-ecshost.	• Extended sequential value: If *ess_extend_begin* is not specified, *ess_extend_begin* is set to *0*. If *ess_extend_bits* is not specified, *ess_extend_bits* is set to *3*. • Base sequential value: If *begin_number* is not specified, *begin_number* is set to *0*. If *bits* is not specified, *bits* is set to *6*.

Example	Existing hostname with the maximum sequential value in a scaling group	Hostname (three created ECS instances)	Description
ess-node-(0,1)[0,1]-ecshost	ess-node-08-ecshost	ess-node-10-ecshost, ess-node-11-ecshost, and ess-node-12-ecshost.	• When Auto Scaling creates instances in a scaling group not for the first time, the base sequential value increments starting from the maximum sequential value of existing hostnames in the scaling group. • When the sum of the maximum base sequential value of existing hostnames and the number of instances to be created in a scaling group is greater than or equal to the maximum base sequential value, the hostnames of instances in the scaling group may not be consecutively incremented. To ensure that the hostnames of ECS instances in the scaling group are consecutively incremented, we recommend that you set the number of digits of base sequential values to at least 3.
ess-node-(0,1)[0,1]-ecshost	ess-node-Z9-ecshost	An error is reported and the scale-out event is suspended.	• When Auto Scaling creates instances in a scaling group not for the first time, the base sequential value increments starting from the maximum sequential value of existing hostnames in the scaling group. • If Auto Scaling needs to create more ECS instances after the upper limits of extended sequential values and base sequential values are both reached, an error is reported and the scale-out event is suspended. In this case, you must set the rules for generating hostnames again.

Example	Existing hostname with the maximum sequential value in a scaling group	Hostname (three created ECS instances)	Description
ess-node-(0,1)[0,3]	ess-node-0099-ecshost	ess-node-0100, ess-node-0101, and ess-node-0103.	• When Auto Scaling creates instances in a scaling group not for the first time, the base sequential value increments starting from the maximum sequential value of existing hostnames in the scaling group. • By default, the system sequentially increments the values. If a created ECS instance in a scaling group cannot be started, Auto Scaling removes the instance from the scaling group and creates another instance. Therefore, the hostnames of ECS instances in the scaling group may not be consecutively incremented. In this example, the hostname of the ECS instance that fails to start is ess-node-0102.
ess-node-(0,1)[99,1]-ecshost	ess-node-0000099-ecshost	ess-node-0000100-ecshost, ess-node-0000101-ecshost, and ess-node-0000102-ecshost.	• When Auto Scaling creates instances in a scaling group not for the first time, the base sequential value increments starting from the maximum sequential value of existing hostnames in the scaling group. • By default, when the number of digits of the specified *begin_number* value is greater than the *bits* value, *bits* is set to *6*.

16.Configure a combination policy for removing instances

When you create a scaling group, you must specify an appropriate combination policy for removing Elastic Compute Service (ECS) instances. This topic describes how combination policies take effect and provides policy examples to help you choose a combination policy that matches your business scenarios.

Context

A combination policy for removing instances consists of the following policies. ECS instances are removed from a scaling group based on a combination policy.

- Scaling policy: adds ECS instances to or removes ECS instances from a scaling group based on zones or instance costs. When a scale-in event is triggered, Auto Scaling filters and removes ECS instances based on the scaling policy. Valid values for the Scaling Policy parameter:
  - **Priority Policy**: The first specified vSwitch has the highest priority. Auto Scaling preferentially attempts to scale ECS instances in the zone where the vSwitch with the highest priority resides. If the scaling fails, Auto Scaling attempts to scale instances in the zone where the vSwitch with the next highest priority resides.
  - **Balanced Distribution Policy**: This policy is valid when a scaling group is associated with multiple vSwitches that are distributed across more than two zones. After a scale-in or scale-out event is complete, Auto Scaling evenly distributes ECS instances across the zones where the vSwitches reside.
  - **Cost Optimization Policy**: This policy is valid when you specify multiple instance types in the active scaling configuration. When a scale-out event is triggered, Auto Scaling attempts to create ECS instances based on the unit prices of vCPUs in ascending order. When a scale-in event is triggered, Auto Scaling attempts to remove ECS instances based on the unit prices of vCPUs in descending order.
- Instance removing policy: filters ECS instances that meet the specified conditions in a scaling group based on the order of time. Valid values for the two steps of an instance removing policy:

> **Note** An instance removing policy consists of two steps:
>
> - If you configure only one step for the policy, Auto Scaling filters instances to find the ones that meet the requirements of the one step.
> - If you configure two steps for the policy, Auto Scaling filters instances to find the ones that meet the requirements of the first step and then filters instances in the filtering result to find the ones that meet the requirements of the second step. You cannot configure the same value for the two steps.

- **Earliest Instance Created Using Scaling Configuration**: filters instances to find the ones that were created based on the earliest scaling configuration and launch template. Manually added instances are not associated with scaling configurations or launch templates. Therefore, manually added instances are not filtered first. If all instances associated with all scaling configurations and launch templates have been removed but Auto Scaling must remove more instances from the scaling group, manually added instances are removed at random.

  > **Note** Scaling Configuration in **Earliest Instance Created Using Scaling Configuration** indicates the instance configuration source, which includes scaling configurations and launch templates.

  The version of a launch template does not indicate the order in which the template was added. For example, you select the lt-foress V2 template when you create a scaling group. Then, you select the lt-foress V1 template to modify the scaling group. The scaling group considers the lt-foress V2 launch template as the template that was added earlier.
- **Earliest Created Instance**: filters instances to find the instances that were created at the earliest point in time.
- **Most Recent Created Instance**: filters instances to find the most recently created instances.
- **No Policy**: This value is available only for **Then Remove from Results**. This value indicates that Auto Scaling does not filter instances based on the Then Remove from Results field.

> **Note**
> - For information about how to specify a combination policy, see Create a scaling group.
> - If multiple ECS instances in a scaling group meet the requirements of a combination policy, Auto Scaling randomly removes one of these ECS instances from the scaling group.
> - For information about how to avoid removing manually added ECS instances, see Put an ECS instance into the Protected state.

The following table describes the ECS instances in a scaling group. In the following examples, different types of scaling policies are used to demonstrate how Auto Scaling removes an ECS instance based on different combination policies.

> **Note** Data of each instance in the following table are for reference only. The actual data in the Auto Scaling console prevails.

Instance ID	Zone	Added at	Scaling configuration (asc-1 was added at the earliest point in time)	Unit price of vCPUs (USD)
i-1	Hangzhou Zone H	11:05, May 17, 2021	asc-1	1
i-2	Hangzhou Zone I	11:05, May 18, 2021	asc-1	2

Instance ID	Zone	Added at	Scaling configuration (asc-1 was added at the earliest point in time)	Unit price of vCPUs (USD)
i-3	Hangzhou Zone I	11:05, May 19, 2021	asc-1	3
i-4	Hangzhou Zone H	11:05, May 20, 2021	asc-2	3
i-5	Hangzhou Zone I	11:05, May 21, 2021	asc-2	3

Scenario 1: The priority policy is used as the scaling policy by default

Auto Scaling filters ECS instances to find the ones that meet the requirements of the instance removing policy. The priority policy does not affect the filtering result. The following table describes the final effect of the combination policy.

Valid value for the first step	Valid value for the second step	Description	ID of the removed instance
Earliest Instance Created Using Scaling Configuration	**Earliest Created Instance**	Filters instances to find the ones that were created based on the earliest scaling configuration and then removes the instances that were created at the earliest points in time from the filtering result.	i-1
	Most Recent Created Instance	Filters instances to find the ones that were created based on the earliest scaling configuration and then removes the most recently created instances from the filtering result.	i-3
	No Policy	Filters instances to find the ones that were created based on the earliest scaling configuration and then randomly removes an instance from the filtering result.	• i-1 • i-2 • i-3
Earliest Created Instance	**Most Recent Created Instance**	Removes the instances that were created at the earliest points in time.	i-1
	Earliest Instance Created Using Scaling Configuration	Filters instances to find the ones that were created at the earliest points in time and then removes the instances that were created based on the earliest scaling configuration from the filtering result.	i-1

Valid value for the first step	Valid value for the second step	Description	ID of the removed instance
	No Policy	Removes the instances that were created at the earliest points in time.	i-1
Most Recent Created Instance	**Earliest Created Instance**	Removes the most recently created instances.	i-5
	Earliest Instance Created Using Scaling Configuration	Filters instances to find the most recently created ones and then removes the instances that were created based on the earliest scaling configuration from the filtering result.	i-5
	No Policy	Removes the most recently created instances.	i-5

Scenario 2: The scaling policy is a balanced distribution policy

Auto Scaling finds the zones where the ECS instances reside based on the balanced distribution policy and then filters and removes the instances that meet the requirements of the instance removing policy. This ensures that the existing instances in the scaling group are evenly distributed across the zones after instances are removed from the scaling group.

In this example, Auto Scaling filters instances to find the i-2, i-3, and i-5 instances in Hangzhou Zone I based on the balanced distribution policy. This is because Hangzhou Zone I has one more instance than Hangzhou Zone H. The following table describes the final effect of the combination policy.

Valid value for the first step	Valid value for the second step	Description	ID of the removed instance
Earliest Instance Created Using Scaling Configuration	**Earliest Created Instance**	Filters instances to find the ones that were created based on the earliest scaling configuration and then removes the instances that were created at the earliest points in time from the filtering result.	i-2
	Most Recent Created Instance	Filters instances to find the ones that were created based on the earliest scaling configuration and then removes the most recently created instances from the filtering result.	i-3
	No Policy	Filters instances to find the ones that were created based on the earliest scaling configuration and then randomly removes an instance from the filtering result.	• i-2 • i-3
	Most Recent Created Instance	Removes the instances that were created at the earliest points in time.	i-2

Valid value for the first step	Valid value for the second step	Description	ID of the removed instance
Earliest Created Instance	**Earliest Instance Created Using Scaling Configuration**	Filters instances to find the ones that were created at the earliest points in time and then removes the instances that were created based on the earliest scaling configuration from the filtering result.	i-2
	No Policy	Removes the instances that were created at the earliest points in time.	i-2
Most Recent Created Instance	**Earliest Created Instance**	Removes the most recently created instances.	i-5
	Earliest Instance Created Using Scaling Configuration	Filters instances to find the most recently created ones and then removes the instances that were created based on the earliest scaling configuration from the filtering result.	i-5
	No Policy	Removes the most recently created instances.	i-5

Scenario 3: The scaling policy is a cost optimization policy

Auto Scaling filters and removes the ECS instances that have the highest vCPU unit price based on the cost optimization policy. If multiple instances that have the highest vCPU unit price exist in the scaling group, Auto Scaling filters and removes instances based on the instance removal policy.

In this example, Auto Scaling filter instances to find the i-3, i-4, i-5 instances based on the cost optimization policy. This is because these instances have the same highest unit price of vCPU, which is USD 3. The following table describes the final effect of the combination policy.

Valid value for the first step	Valid value for the second step	Description	ID of the removed instance
Earliest Instance Created Using Scaling Configuration	**Earliest Created Instance**	Filters instances to find the ones that were created based on the earliest scaling configuration and then removes the instances that were created at the earliest points in time from the filtering result.	i-3
	Most Recent Created Instance	Filters instances to find the ones that were created based on the earliest scaling configuration and then removes the most recently created instances from the filtering result.	i-3

Valid value for the first step	Valid value for the second step	Description	ID of the removed instance
	No Policy	Filters instances to find the ones that were created based on the earliest scaling configuration and then randomly removes an instance from the filtering result.	i-3
Earliest Created Instance	**Most Recent Created Instance**	Removes the instances that were created at the earliest points in time.	i-3
	Earliest Instance Created Using Scaling Configuration	Filters instances to find the ones that were created at the earliest points in time and then removes the instances that were created based on the earliest scaling configuration from the filtering result.	i-3
	No Policy	Removes the instances that were created at the earliest points in time.	i-3
Most Recent Created Instance	**Earliest Created Instance**	Removes the most recently created instances.	i-5
	Earliest Instance Created Using Scaling Configuration	Filters instances to find the most recently created ones and then removes the instances that were created based on the earliest scaling configuration from the filtering result.	i-5
	No Policy	Removes the most recently created instances.	i-5