

Enterprise Distributed Application Service (EDAS)

Console User Guide

Console User Guide

Resources

ECS

Create ECS instances

If you do not have any ECS instances, follow the instructions in this document to create an ECS instance.

Prerequisites

Before creating an ECS instance, determine the network type of your ECS instance:

Alibaba Cloud stopped providing ECS instances in classic networks to new ECS instance users at 17:00 (UTC +8:00) on June 14, 2017. For first-time purchasing of ECS instances, you can choose a VPC only. If you have already purchased ECS instances in classic networks, you can still select them.

To create an ECS instance in a VPC, you must select a specific VPC. The ECS instance created in the selected VPC cannot be moved to another VPC. Therefore, make sure you select the correct VPC before creating the ECS instance. If you do not have any VPCs, create a VPC first.

Procedure

Log on to the EDAS console.

In the left-side navigation pane, choose **Resource Management** > **ECS**.

On the **ECS** page, select **Region** and **Namespace** (optional) for the ECS instance to create. Then, click **Create Instance** in the upper-right corner of the page.

On the ECS purchase page, select ECS type and configuration and pay for it. See [Create ECS instances](#) for details.

Note: To use and manage an ECS instance in the EDAS console, you must install the EDAS Agent on the ECS instance. When purchasing an ECS instance, you can choose the EDAS base image to automatically install the EDAS Agent. The procedure is as follows:

In the **Image** section, select **Marketplace Images**. Then, click **Select from image market (including operating system)**.

Enter **EDAS** in the search box and click **Search**.

In the search results, choose **EDAS Java Environment (common ECS instance)** and the version (the latest version is selected by default). Then, click **Apply**.

Result verification

Click **Management Console** to return to the **ECS Console**. The ECS instance creation process usually takes one to five minutes. Click the Refresh button, and the ECS instance status will change to **Running**, indicating that the ECS instance has been created successfully.

Import ECS instances

After purchasing an ECS instance, you need to import it in order to install EDAS Agent and synchronize it to EDAS.

If you did not select any EDAS base image when purchasing an ECS instance, you can click **Import ECS** in the EDAS console to install EDAS Agent. The process of importing ECS is divided into direct import and import after conversion.

An ECS instance cannot be directly imported in any of the following conditions:

- The ECS instance was created before December 1, 2017.
- A classic network ECS instance is imported into a classic network cluster.
- The ECS instance is not running (it is stopped, starting, or stopping).
- The ECS instance is a Windows instance or does not support simple shell commands.
- The ECS instance is not imported from an ECS cluster.

Note: The ECS instance is formatted if image conversion is required. If you do not want the disks of the ECS instance to be formatted, click **Use command script for manual installation** on the conversion page.

Log on to the EDAS console. In the left-side navigation pane, choose **Resource Management > ECS**.

On the **ECS** page, select **Region** and **Namespace** (optional). Then, click **Import ECS** in the upper-right corner of the page.

On the **Select Cluster and ECS** page, select **Namespace** and **Select Cluster to Import**. Then, select an ECS instance and click **Next**.

Note:

- If the desired cluster is not in the list, click **Create Cluster** to the right of **Select Cluster to Import** to create a cluster.
- To manually install EDAS Agent, select **Switch to Manual Installation** in the upper-right corner to go to the **Install EDAS Agent on Single Instance Manually** page, and use the script to manually install EDAS Agent.

On the **Ready to Import** Page, view the information of the selected instance.

- If the ECS instance can be imported directly, click **Confirm and Import**. If the ECS instance needs to be imported after conversion, select **I agree to convert the above instances, and fully understand that the data in the original systems will be lost after conversion** and enter the new password for root user logon after conversion. Click **Confirm and Import**.

On the **Import** tab page, view the import progress of the instance.

- If the ECS instance supports direct import, you can view its import progress on the **Import** tab page. If the message **Instance transfer succeeded** is displayed, the ECS instance has been imported successfully. Click **Click to return to the Cluster Details page**. When the ECS instance status changes to **Running**, the ECS instance is imported to the cluster successfully.
- For an ECS instance that needs to be converted before being imported, the import progress of the ECS instance displayed on the **Import** tab page is **Converting now**. **This might take 5 minutes**. If you click **Click to return to the Cluster Details page**

before the import is completed, the health check status **Converting** and the conversion progress in percentage are displayed. When the import is completed, the health check status **Running** is displayed, indicating that the ECS instance has been successfully imported.

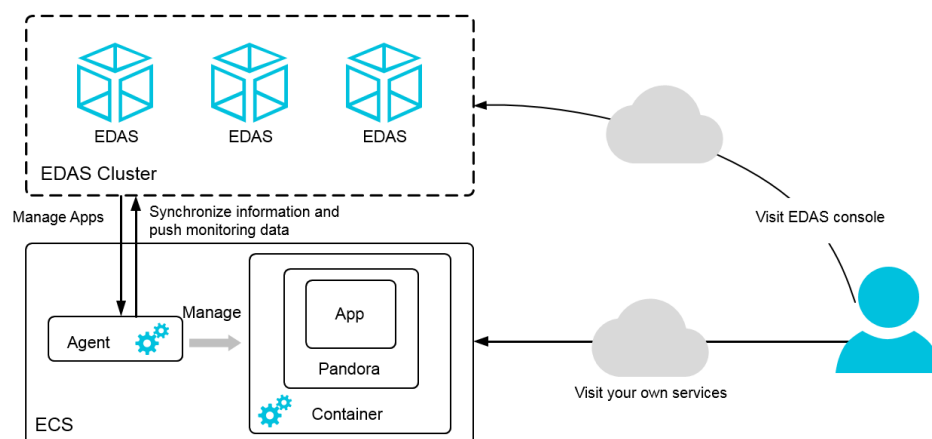
Install EDAS Agent

EDAS Agent is the daemon program installed on ECS instances that communicates between the EDAS cluster and the applications deployed on the corresponding ECS instances. EDAS Agent provides the following functions:

- Application management: deploys, starts, and stops applications.
- Status reporting: reports application viability status, health check results, and Ali-Tomcat container status.
- Information retrieval: retrieves the monitoring information on ECS instances and containers.

In addition to this application-based management, EDAS Agent is also used to communicate between the EDAS console and your applications. For example, EDAS Agent must be used to determine whether an ECS instance has correctly and promptly published a service that an application publishes.

Note: You can transparently access these functions by installing EDAS Agent.



Install EDAS Agent

EDAS deploys applications (including first-time installation and later on expansion) on ECS instances that are installed with EDAS Agent only. The application instances in the EDAS billing system refer to the ECS instances which are installed with EDAS Agent and deployed with applications. To use EDAS, you must install EDAS Agent after you purchase an ECS instance.

There are three ways to install EDAS Agent:

- Use an EDAS base image
- Import an ECS instance
- Manually execute the script for installation

Note:

- JDK 8 is installed on EDAS Agent by default in these three scenarios. To use JDK 7 or another version, run the corresponding installation script.
- This script requires you to log on to your ECS instance as the root user. This script can be executed repeatedly, which overwrites the existing version of EDAS Agent installed on the ECS instance. Therefore, you can upgrade EDAS Agent by running the same script.
- The script for installation is region-specific. You must switch to the appropriate region before clicking **Install EDAS Agent**.
- Different installation methods or different images or clusters selected during installation result in different EDAS Agent statuses. This determines the type of applications that you can create on the ECS instance. When using an installation method, follow the related instructions.

Use an EDAS base image to automatically install EDAS Agent when purchasing an ECS instance

The easiest method of installation of EDAS Agent is to use the EDAS base image when purchasing an ECS instance.

Note: This method requires disk formatting. To avoid formatting disks, we recommend that you use the command script to manually install EDAS Agent.

Log on to the EDAS console. In the left-side navigation pane, choose **Resource Management > ECS**.

In the upper-right corner of the **ECS** page, click **Create Instance**.

On the purchase page, select **Image Market** in the **Image** section, and then click **Select from image market (including operating system)**.

Enter **EDAS** in the search box and click **Search**.

Select an image from the search results based on the requirements of your applications. By default, the latest version of image is selected (we recommend that you do not select an earlier version). Then, click **Use**.

To create a **Common Application**, select **EDAS Java Environment (Common ECS)**.

To create a **Docker application**, select **EDAS**.

Follow the instructions on the page to purchase an ECS instance.

Import an ECS instance to automatically install EDAS Agent

If you did not select any EDAS base images when purchasing an ECS instance, you can use the **Import ECS** function in the EDAS console to install EDAS Agent. An ECS instance can be imported either directly or after image conversion.

Under any of the following conditions, direct import of an ECS instance is not possible:

- The ECS instance was created before December 1, 2017.
- A classic network ECS instance is imported into a classic network cluster.
- The ECS instance is not running (it is stopped, starting, or stopping).
- The ECS instance is a Windows instance or does not support simple shell commands.
- The ECS instance is not imported from an ECS cluster.

Note: If image conversion is required, disk formatting will be performed. To avoid formatting disks, we recommend that you manually execute the script to install EDAS Agent.

In the left-side navigation pane of the EDAS console, choose **Resource Management > ECS**.

On the **ECS** page, select **Region** and **Namespace**. Then, click **Import ECS** in the upper right corner of the page.

On the **Select Cluster and ECS** page, enter **Namespace** and **Select Cluster to Import**. In the ECS instance list, select the ECS instance that needs to be imported, and click **Next**.

Note:

- If the desired cluster is not in the list, click **Create Cluster** to the right of **Select Cluster to Import** to create a cluster.
- To manually install EDAS Agent, click **Switch to Manual Installation** in the upper-right corner to go to the **Install EDAS Agent on Single Instance Manually** page and use the command script to manually install EDAS Agent.

On the **Ready to Import** Page, view the information of the selected instance.

- If the ECS instance can be imported directly, click **Confirm and Import**. If the ECS instance needs to be converted before import, select **I agree to convert the above instances, and fully understand that the data in the original systems will be lost**

after conversion and enter the new password for root user logon after conversion. Click **Confirm and Import**.

On the **Import** tab page, view the import progress of the instance.

- If the ECS instance supports direct import, you can view its import progress on the **Import** tab page. If the message **Instance transfer succeeded** is displayed, the ECS instance has been imported successfully. Click **Click to return to the Cluster Details page**. If the status of the ECS instance is **Running**, the ECS instance has been imported to the cluster and EDAS Agent has been successfully installed.
- For an ECS instance that needs to be converted before being imported, the import progress of the ECS instance displayed on the **Import** page is **Converting now. This may take 5 minutes**. If you click **Click to return to the Cluster Details page** before the import is completed, the health check status **Converting** and the conversion progress in percentage are displayed. When the import is completed, the health check status **Running** is displayed, indicating that the ECS instance has been imported to the cluster and EDAS Agent has been installed successfully.

Use the command script to manually install EDAS Agent

Note: This approach applies to ECS instances in the ECS cluster only.

1. In the left-side navigation pane, choose **Resource Management > ECS**.

On the **ECS** page, select **Region** and **Namespace**. Then, click **Import ECS** in the upper right corner of the page.

Click **Switch to Manual Installation** in the upper-right corner of the **Import ECS** page. On the **Install EDAS Agent on Single Instance Manually** page, click **Click to Copy**.

Note: To install EDAS Agent with an image, click **Switch to Image Installation** in the upper-right corner of the page and then import an ECS instance.

Log on to the ECS instance where the EDAS Agent is to be installed as the root.

On the ECS instance, paste the copied command and execute it.

Result verification

After installing EDAS Agent, choose **Resource Management > ECS** from the left-side navigation pane of the EDAS console. On the **ECS** page, choose the appropriate region to view the **Agent Status**.

If the EDAS Agent installation is successful, the Agent status is **Online** (for ECS clusters) or

Docker Online (for Swarm or Kubernetes clusters).

If the EDAS Agent installation failed, the Agent status is **Unknown**.

Upgrade EDAS Agent

The procedure for upgrading EDAS Agent is similar to that for installing EDAS Agent with a command script. For details, see [Use the command script to manually install EDAS Agent](#).

SLB

If you have purchased Alibaba Cloud Server Load Balancer, EDAS synchronizes the SLB instances to the EDAS console, allowing you to use and configure load balancing functions.

View SLB instances

Log on to the EDAS console.

In the left-side navigation pane, choose **Resource Management** > **SLB**.

On the **SLB** page, select **Region** to view information about SLB instances in the region.

Description of SLB instances:

- **SLB ID/Name:** The ID is automatically generated by the system while the instance name is user-defined. Click the SLB ID to go to the SLB console.
- **Service Address:** indicates the Internet or intranet IP address of the SLB instance.
- **Backend Server:** indicates the ECS instance added in the SLB console that is used to receive the requests distributed by the SLB instance.
- **Status:** indicates the status of the SLB server, which can be running or stopped. Expired SLB instances are not displayed.

Note: To create an SLB instance, click **Create Instances** in the upper-right corner of the page. Then, go to the SLB purchase page on the Alibaba Cloud official website to purchase and create an SLB instance. For more information, see the [SLB documentation](#).

Configure SLB

For load balancing configuration information, see [Create SLB instances](#) and [Configure SLB instances](#).

VPC

Alibaba Cloud provides two network types:

Classic networks

Cloud products on a classic network are all deployed in Alibaba Cloud's public infrastructure and planned and managed by Alibaba Cloud. These products are suitable for customers who have high ease-of-use requirements.

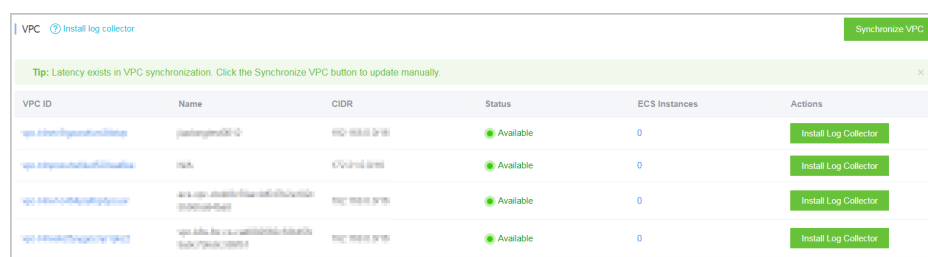
VPC networks

VPC networks are virtual private clouds that allow custom isolation settings. You can define the custom VPC topology and IP addresses. VPCs are suitable for customers with high cybersecurity requirements and network management capabilities.

After purchasing a VPC and synchronizing it to the EDAS console, you can view its information in the EDAS console.

Log on to the EDAS console.

In the left-side navigation pane, choose **Resource Management** > **VPC**.



VPC ID	Name	CIDR	Status	ECS Instances	Actions
vpc-12345678901234567890	test-vpc-1	192.168.0.0/24	Available	0	Install Log Collector
vpc-23456789012345678901	test-vpc-2	192.168.0.0/24	Available	0	Install Log Collector
vpc-34567890123456789012	test-vpc-3	192.168.0.0/24	Available	0	Install Log Collector
vpc-45678901234567890123	test-vpc-4	192.168.0.0/24	Available	0	Install Log Collector

On the **VPC** page, select **Region** to view information about VPC instances in the region.

Description of VPC instances:

- **VPC ID:** automatically generated by the system when the VPC is created. Click the

VPC ID to go to the VPC console.

- **Name:** indicates the VPC name that you specified during VPC creation.
- **CIDR Block:** indicates the VPC's CIDR block you specified when the VPC is created.
- **Status:** indicates the status of the VPC instance: Running or Stopped. Expired VPC instances are no longer displayed.
- **ECS instances:** indicates the number of ECS instances created in this VPC network. Click the number to go to the ECS page, where you can see all the ECS instances in this VPC.

In a VPC network, ECS instances are isolated from EDAS instances. Therefore, you must install the log collector to collect the information on ECS instances. Click **Install Log Collector** in the Actions column on the Instances page. For log collector installation instructions, see [Install log collector](#).

Resource groups

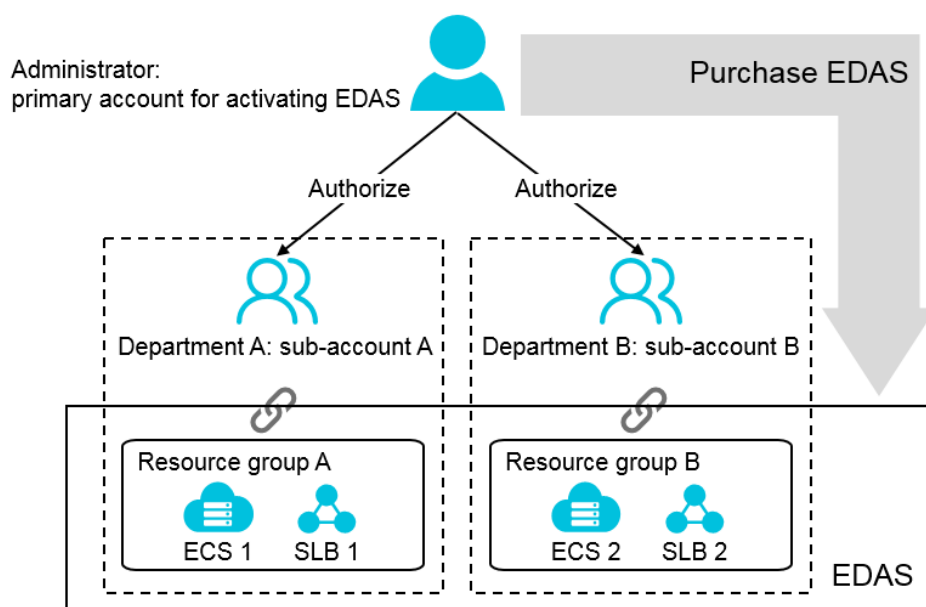
Overview

Resource groups are groups of EDAS resources, which can be ECS instances, SLB instances or clusters, but not VPCs. You can use a primary account to purchase and group resources and assign appropriate permissions for the resource groups to sub-accounts for application O&M. You can assign permissions to access resource groups to your sub-accounts. Each sub-account has the right to operate on all the resources in the specified group. For more information about primary accounts and sub-accounts, see [EDAS account system](#).

Typical scenarios

- A company uses EDAS to create business applications. Department A is responsible for user-related applications and Department B for goods-related ones.
- The company registers for an EDAS account (the primary account) to activate EDAS and creates two sub-accounts for Departments A and B. Department A has a several ECS instances and SLB instances for deploying user-related applications.
- The company creates a resource group in EDAS, adds Department A's ECS instances and SLB instances to the resource group, and grants permissions for this resource group to Department A's sub-account.
- Department A can use its sub-account to operate on the resources in this resource group only. The resources for Department A and Department B are not conflicted.

This is shown in the following figure:



Resource group operations

Resource group operations are performed in the EDAS console. Follow the procedure discussed below to go to the Resource Groups page.

Log on to the EDAS console.

In the left-side navigation pane, choose **Resource Management > Resource Groups**.

On the **Resource Groups** page, select a region to view the resource groups in the region and the ECS and SLB instances in each group.

View resource groups

On the **Resource Groups** page, you can view information about resource groups, including resource group names, descriptions, ECS instances (intranet/Internet IP addresses), and SLB instances, and clusters.

Create a resource group

1. In the upper-right corner of the **Resource Groups** page, click **Create Resource Group**.
2. In the **Create Resource Group** dialog box, enter **Resource Group Name** and **Resource Group Description** and click **OK**.

Add ECS instances to the resource group

1. In the resource group list, locate the row that contains the target resource group, and click **Bind ECS** in the Actions column.
2. In the **Bind ECS** dialog box, select the target ECS instance and click **OK**.

Add an SLB instance to the resource group

1. In the resource group list, locate the row that contains the target resource group, and click **Bind SLB** in the Actions column.
2. In the **Bind SLB** dialog box, select the target SLB instance and click **OK**.

Edit a resource group

1. In the resource group list, locate the row that contains the target resource group, and click **Edit** in the Actions column.
2. In the **Edit Resource Group** dialog box, edit the resource group name and description and click **OK**.

Grant resource group permissions to a sub-account

1. Log on to the EDAS console using the primary account.
2. In the left-side navigation pane, choose **Account Management > Sub-Accounts**.
3. Locate the row that contains the target account and click the **Resource Group Permission** button in the Actions column.
4. In the **Resource Group Permission** dialog box, select a resource group and click **OK**.

Delete a resource group

1. In the resource group list, locate the row that contains the target resource group, and click **Delete** in the Actions column.
2. In the **Delete Resource Group** dialog box, click **OK** to confirm the deletion.

Clusters

Create ECS clusters

To create an ECS cluster for an application, you need to complete the following steps:

1. [Create an ECS cluster]#CreateEmptyECSCluster)
2. Add ECS instances

When no application is deployed on an ECS instance in a cluster and the ECS instance is no longer needed, you can remove to release the ECS instance.

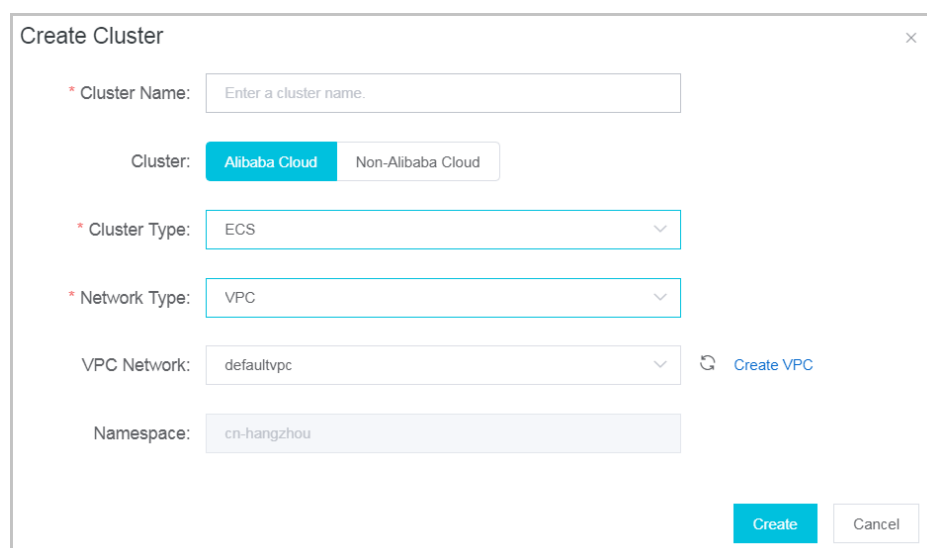
Create clusters

Log on to the EDAS console.

In the left-side navigation pane, choose **Resource Management** > **Clusters**.

On the **Clusters** page, select **Region** and **Namespace**. Then, click **Create Instance** on the right of the page.

In the **Create Cluster** dialog box, set the cluster parameters and click **Create**.



Description of cluster parameters:

- **Cluster Name:** Enter the cluster name. The name can only contain letters, digits, underscores (_), and periods (.), with a length up to 64 characters.
- **Cluster type:** Select **ECS**.
- **Network Type:** Select **Classic Network** or **VPC**. Select a network type from the drop-down list as needed. **If you select the VPC Network, ensure that you have created a VPC instance first.**
- **VPC Network:** Select a VPC instance you created from the drop-down list.
- **Namespace:** indicates the namespace displayed on the Clusters page, which cannot be modified. If you do not make a selection, the name of the displayed region is

selected by default.

After creating the cluster, you will see the message **Created successfully** in the upper-right corner of the page, and the cluster is displayed in the cluster list.

Note: The created cluster is empty. Only after you add an ECS instance can you meet the requirements of applications.

Add ECS instances

There are currently two ways of adding ECS instances to a cluster in EDAS:

- Direct import: Image conversion is not required.
- Import after conversion: The EDAS official image is used to reinstall the system. After re-installation, all data in the ECS instance is deleted and the ECS instance login password needs to be reset.

An ECS instance cannot be directly imported in any of the following conditions:

- The ECS instance was created before December 1, 2017.
- A classic network ECS instance is imported into a classic network cluster. The ECS instance is not running (it is stopped, starting, or stopping).
- The ECS instance is a Windows instance or does not support simple shell commands.

Direct import

On the **Clusters** page, click the name of the cluster you created.

On the **Cluster Details** page, click **Add ECS Instance** in the upper-right corner.

On the **Select Cluster and ECS** page, select an import method and ECS instance in the instance list, and then click **Next**.

- **Import ECS:** You cannot modify the namespace and the imported cluster. Additionally, ECS instances are imported from the default namespace and cluster in this region.
- **From Existing Cluster:** A namespace and a cluster are selected from the region. Then, the ECS instance is moved from the left pane to the right pane on the page.

If no instances meet the necessary conditions, click **Create ECS Instance** in the upper-right corner of the page. This takes you to the ECS purchase page on the Alibaba Cloud official website, where you can purchase and create a new ECS instance. For more information, see

Create ECS instances.

! [Add ECS instances].(http://docs.aliyun.cn-hangzhou.oss.aliyun-inc.com/assets/pic/57453/cn_zh/1546428404616/edas-resourcegmt-cluster-addECS-selectClusterandECS.png)

On the **Ready to Import** page, view the selected ECS instance information, and click **Confirm and Import**.

On the **Import** page, view the import progress of the ECS instance.

When **Direct import successful** is displayed, the ECS instance has been successfully added to the cluster.

Click **Click to return to the Cluster Details page**. After the ECS instance is imported, the **Health Check** status of the ECS instance is **Running**.

Import after conversion

You need to reset the password during import. An ECS instance may be reinstalled and the original data may be deleted in the process.

See steps 1 to 3 of [Direct import] to select an ECS instance.

On the **Ready to import** page, check the information about the selected ECS instance and select **I agree to convert the above instances, and fully understand that the data in the original systems will be lost after conversion**. Enter the new password of the root user for logging on to the system after conversion, and click **Confirm and Import**.

On the **Import** page, view the import progress of the ECS instance.

When you begin the import, the **Status** of the ECS instance is **Converting now. This might take 5 minutes**.

Click **Click to return to the Cluster Details page**. The **Health Check** status of the ECS instance is **Converting** and the progress percentage is displayed.

When the ECS instance is imported, the **Health Check** status is **Running**.

Remove an ECS instance

In the **ECS Instances and Applications** area of the **Cluster Details** page, locate the row that contains the target ECS instance and click **Remove** in the Actions column.

In the **Remove an ECS instance** dialog box, confirm the instance information and click **Remove**.

In the removal process, the **Health Check** status of the ECS instance is **Deleting** and the progress percentage is displayed.

After the ECS instance is deleted, it is no longer displayed in the instance list.

Create Container Service Kubernetes

Container Service for Kubernetes provides enterprise-level high-performance and flexible management of Kubernetes containerized applications throughout the application lifecycle. This service simplifies cluster creation and expansion and integrates Alibaba Cloud capabilities in virtualization, storage, network, and security, providing an improved running environment for Kubernetes containerized applications.

To create a Container Service Kubernetes cluster for an application, you need to complete the following steps:

1. Create Container Service Kubernetes clusters in the Container Service for Kubernetes console
2. Import Container Service Kubernetes clusters into the EDAS console

Prerequisites

- You have activated EDAS.
- You have activated Container Service for Kubernetes and completed Role authorization.

Note: We recommend that you use the same account for Container Service and EDAS. If you use different accounts, make sure they belong to the same primary account and have been authorized.

Create Container Service Kubernetes clusters in the Container Service for Kubernetes console

1. Log on to the Container Service console.

Create a Container Service Kubernetes cluster.

In Container Service, you can create **Kubernetes**, **Kubernetes hosting**, and **Multi-Zone Kubernetes** clusters. Select any type, as needed.

- **Create Kubernetes clusters:** Three of the instances that you purchase and add must serve as master nodes. Applications cannot be deployed on these three instances. You can only deploy applications on the other instances (workers).
- **Create Kubernetes hosting clusters:** All the instances that you purchase and add must serve as worker nodes. Applications can be deployed on these instances.
- **Create multi-zone Kubernetes clusters:** In contrast to Kubernetes clusters, this type of cluster has nodes distributed in different zones. If one zone is unavailable, the nodes can work in other zones. Three of the instances that you purchase and add must serve as master nodes. Applications cannot be deployed on these three instances, but can only be deployed on the worker instances that you purchase separately.

Import Container Service Kubernetes clusters into the EDAS console

Log on to the EDAS consoleEDAS console..

In the left-side navigation pane, choose **Resource Management** > **Clusters**.

On the **Clusters** page, click **Container Service K8S Cluster**. In the cluster list, locate the row that contains the Container Service Kubernetes cluster you created and click **Import** in the **Actions** column. In the **Import Kubernetes Cluster** dialog box, click **Import**.

When the option in the **Actions** column changes to **Delete** and the cluster status is **Running**, the Container Service Kubernetes cluster has imported successfully.

Manage clusters

View the cluster list

Log on to the EDAS console.

In the left-side navigation pane, choose **Resource Management** > **Clusters**.

On the **Clusters** page, select **Region** to view information about clusters in the region.

Note:

- There are two types of clusters: **EDAS Cluster** and **Container Service K8S Cluster**. **EDAS Cluster** includes **ECS Cluster** and **Swarm Cluster**.
- Two network types, namely, **VPC** and **Classic Network**, are available.

View cluster details

On the **Clusters** page, click the name of the target cluster.

On the **Cluster Details** page, view the cluster details.

The **Cluster Details** page mainly consists of Cluster Information and ECS Instances and Applications.

Cluster Information: displays basic information about the cluster.

ECS Instances and Applications: displays the list of ECS instances in the cluster, information about specific ECS instances, and applications deployed.

Deployed Applications: displays the applications deployed on the ECS instance. Click the name of the target application to go to the Application Details page.

Click a specific button in the **Actions** column to perform relevant operations on the ECS instance.

View Details: View details of the instance.

- **Event** (applies to Swarm clusters only): View the events that occurred to the ECS instance in the cluster. Event information helps you locate problem causes.
- **Remove:** Remove the ECS instance. If any application is deployed on the ECS instance, the Remove button is unavailable.

Transfer ECS instances (for ECS clusters)

In the **ECS Instances and Applications** area on the **Cluster Details** page, select an ECS instance and then click **Transfer ECS Instances** on the right side.

On the **Select Target Cluster** page, select **Namespace** and **Target Cluster**, and click **Next**.

On the **Ready to Import** page, click **Confirm and Import**.

- If the ECS instance cannot be imported directly, select **I agree to convert the above instances, and fully understand that the data in the original systems will be lost after conversion**. Reset the password of the root user for logging on to the ECS instance.

Application management

Namespaces

Overview

You can use namespace to isolate resources and services.

For example, you have three environments that are used separately used for development, testing and production. You can create three namespaces such as *Dev* , *Test* and *Prod* for the three environments. Then you can create clusters and deploy applications in these namespaces. The resources like ECS instances, applications and services are isolated with each other. The services cannot be invoked between different namespaces. Configurations can be pushed in only one namespace.

Create namespaces

Log on to the EDAS console.

In the left-side navigation pane, choose **Application Management** > **Namespace**.

On the **Namespace** page, select **Region** and click **Create Namespace** in the upper-right

corner.

In the **Create Namespace** dialog box, set **Namespace Name** (required), **Namespace ID** (required) and **Description**, and then click **Create**.

Note: The prefix of the namespace ID is determined by the specified region (target region) and cannot be modified. Rather, only the custom part can be modified.

Edit namespaces

On the Namespaces page, locate the row that contains the target namespace and click **Edit** in the Actions column.

In the **Edit Namespace** dialog box, edit the **Namespace Name** and description. Then, click **OK**.

Note: You cannot change the namespace ID and the namespace type.

Delete namespaces

The following conditions must be met before you can delete namespaces:

- No ECS instances exist in the namespace.
- No clusters exist in the namespace.

On the Namespaces page, locate the row that contains the target namespace and click **Delete** in the Actions column.

In the **Delete Namespace** dialog box, confirm the namespace to be deleted and click **Delete**.

Applications in ECS Cluster

Deploy applications in ECS clusters

In the ECS cluster, you can deploy applications through **WAR** packages or **JAR** packages, with similar parameters and operations.

You can deploy applications in the following scenarios:

- If you have deployed an application after creating it, you can upgrade its version through redeployment.
- If you just create an empty application, you can deploy it by following the instructions provided in this document.

After change process for deploying an application is completed, the application has been started without clicking **Start Application**. The **Start Application** button is unavailable until the application is stopped.

You can use the EDAS console to deploy applications in an ECS cluster.

Configure ECS applications

This topic describes how to set the **JVM** and **Tomcat** parameters as well as **Basic Information** for applications in the ECS cluster.

Note: the JVM and Tomcat parameters are application parameters. That is, they are valid for the entire application. If JVM and Tomcat parameters are set by group on the **Instance Information** page, the priority of the parameters is higher than application-level settings.

Log on to the EDAS console.

In the left-side navigation pane, choose **Application Management** > **Applications**.

On the **Applications** page, click the name of the target application.

On the **Basic Information** page, click **Settings** in the **Application Settings** area.

Set the JVM parameters

JVM parameters are container parameters that must be configured when an application is started. Correct configuration of JVM parameters helps reduce the overhead of garbage collection (GC) and thus shorten the server response time and improve throughput. If container parameters are

not set, JVM parameters are allocated by default.

In the **Application Settings** dialog box, click the **JVM** tab.

Click **Memory Configuration**, **Application**, **GC Policy**, **Tool**, and **Custom** to set relevant parameters. Click **Save** to save the settings.

Note: Manually restart the application to make the JVM parameters take effect.

Set the Tomcat parameters

You can configure the following parameters for the Tomcat container: port number, application access path, and maximum number of threads.

In the Application Settings dialog box, click the **Tomcat** tab.

Set the Tomcat parameters and click **Configure Tomcat**.

Description of Tomcat parameters:

Configuration	Description
Application Port	The parameter value ranges from 1024 to 65535. The admin authority is used by container configuration and the root authority is required to operate ports with numbers less than 1024. Therefore, enter a port number greater than 1024. If this parameter is not set, the default value 8080 is used.
Tomcat Context	Select an application access path: - If you select the Package Name, you do not have to set the Custom Path parameter. The application access path is the name of the WAR package. - If you select the Root Directory, you do not have to set the Custom Path parameter. The application access path is /. - If you select Custom, you must set the Custom Path parameter. If the "Custom Path" parameter is not set, the default application access path is the same as the

	name of the WAR package.
Maximum Threads	Set the number of connections to the connection pool. The corresponding parameter is maxThreads, which is 400 by default. This parameter has significant implications for performance. We recommend that this parameter be set under professional guidance.
Tomcat Encoding	Select a Tomcat encoding format: UTF-8, ISO-8859-1, GBK, or GB2312. The default format is ISO-8859-1.

Set basic information

In the Application Settings dialog box, click the **Basic Information** tab, set the **Application Name** and **Application Description**, and then click **Modify**.

Stop and start applications

After an application is released, you can stop and start the application on the Enterprise Distributed Application Service (EDAS) console.

Stop the application

Log on to the EDAS console.

Click **Applications** in the left-side navigation pane, and click an application on the Applications page to go to the **Basic Information** page.

When you click **Stop**, the application is stopped.

Start the application

Common applications and Docker applications are released upon deployment. You do not need to start them. Therefore, you only need to restart it after it is stopped.

Log on to the EDAS console.

Click **Applications** in the left-side navigation pane, and click an application on the Applications page to go to the **Basic Information** page.

When you click **Start**, the application is restarted.

Scale out and scale in applications

Application scale-out or scale-in increases or decreases the computational capacity of an application by changing the number of ECS instances. You can add a new ECS instance when the ECS instance where an application resides is overloaded, or can remove the ECS instance when it is no longer required. You can also use **Auto Scaling** to dynamically adjust the number of ECS instances.

Scale out

When the ECS instance for hosting an application is overloaded, you can manually scale the application out.

Note: The running status of the ECS instance added depends on the running status of the application.

- If the application is running during scale-out, the ECS instance automatically deploys, starts, and runs the application.
- If the application is stopped during scale-out, the ECS instance automatically deploys but does not start or run the application.

Log on to the EDAS console.

In the left-side navigation pane, choose **Application Management > Applications**. On the **Applications** page, click the name of the target application.

On the Application Details page, click **Scale Out** in the upper-right corner.

In the **Scale-Out Method** dialog box, select **Target Group** for scale-out.

Select **Scale-Out Method**, and complete the subsequent scale-out steps.

Select from Current Cluster

Idle ECS instances in the cluster are added to the application for scale-out.

Select an ECS instance, and then click **Scale-Out**.

The message **Scale-out successful** is displayed on the page.

Go to the **Instance Information** page of the application to view the running status of the added ECS instance.

If **Normal** is displayed, the scale-out operation is successful.

Purchases based on current ECS specifications

This method allows you to select any ECS instance in the cluster as the specification template. Then, new purchases are configured according to the specifications of the template.

Select the ECS instance you want to use as the specification template, and click **Next**.

On the **Purchase Details** page, enter **Purchase Quantity** and **Login Key**, select **ECS Service Terms | Image Service Terms**, and then click **Next**.

On the **Confirm** page, view the information about the purchased ECS instance, and click **Confirm**.

On top of the page, the message **Automatic purchasing is triggered. Check the real-time information in the application change process.** is displayed after your submission.

Go to the **Instance Information** page of the application to view the running status of the added ECS instance.

If **Normal** is displayed, the scale-out operation is successful.

Note:

- The “Purchase based on current ECS specifications” method involves two change orders: the first is made by EDAS for your ECS instance purchase, while the other automatically adds the purchased ECS instance to the application.
- It takes about three minutes between the request submission

and the start of application scale-out. The two change orders are executed at an interval of 10 seconds.

- The purchased ECS instance carries the same basic information as the selected one, including the instance specifications, disks, network, user data, and labels.
- All billing details are subject to normal ECS and EDAS charges. This operation itself does not generate any additional charges.
- The default logon information in the ECS instance is based on the key pair that you set. EDAS does not intrude on any of your private information.

Scale in

1. Log on to the Application scale-out or scale-in increases or decreases the computational capacity of an application by changing the number of ECS instances. You can add a new ECS instance when the ECS instance where an application resides is overloaded, or can remove the ECS instance when it is no longer required. You can also use **Auto Scaling** to dynamically adjust the number of ECS instances.

Scale out

When the ECS instance for hosting an application is overloaded, you can manually scale the application out.

Note: The running status of the ECS instance added depends on the running status of the application.

- If the application is running during scale-out, the ECS instance automatically deploys, starts, and runs the application.
- If the application is stopped during scale-out, the ECS instance automatically deploys but does not start or run the application.

Log on to the EDAS console.

In the left-side navigation pane, choose **Application Management** > **Applications**. On the **Applications** page, click the name of the target application.

On the Application Details page, click **Scale Out** in the upper-right corner.

In the **Scale-Out Method** dialog box, select **Target Group** for scale-out.

Select **Scale-Out Method**, and complete the subsequent scale-out steps.

Select from Current Cluster

Idle ECS instances in the cluster are added to the application for scale-out.

Select an ECS instance, and then click **Scale-Out**.

The message **Scale-out successful** is displayed on the page.

Go to the **Instance Information** page of the application to view the running status of the added ECS instance.

If **Normal** is displayed, the scale-out operation is successful.

Purchases based on current ECS specifications

This method allows you to select any ECS instance in the cluster as the specification template. Then, new purchases are configured according to the specifications of the template.

Select the ECS instance you want to use as the specification template, and click **Next**.

On the **Purchase Details** page, enter **Purchase Quantity** and **Login Key**, select **ECS Service Terms | Image Service Terms**, and then click **Next**.

On the **Confirm** page, view the information about the purchased ECS instance, and click **Confirm**.

On top of the page, the message **Automatic purchasing is triggered. Check the real-time information in the application change process.** is displayed after your submission.

Go to the **Instance Information** page of the application to view the running status of the added ECS instance.

If **Normal** is displayed, the scale-out operation is successful.

Note:

- The “Purchase based on current ECS specifications” method involves two change orders: the first is made by EDAS for your ECS instance purchase, while the other automatically adds the purchased ECS instance to the application.
- It takes about three minutes between the request submission

and the start of application scale-out. The two change orders are executed at an interval of 10 seconds.

- The purchased ECS instance carries the same basic information as the selected one, including the instance specifications, disks, network, user data, and labels.
- All billing details are subject to normal ECS and EDAS charges. This operation itself does not generate any additional charges.
- The default logon information in the ECS instance is based on the key pair that you set. EDAS does not intrude on any of your private information.

Scale in

Log on to the EDAS console.

In the left-side navigation pane, choose **Application Management** > **Applications**. On the **Applications** page, click the name of the target application.

On the Application Details page, click the **Instance Information** tab.

On the **Instance Information** tab page, delete the ECS instance.

If the ECS instance is running, click **Stop** and then click **Delete**.

If the ECS instance is stopped, click **Delete**.

In the left-side navigation pane, choose **Application Management** > **Applications**. On the **Applications** page, click the name of the target application.

On the Application Details page, click the **Instance Information** tab.

On the **Instance Information** tab page, delete the ECS instance.

If the ECS instance is running, click **Stop** and then click **Delete**.

If the ECS instance is stopped, click **Delete**.

Manage instance groups

Overview

This function groups all ECS instances for an application so that you can deploy different application package versions for the ECS instances in different groups.

For example, there are 10 instances in the "itemcenter" application, which are divided into two groups: "Default Group" and "Beta Group". The default group contains six instances and the Beta group contains four. Now there are two groups of instances in the application to which you can deploy different versions of the application package.

Note:

1. When an application is created in EDAS, a new group "Default Group" will be created by default for the application, and cannot be deleted.
2. If no multi-version deployment is required, the default group is generally adequate.

Create groups

Gated release is often used for the launch of new application versions, so that the new version can be test run without affecting the traffic in the production environment. In this case, you need to create a new group for the application.

Log on to the EDAS console.

In the left-side navigation pane, choose **Application Management** > **Applications**. On the **Applications** page, click the name of the target application.

On the Application Details page, click the **Instance Deployment Information** tab, and click **Create Group** in the upper-right corner of the page.

In the **Create Group** dialog box, enter the **Group Name** and then click **Create**.

After you successfully create a group, the message **Group created successfully** is displayed in the upper-right corner of the page.

Add instances

After creating a group, you can add instances to it through **Scale Out** and **Change Group**. Perform the following operations:

Add instances to the group through **Scale Out**. For details, see [Scale out and scale in applications \(ECS cluster\)](#).

Add instances to the group through **Change Group**

Click the **Instance Information** tab on the **Application Details** page, select the instance whose group you want to change and click **Change Group** on the right of the list.

In the **Change Group** dialog box, select **Target Group**, and then click **OK**.

If the application package version for the instance is different from that of the target group, use the version for the target group or keep the existing version for the instance.

Notes about changing instance groups

- If no package version is available for the target group and a package is already deployed to the instance whose group you want to change, the package version for the instance will be used as the package version for the group.
- Choose **Redeploy current instance for target group** to redeploy the deployment package on the instance using the version in the group.
- If you select **Change Group Only Without Re-Deployment**, the deployment status of the instance remains unchanged.
- If the package version for the instance is different from that of its group, a prompt is displayed.

View groups

Log on to the EDAS console.

In the left-side navigation pane, choose **Application Management** > **Applications**. On the **Applications** page, click the name of the target application.

On the Application Details page, click the **Instance Information** tab to check the application group information and deployment package versions for different groups.

Notes about groups:

- Each group corresponds to one deployment package version, which is displayed following the group name.
- No package version is available for a new group. The package version for a group is always the version of the package last deployed to the group.
- Application instances are displayed by instance group.

Set group parameters

On the **Application Details** page, click the **Instance Information** tab and click the **Group Settings** button on the right of the group.

On the **Group Settings** page, click **JVM** or **Tomcat**, set JVM and Tomcat parameters, and then click **Save** or **Configure Tomcat Parameters**.

JVM and Tomcat parameters are described in [Application settings \(ECS cluster\)](#).

Delete groups

A group with no instances in it can be deleted. The delete operation cannot be undone. Exercise caution when performing this operation.

On the **Application Details** page, click the **Instance Information** tab and click the **Delete Group** button.

In the **Delete Group** dialog box, click **Delete**.

Roll back ECS application

After updating an application, you can roll it back to an earlier version if any problem is detected.

Log on to the EDAS console.

In the left-side navigation pane, choose **Application Management** > **Applications**. On the **Applications** page, click the name of the target application.

Click **Roll back application** in the upper-right corner of the Application Details page.

In the **Roll Back Application** dialog box, select a **deployment package version** and **group**, set the **batch** and **batch mode**, and click **Roll Back**.

Delete applications

After an application is deleted, all information relating to the application is deleted, all instances under the application are released, and all WAR packages and container files are deleted from the ECS instance.

Note: Before deleting an application, disable the application and ensure that the logs, WAR packages, and configurations of all instances under the application are backed up.

Log on to the EDAS console.

In the left-side navigation pane, choose **Application Management** > **Applications**. On the Applications page, click the name of the target application.

On the Application Details page, click **Delete Application**.

Delete the application as instructed on the page.

After you delete the application, the **Application deletion triggered successfully** message is displayed in the upper-right corner of the page.

Application in Container Service Kubernetes Cluster

Overview of Container Service Kubernetes

application lifecycle management

Kubernetes is a popular orchestration technology for open-source containers. Publication of applications with Kubernetes offers unique management advantages. For more information, see [Kubernetes documentation](#). The Container Service Kubernetes clusters provided by Alibaba Cloud have passed CNCF standardized tests and can operate stably while integrated with other Alibaba Cloud products, such as SLB and Network Attached Storage (NAS).

After creating a Kubernetes cluster in Container Service and importing it to EDAS, you can use image, WAR and JAR to deploy applications in the Container Service Kubernetes cluster from EDAS console.

After publishing the application, you can scale it in or out as needed.

Create images for Container Service Kubernetes applications

This topic describes how to create custom images in HSF:

- Create a standard Dockerfile
- Customize the Dockerfile
- Save the image and upload it to the image repository

Note: Before creating an image for a Container Service Kubernetes application, read [Restrictions on creating images](#) and create an image accordingly.

Create a standard Dockerfile

Dockerfile is a configuration file in text format. You can use a Dockerfile to quickly create a custom image. A custom Dockerfile defines all instructions in the runtime environment for EDAS applications, including the instructions on downloading, installing, and starting JDK, Tomcat, WAR, and JAR packages.

By modifying the Dockerfile, you can replace the JDK version, modify the Tomcat configuration, change the runtime environment, and make other changes.

The following example (which is updated from time to time to introduce the latest EDAS features) shows how to create and deploy an application in EDAS.

Sample Dockerfile used to deploy applications on Ali-Tomcat with a WAR package

```
FROM centos:7
MAINTAINER EDAS R&D team<edas-dev@list.alibaba-inc.com>

# Install and package the required software
RUN yum -y install wget unzip

# Prepare JDK or Tomcat system variables and paths
ENV JAVA_HOME /usr/java/latest
ENV CATALINA_HOME /home/admin/taobao-tomcat
ENV PATH $PATH:$JAVA_HOME/bin:$CATALINA_HOME/bin

# Set EDAS Container or Pandora Container version
ENV EDAS_CONTAINER_VERSION V3.5.0
LABEL pandora V3.5.0

# Download and install JDK 8
RUN wget http://edas-hz.oss-cn-hangzhou.aliyuncs.com/agent/prod/files/jdk-8u191-linux-x64.rpm -O /tmp/jdk-8u191-linux-x64.rpm && \
yum -y install /tmp/jdk-8u191-linux-x64.rpm && \
rm -rf /tmp/jdk-8u191-linux-x64.rpm

# Download and install Ali-Tomcat 7.0.85 to the /home/admin/taobao-tomcat directory
RUN wget http://edas-hz.oss-cn-hangzhou.aliyuncs.com/edas-container/7.0.85/taobao-tomcat-production-7.0.85.tar.gz -O /tmp/taobao-tomcat.tar.gz && \
mkdir -p ${CATALINA_HOME} && \
tar -xvf /tmp/taobao-tomcat.tar.gz -C ${CATALINA_HOME} && \
mv ${CATALINA_HOME}/taobao-tomcat-production-7.0.59.3/* ${CATALINA_HOME}/ && \
rm -rf /tmp/taobao-tomcat.tar.gz ${CATALINA_HOME}/taobao-tomcat-production-7.0.59.3 && \
chmod +x ${CATALINA_HOME}/bin/*sh

# Download and install EDAS Container or Pandora Container based on environment variables
RUN wget http://edas-hz.oss-cn-hangzhou.aliyuncs.com/edas-plugins/edas.sar.${EDAS_CONTAINER_VERSION}/taobao-hsf.tgz -O /tmp/taobao-hsf.tgz && \
tar -xvf /tmp/taobao-hsf.tgz -C ${CATALINA_HOME}/deploy/ && \
rm -rf /tmp/taobao-hsf.tgz

# Download and deploy the EDAS demo WAR package
RUN wget http://edas.oss-cn-hangzhou.aliyuncs.com/demo/hello-edas.war -O /tmp/ROOT.war && \
unzip /tmp/ROOT.war -d ${CATALINA_HOME}/deploy/ROOT/ && \
rm -rf /tmp/ROOT.war

# Set the Tomcat installation directory as the container startup directory, start Tomcat in run mode, and output the catalina log on the standard CLI.
WORKDIR ${CATALINA_HOME}
CMD ["catalina.sh", "run"]
```

Sample Dockerfile used to deploy applications on Ali-Tomcat with a JAR package

```

FROM centos:7
MAINTAINER EDAS R&D team<edas-dev@list.alibaba-inc.com>

# Install and package the required software
RUN yum -y install wget unzip

# Prepare JDK or Tomcat system variables and paths
ENV JAVA_HOME /usr/java/latest
ENV CATALINA_HOME /home/admin/taobao-tomcat
ENV PATH $PATH:$JAVA_HOME/bin

# Set EDAS Container or Pandora Container version
ENV EDAS_CONTAINER_VERSION V3.5.0
LABEL pandora V3.5.0

# Download and install JDK 8
RUN wget http://edas-hz.oss-cn-hangzhou.aliyuncs.com/agent/prod/files/jdk-8u191-linux-x64.rpm -O /tmp/jdk-8u191-linux-x64.rpm && \
yum -y install /tmp/jdk-8u191-linux-x64.rpm && \
rm -rf /tmp/jdk-8u191-linux-x64.rpm

# Download and install EDAS Container or Pandora Container based on environment variables
RUN mkdir -p ${CATALINA_HOME}/deploy/
RUN wget http://edas-hz.oss-cn-hangzhou.aliyuncs.com/edas-plugins/edas.sar.${EDAS_CONTAINER_VERSION}/taobao-hsf.tgz -O /tmp/taobao-hsf.tgz && \
tar -xvf /tmp/taobao-hsf.tgz -C ${CATALINA_HOME}/deploy/ && \
rm -rf /tmp/taobao-hsf.tgz

# Download and deploy the EDAS demo JAR package
RUN mkdir -p /home/admin/app/ && wget http://edas.oss-cn-hangzhou.aliyuncs.com/demoapp/fatjar-test-case-provider-0.0.1-SNAPSHOT.jar -O /home/admin/app/provider.jar

# Include the startup command in the startup script start.sh
RUN echo '$JAVA_HOME/bin/java -jar $CATALINA_OPTS -Djava.security.egd=file:/dev/./urandom -Dcatalina.logs=${CATALINA_HOME}/logs -Dpandora.location=${CATALINA_HOME}/deploy/taobao-hsf.sar "/home/admin/app/provider.jar" --server.context-path=/ --server.port=8080 --server.tomcat.uri-encoding=ISO-8859-1 --server.tomcat.max-threads=400' > /home/admin/start.sh && chmod +x /home/admin/start.sh

WORKDIR $CATALINA_HOME
CMD ["/bin/bash", "/home/admin/start.sh"]

```

Customize the Dockerfile

You can configure the following custom settings in the preceding standard Dockerfile.

Upgrade JDK

Change the download and installation methods in the standard Dockerfile. The following uses JDK 8 as an example.

```

# Download and install JDK 8
RUN wget http://edas-hz.oss-cn-hangzhou.aliyuncs.com/agent/prod/files/jdk-8u191-linux-x64.rpm -O /tmp/jdk-

```

```
8u191-linux-x64.rpm && \  
yum -y install /tmp/jdk-8u191-linux-x64.rpm && \  
rm -rf /tmp/jdk-8u191-linux-x64.rpm
```

Upgrade EDAS Container

When using a WAR package and Tomcat, upgrade EDAS Container to use new middleware features or fix known bugs. The upgrade procedure is as follows:

1. Log on to the EDAS console. In the left-side navigation pane, choose **Application Management** > **Applications**. On the **Applications** page, click **Create Application** in the upper-right corner. On the **Application Information** page, check **Application Runtime Environment** and the latest EDAS Container version (3. X.X).
2. For more information about Pandora Container versions, see [Container versions](#).
3. Replace the version number in the Dockerfile, such as 3.5.0.
4. Re-create and publish an application image.

```
# Set EDAS Container or Pandora Container version  
ENV EDAS_CONTAINER_VERSION V3.5.0
```

Add the EDAS runtime environment to Tomcat startup parameters

For the environment variables, see [Environment variables during application runtime](#). EDAS provides the JVM environment variable `EDAS_CATALINA_OPTS`, which contains the minimum parameters required during runtime. Ali-Tomcat provides the custom JVM parameter configuration option `JAVA_OPTS` for setting `Xmx`, `Xms`, and other parameters.

```
# Set the JVM parameters of the EDAS application  
ENV CATALINA_OPTS ${EDAS_CATALINA_OPTS}  
# Set the JVM parameters  
ENV JAVA_OPTS="\  
-Xmx3550m \  
-Xms3550m \  
-Xmn2g \  
-Xss128k"
```

Upload an image to the image repository

After the image is created, you can upload the local image to Alibaba Cloud Docker registry.

For example, run the following command to push the packaged local image to the remote server for deployment:

```
docker push registry.cn-hangzhou.aliyuncs.com/edas/demo-frontend-service:20181111
```

- EDAS console: <https://edas.console.aliyun.com>
- Image repository console: <https://cr.console.aliyun.com>
- Image network accelerator: <https://cr.console.aliyun.com/#/accelerator>

Restrictions on creating images

Observe the following specifications and constraints when creating a custom image by using a Dockerfile:

Tenant and encryption information

The tenant and encryption information is used for the user authentication and credential encryption of EDAS applications.

Resources

Resource type	Resource name	Description
Secret	edas-certs	Encryption dictionary, which stores EDAS tenant information

Environment variables

Environment variable key	Type	Description
tenantId	String	EDAS tenant ID
accessKey	String	Access Key ID for authentication
secretKey	String	Access Key Secret for authentication

Local files

Path	Type	Description
/home/admin/.spas_key/default	File	EDAS tenant authentication information, which includes the preceding environment variable information

Service information

The service information includes the EDAS domain and port to be connected to during runtime.

Resources

Resource type	Resource name	Description
ConfigMap	edas-envs	EDAS service information

Environment variables

Environment variable key	Type	Description
EDAS_ADDRESS_SERVER_DOMAIN	String	Service domain or IP address of the configuration center
EDAS_ADDRESS_SERVER_PORT	String	Service port of the configuration center
EDAS_CONFIGSERVER_CLIENT_PORT	String	CS service port

Environment variables during application runtime (required)

The following environment variables are provided during EDAS deployment to ensure the proper running of applications. For this reason, do not overwrite the current configuration.

Environment variable key	Type	Description
POD_IP	String	POD IP
EDAS_APP_ID	String	EDAS application ID
EDAS_ECC_ID	String	EDAS ECC ID
EDAS_PROJECT_NAME	String	Same as EDAS_APP_ID, which is used to parse the trace
EDAS_JM_CONTAINER_ID	String	Same as EDAS_ECC_ID, which is used to parse the trace
EDAS_CATALINA_OPTS	String	CATALINA_OPTS parameter, which is required during middleware operation
CATALINA_OPTS	String	Same as EDAS_CATALINA_OPTS, which is the default Tomcat startup parameter

Publish Container Service Kubernetes applications

At present, EDAS only allows you to publish Container Service Kubernetes applications by using images.

Prerequisites

- You have created a Container Service Kubernetes cluster.
- You have created a Container Service Kubernetes application image.

Create applications

Log on to the EDAS console.

In the left-side navigation pane, choose **Application Management** > **Applications**.

On the Applications page, select **Region** and click **Create Application** in the upper-right corner.

On the **Application Information** page, set the basic information about the application. Then click **Next Step: Application Configurations**.

- **Namespace:** Select **Region** and **Namespace** from the drop-down list. If you do not select a namespace, the **default** namespace is automatically selected.
- **Cluster Type:** Select **Container Service K8S Cluster** from the drop-down list and then select a specific cluster.
- **Application Name:** Enter the application name.
- **Application Description:** Enter up to 100 characters of basic application information.

Go to the **Application Configuration** page and configure an image.

Select **Image** for **Application Deployment Method**.

Select an image in **My Image**.

Note:

If you are publishing a Container Service Kubernetes application for the first time, no images will be available in the image market. Follow the instructions provided in **Create a Container Service Kubernetes**

application image to create an image and upload it to the repository.

You cannot pull images across regions.

Set pods.

Pods are the smallest units for deploying an application. An application can contain multiple pods. In the event that load balancing can be achieved, a request is randomly allocated to a pod for processing.

Set Total Pods.

When the pods of an application fail to run or are faulty, they can be automatically restarted or quickly migrated to ensure the high availability of the application. For stateful applications using persistent storage, instance data can be saved. For redeployed stateless applications, instance data is not saved.

Set Single Pod Resource Quota.

No quota is set by default. Therefore, both the values of **CPU Cores** and **Memory** of a single pod are 0. To set the quota, enter a number.

Set the startup command and parameters.

Note: We recommend that you do not modify the custom startup command and parameters if you are not familiar with the **CMD** and **ENTRYPOINT** of the original Dockerfile. An incorrect custom command could lead to an application creation failure.

- **Startup Command:** Enter content in [""] only. For example, to run CMD ["/usr/sbin/sshd" ," -D"], enter /usr/sbin/sshd -D only.
- **Startup Parameters:** Enter one parameter per line. For example, args: ["-c" ; "while sleep 2" ; "do echo date" ; "done"] contains four parameters. Enter them separately in four lines.

Set environment variables.

When creating the application, inject the entered environment variables in the container to be generated. This saves you from having to repeatedly add common environment variables.

If you are using a MySQL image, refer to the following environment variables:

- **MYSQL_ROOT_PASSWORD** (required) allows you to set a root password for MySQL.
- **MYSQL_USER** and **MYSQL_PASSWORD** (optional) allows you to add an account in

- addition to the root account, and to set a password.
- `MYSQL_DATABASE` (optional) allows you to set the database to be created when the container is generated.

If you are using another type of image, configure the environment variables as needed.

Set persistent storage.

In the Container Service Kubernetes cluster, the native Volume object corresponds to non-persistent physical storage, whose lifecycle is the same as the Kubernetes pods, which are a transient storage object. **Network Attached Storage (NAS)** is a persistent storage service that can store instance data permanently and ensure data integrity after application upgrade or migration.

Note: Before configuring persistent storage, ensure that the NAS service is active for your EDAS account. Because the NAS billing method is Pay-As-You-Go, ensure that your account has a sufficient balance or adopts post payment.

Description of persistent storage parameters:

- **Storage Type:** This parameter is NAS by default and cannot be configured manually.
- **Storage Service Type:** Currently, this parameter only supports SSD and cannot be configured manually.

Select NAS:

Buy a new NAS: Select a NAS mount directory and a local mount directory.

A single region supports up to 10 NAS instances. Once you have 10, you cannot create any more instances. If you need to create more instances, submit a ticket.

Use Existing NAS: Select an existing NAS instance.

You can create up to two mount points. Only compliant NAS instances are displayed in the drop-down list.

Set local storage.

You can map part of the file system of the host machine to the container, as needed. Before using this feature, read this document to determine if this is the correct solution.

Description of file types:

Name	value	Description
Default	Null string	Mount directly without checking the type.
(New) File directory	DirectoryOrCreate	File directory; a new directory is created if this does not exist.
File directory	Directory	File directory; container startup fails if this does not exist.
(New) File	FileOrCreate	File; a new file is created if this does not exist.
File	File	File; container startup fails if this does not exist.
Socket	Socket	Standard Unix Socket file; container startup fails if this does not exist.
CharDevice	CharDevice	Character device file; container startup fails if this does not exist.
BlockDevice	BlockDevice	Block device file; container startup fails if this does not exist.

Note: You do not need to pay attention to the Value column in this step. However, the Value column may be used by APIs after the application is created.

Set the lifecycle management script of the application (**for stateful applications**).

Container Service Kubernetes applications can be stateful or stateless:

Stateless: A stateless application supports multi-replica deployment. When a stateless application is redeployed, instance data is not saved. A stateless application can be:

- A web application that does not retain instance data during upgrade or migration.
- An application that can be scaled up horizontally to address changing service volumes.

Stateful: A stateful application stores data that requires persistent storage and retains instance data during upgrade or migration. A stateful application can be:

- An application that frequently operates on containers through SSH.

- An application that is used for persistent data storage (such as the database application MySQL) or that supports inter-cluster election and service discovery, such as ZooKeeper and etcd.

You can set lifecycle management for a stateful application as needed.

Description of the lifecycle management script:

PreStop script: a container hook, which is triggered before a container is deleted. The corresponding hook handler must be completed before the container deletion request is sent to Docker daemon. Docker daemon sends an SGTERN semaphore to itself to delete the container, regardless of the result of hook handler execution. For more information, see [Container Lifecycle Hooks](#).

Liveness script: a container status probe, which monitors the health status of applications. If an application is unhealthy, the container is deleted and created again. For more information, see [Pod Lifecycle](#).

Readiness script: a container status probe, which monitors whether applications have started successfully and are running properly. If an application is abnormal, the container status is updated. For more information, see [Pod Lifecycle](#).

Poststart script: a container hook, which is triggered immediately after a container is created to notify the container of its creation. The hook does not transfer any parameters to the corresponding hook handler. If the corresponding hook handler fails to run, the container is killed and the system determines whether to restart the container according to the container's restart policy. For more information, see [Container Lifecycle Hooks](#).

Then, click **Create**.

Result verification

Creating an application can take up to several minutes. During the creation process, you can use an [application publishing order](#) to track the creation progress. After the application has been created, return to the [Application Details](#) page and check whether the application status is **Normal** on the [Application Information](#) tab page.

The Container Service Kubernetes application does not need to be deployed. The application is published once it is created.

Scale out and scale in Container Service Kubernetes applications

Compared with ECS applications and Swarm applications, Container Service Kubernetes applications feature much greater scalability due to the advantages of Kubernetes in container orchestration.

Log on to the EDAS console.

In the left-side navigation pane, choose **Application Management > Applications**. On the **Application Management** page, click a Container Service Kubernetes application.

On the Application Details page, click **Application Scaling** in the upper-right corner.

In the **Application Scaling** dialog box, set **Total Application Pods** to be added or removed and click **OK**.

If the number of pods is 0, the system physically deletes all pods under the application and retains only the basic creation information of the application.

After configuration, scale-out or scale-in of Container Service Kubernetes applications is automatically completed in the cluster without manual operations.

View application changes

After lifecycle operations in the EDAS console, such as application deployment, starting, scale-out or scale-in, you can go to the **Application Details** page to check the change status or go to the **Change Logs** page to check historical change logs.

View application changes

The following uses an example of application deployment to describe how to view application changes.

Return to the **Application Details** page.

At the top of the **Application Details** page, the message **A change order is being executed** is displayed.

Click **Details**. On the **Change Details** page, check change information and real-time status of the application.

The page contains two areas: change summary and change process execution information.

- Change summary: includes the change order ID, execution time, batch number, and batch mode (manual or automatic trigger).

Change process execution information: includes each stage of the entire process. For example, *Process Start*, *SLB Offline*, *Deploy*, *SLB Online*, and *Process Complete* are included. The execution results of each stage are identified by icons.

- ✓ : indicates that the step is completed successfully.
- Running: indicates that the step is being executed.
- ✗ : indicates that the step failed. Click **View Details** to check specific information and locate failure causes.

Deploy is a virtual stage.

Click **View Details** under **Deploy** to check the execution process and stages of an application.

Note: **Host** indicates application instance information.

Click **View Log** under a specific stage of the application instance to check task execution logs.

Click a task, such as *Start Application Instance*. The task execution log is displayed in the right-side area.

View application change logs

On the **Application Details** page, click **Change Logs** in the left-side navigation pane to check the logs of all application changes.

You can click **View** in the **Actions** column to check change orders and details about actions.

Health check

In the health check process, EDAS Agent periodically checks and reports the status of containers and applications and then sends the results to the console. Health checking can help you understand the overall condition of service operation in the cluster environment, and facilitates auditing and troubleshooting. You can configure a health check URL on the EDAS console to check whether the deployed applications are running properly.

If your business is highly sensitive to the traffic load, frequent health checking may impact normal service. You can reduce this impact by reducing the frequency of health checking or increasing the health check interval. To guarantee the service availability, we recommend that you do not disable health checking.

Health check process

You can configure health checks to monitor end nodes specified by IP address or domain name. EDAS health check automatically submits requests to your applications, servers, or other resources at a fixed interval you specify to verify whether it is accessible, available, and functional. You can also make a request similar to a user's request by configuring a health check URL to verify the running status of the application features.

A health check is triggered every 10 seconds. The details of Steps 1 and 2 in the figure are as follows:

EDAS Agent checks whether the Ali-Tomcat process for running your application is alive.

If the process is alive, EDAS Agent proceeds to Step 2.

If the process is not alive, the health check ends and the check result is "Fail".

EDAS Agent checks whether status code 200 is returned for the set URL.

If no URL is configured, health checking stops. If you set a URL, EDAS Agent checks whether the HTTP code 200 is returned for the set URL.

Create, update, and delete health checks (console)

In the EDAS console, you can view the running status of applications ("Normal" or "Runtime Error") and health checking process reports. For all health checks, except calculated health checks, you can also view the reasons for failure of the last health check.

You can create a health check URL when creating an application. You can also add or modify health checks on the **Application Settings** page after an application has been created and deployed. For ECS applications and Swarm applications, you can also click the **Settings** button on the **Application Settings** page, and set the Health Check parameters.

Log on to the EDAS console.

In the left-side navigation pane, choose **Application Management** > **Applications**. On the **Applications** page, click **Create Application** in the upper-right corner.

In the **Create Application** dialog box, enter application information and click **Next Step: Application Configurations**. On the **Application Configuration** page, click **Deployment Method** and follow the instructions on the page. On the **Application Configuration** page, you can create *Application Health Check*.

Note: The port range is 1 to 65535.

应用健康检查:	http://127.0.0.1:8080/healthCheck.html
---------	--

Example: When deploying a WAR package, you can set the health check URL to `http://127.0.0.1:8080/order/healthCheck.html` if no other container parameters are configured. Also, you can set the health check URL to `http://127.0.0.1:8081/healthcheck.html` if the container path is set to the root path, the port is set to "8081", and the WAR package contains the "healthcheck.html" file used for health status identification.

After creating a health check, you can view the settings of the health check in the **Application Settings** area on the Application Details page. You can also click **Modify** to modify or delete the health check settings.

On the Application Details page, click the **Settings** button to the right of the **Application Settings** area for ECS applications and Swarm applications. In the displayed Application Settings dialog box, click the **Health Check** tab. On the tab page that is displayed, follow the instructions in Step 3 to add, modify, or delete health check settings.

Description of health check status:

After you configure the application health check, the application displays different statuses.

Container Exited: displayed if EDAS Agent detects that the Ali-Tomcat process is not alive in Step 1 of the health check flow chart.

Application Exception: displayed when any code other than 200 is returned for the URL set in step 2 of the health check flow chart.

Normal: displayed if no exception occurs in steps 1 and 2 of the health check flow chart.

If EDAS Agent detects that no URL is set, the Normal state followed by an exclamation mark is displayed. When you move the cursor over the status, the prompt “Please configure the application’s health check URL so that its running status can be reflected more accurately” is displayed.”

Agent Exception: displayed if EDAS Agent does not report status information to the EDAS server within 30 seconds.

Log directories

The EDAS console allows you to view application runtime logs without having to log on to the server. When an exception occurs, you can check logs to troubleshoot the problem.

Follow these steps to view a runtime log:

Log on to the EDAS console. Choose **Applications Management > Applications** in the left-side navigation pane.

On the **Applications** page, click the name of the application you want to check to go to the application details page.

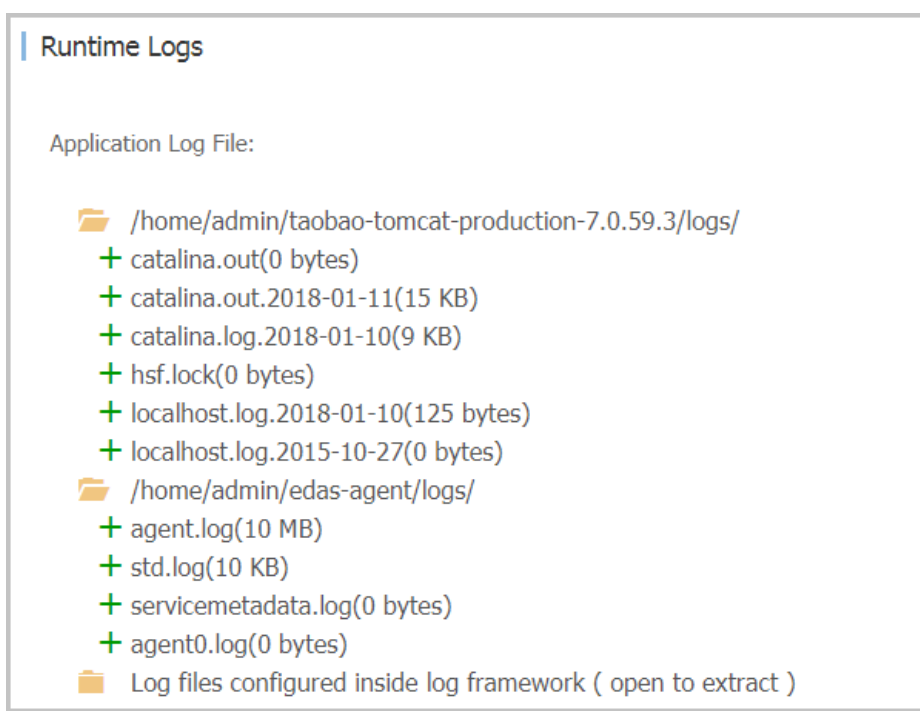
In the left-side navigation pane, select **Logs > Log Directories**. Alternatively, you can also go to the **Instance Information** tab of the application details page, and click **Logs** in the **Actions** column.

On the **Log Directories** page, 3 log directories are displayed by default:

Tomcat container logs directory: The specific paths for Tomcat container logs depend on its actual version.

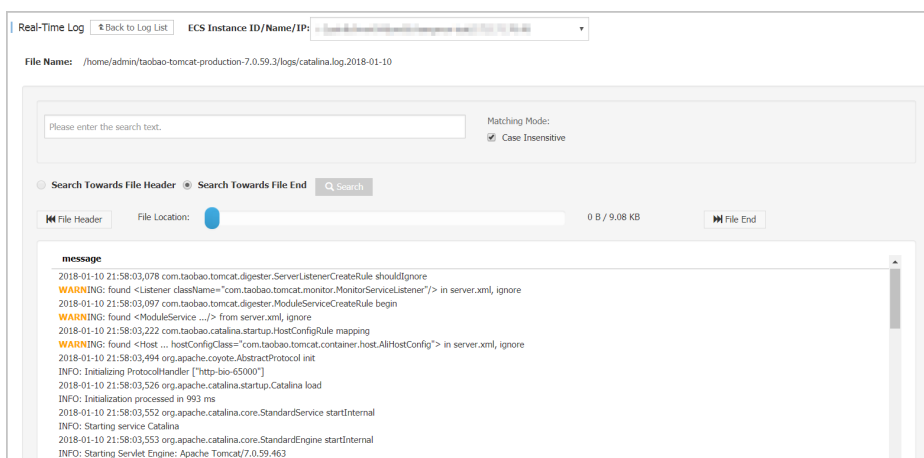
EDAS Agent logs directory.

Log files for log framework configurations.



Note: Only readable files are displayed in the file directory. No folders are displayed.

Double-click a log file to view the details of the log.



At the top of the page, select an instance ID or name in the drop-down list next to **ECS Instance ID/Name/IP Address** to view the details.

At the bottom of the page, click **Enable Real-Time Additions** to keep loading the latest additions to the log file (similar to the tailf command).

In addition to checking the logs in the default path, you can also add log paths to favorites for later viewing, or remove a path from your favorites.

Bookmark a directory

On the **Log Directories** page, click **Bookmark Log Directory** to add a log directory.

Note: This path must be under the /home/admin directory, and must contain wordings "log" or "logs" in the complete path. The file must end with a slash "/" to indicate that it is a folder.

Remove from bookmark

On the **Log Directories** page, click to select a folder name. Then click **Remove Directory from Bookmark** at the top right corner of the page. The path will no longer be displayed on the page. This operation does not delete or change any files on the server.

Real-time logs

On the EDAS console, you can view real-time logs of each pod about **Kubernetes Application** and **Container Service K8S Application**. When an exception occurs, you can check real-time pod logs to troubleshoot.

Log on to the EDAS console. In the left-side navigation pane, choose **Application Management > Applications**. On the Applications page, click **Kubernetes Application** or **Container Service K8S Application**.

On the Application Details page, choose **Real-Time Logs** in the left-side navigation pane.

Select a pod from the **Pod Name** drop-down list and check the real-time log for analysis.

Alarm and notification

EDAS provides the alarm function to notify users of online problems when resource usage exceeds the limit. Based on policies configured by users and data collected in the background, EDAS checks whether the resource usage limit is exceeded. If the limit is exceeded, a text message or an email is sent to specified contacts.

Note: Currently, EDAS only provides SMS and email notification and does not support custom notification.

Configure alarm policies

Follow these steps:

Log on to the EDAS console, select **Applications Management** > **Applications** in the left-side navigation pane, and click the name of the application from the application list.

Select **Alarm and Notification** > **Alarm Rules** in the left-side navigation pane and click **Create Rule** in the upper-right corner.

Enter relevant information in the **Create Rule** page.

*Rule Name: Rule names must only contain numbers, letters and underscores.

*Monitoring Target:

Monitoring Metrics	Compare	Threshold	Actions
CPU Usage	>		%

+ Add Monitor Items

*Trigger Conditions:

*Statistical Cycle:

*Retries Before Alarm:

Field description:

Rule Name: Name of the alarm rule, which can contain numbers, letters, and underscores (_). Use an understandable name.

Monitoring Target: Create comparison policies based on metrics (basic monitoring, HTTP, HSF, and application container) and the configured threshold. You can add more than one target as needed.

Trigger Conditions: Select **Any One of the Indicators** or **All Indicators**.

- **Any One of the Indicators:** An alarm is triggered when any of the indicators of the monitored object meets the alarm rules.
- **All Indicators:** An alarm is triggered when all of the indicators of the monitored object meet the alarm rules.

Statistical Cycle: It can be set to 1 minute, 5 minutes, 15 minutes, 30 minutes, or 1 hour. A false alarm might be generated when the system encounters transient jitter, for example, when CPU usage is high during service startup but recovers to the normal range within 2 minutes. To avoid false alarms, you can select a statistical period to allow alarm trigger only when **alarm rules are continuously satisfied** within this period. For example, if you select the 5-minute statistical period for the metric "CPU usage above 30%", then EDAS determines an exception occurs when the CPU usage of the system exceeds 30% for 5 **consecutive** minutes.

- **Retries Before Alarm:** The number of consecutive statistical cycles when alarm policies are satisfied that are required to trigger an alarm. Optional values include 1, 3, and 5.

Click **OK**.

Alarm rules take effect once created. To disable an alarm rule, select it from the rule list and click "Delete". The rule is obsolete immediately.

Add alarm contacts

Follow these steps:

1. Log on to the EDAS console, select **Applications Management > Applications** in the left-side menu bar, and click the name of the application in the application list.
2. Select **Notification Alert > Alarm Contacts** on the left-side menu bar and click **Add Alarm Contacts** in the upper-right corner.
3. Select the contact from the contact list and click **OK**.

Note:

- **Alarm Contact Source:** You can configure to send alarms to the contacts that have a primary and sub-account relationship with the current account. Details are as follows:
 - Other Alibaba Cloud accounts that are bound as sub-accounts to the primary account
 - RAM sub-accounts with EDAS logon history
- **Contact Info (Email Address and Mobile Number):** By default, emails and contacts are obtained from Alibaba Cloud. For privacy protection, the contact information is only available after logon. If you want to receive notifications using a mobile number other than the one registered at Alibaba Cloud, make modifications on **Personal Information** page.

Add employees as alarm contacts

To add an employee that has never used EDAS as an alarm contact, following these steps. Assume

that the current EDAS primary account is master@alibabacloud.com and the account to be added is employee@company.com:

Add a RAM sub-account:

Log on to the Alibaba Cloud console with the account master@alibabacloud.com and select **Products & Services** > **RAM** to go to the RAM Console.

Click **Users** in the left-side navigation pane to go to the RAM sub-account page, click **Create User** in the upper-right corner, and fill in employee information to create a sub-account (assume that the employee name is "employee").

Log on to EDAS with the sub-account and modify information.

Log on to Alibaba Cloud with the employee RAM sub-account by clicking the link provided in RAM, and select **EDAS** to go to the EDAS console.

Select **Accounts** > **Personal Information** in the left-side navigation pane and enter your mobile number and email address.

After relevant information is modified, follow the steps in the **Add alarm contacts** section to add the employee to the alarm contact list.

View alarm records

After an alert is generated, the system sends the alert to contacts while recording the alert.

1. Log on to the EDAS console, select **Applications Management** > **Applications** in the left-side navigation pane, and select the expected application from the application list.
2. Select **Alarm and Notification** > **Alarm Records** on the left-side menu bar.

Alarm records from the past 10 days are displayed. After an alarm is cleared, a notification is generated and sent to contacts by means of text message and email.

Auto scaling

To ensure the service quality and availability of an EDAS distributed cluster, it is crucial to introduce

O&M capabilities that can detect the status of each server in the cluster and can scale the cluster in or out in real time based on the system load.

If the ECS instances in your cluster are insufficient for scale-out, use the Alibaba Cloud Auto Scaling feature to create hosts (you must use EDAS to activate Alibaba Cloud Auto Scaling in advance). **You are charged in pay-as-you-go mode for hosts created through elastic resources.**

EDAS provides the auto scaling function to automatically scale up or down a cluster based on the CPU, RT, and load metrics of the cluster servers.

Metric descriptions:

- **CPU**: CPU usage of the server in percentage. If multiple servers exist in the application, the average value of all servers is used.
- **RT**: Time for the system to respond to a request in ms.
- **Load**: System load, which is a positive integer.

These metrics must be positive integers without floating-point numbers. If multiple servers exist in the application, the average values of all servers are used for all the preceding metrics. Auto scaling includes automatic scale-in and scale-out, for which the rules can be configured separately.

Automatic scale-out

Log on to the EDAS console.

In the left-side navigation pane, choose **Application Management** > **Applications**. On the **Applications** page, click the name of the target application.

On the Application Details page, click **Auto Scaling** in the left-side navigation pane.

Click the switch to the right of **Scale-out Rule** to enable scale-out rules.

Configure the scale-out rule parameters, and then click **Save**.

Instance Source (for applications other than Docker)

Existing Resources: uses only the idle hosts in the cluster for automatic scale-out.

Elastic Resources: uses only the hosts created by Auto Scaling for automatic scale-out.

Existing Resources First: uses the idle hosts in the cluster first for automatic scale-out. If the cluster does not have sufficient idle hosts, the hosts created with elastic resources are used.

Note: If you select **Elastic Resources** or **Existing Resources First**, the hosts created by Alibaba Cloud Auto Scaling may be used, and in this case you are charged in pay-as-you-go mode.

In addition, you need to set the following parameters for the hosts created with elastic resources:

- **Specifications Template:** uses one of the existing hosts in the cluster as a template for automatic scale-out. Based on the template, new hosts inherit the CPU, memory, network, disk, and security group settings.
- **Network Type** and **Multi-Zone Scaling Policy:** **Network Type** indicates the network where the current application to be scaled out is located and cannot be changed. If the current network is Virtual Private Cloud (VPC), you must specify one or more virtual switches for the new host. If you specify two or more virtual switches, EDAS automatically allocates these switches through the **Multi-Zone Scaling Policy**.

Login Password: This password is used as the administrator (root user) password for a new host.

Trigger Indicators: includes the thresholds of CPU, RT, and Load indicators. Scale-out is triggered when a threshold is exceeded.

Trigger Condition

Any Metric: Automatic scale-up is implemented if any of the set metrics is triggered.

All Metrics: Automatic scale-up is implemented only when all of the set metrics are triggered.

Lasts for More Than: indicates the duration for which the indicators are triggered continuously, in minutes. Within the duration, if the average value of an indicator per minute continuously reaches the set threshold, scale-out is triggered. You can configure the duration based on the sensitivity of the cluster service capabilities.

Number of Instances for Each Scale-Out: indicates the number of servers automatically added upon each scale-out. You can set this parameter based on the service capabilities of a single server of the application.

Maximum Number of Instances: When the number of ECS instances in the cluster reaches the maximum, no more scale-out can be performed. You can set this parameter based on the resource quota.

Automatic scale-in

The **Automatic Scale-In** configuration process is similar to the **Automatic Scale-Out** configuration. For the definitions and configuration methods of the metrics, see **Automatic scale-out**.

Note:

- When you configure both scale-out and scale-in rules, the metrics of the scale-in rules cannot be greater than those of scale-out rules. Otherwise, an error message will be displayed when you click **Save**.
- If elastic resources are used, the hosts created with elastic resources are released first during scale-in.

View auto scaling results

After auto scaling rules have been set, when an automatic scale-in or scale-out is performed, you can check whether the number of ECS instances has changed in **Instance Information** on the **Basic Information** page.

Rate limiting and degradation

Overview of rate limiting and degr

The rate limiting and degradation feature of EDAS resolves slow system responses or crashes caused by the high pressure of the backend core services. This feature is generally used in high-traffic scenarios, for example, flash sales, shopping sprees, great promotions, and anti-empty box scams.

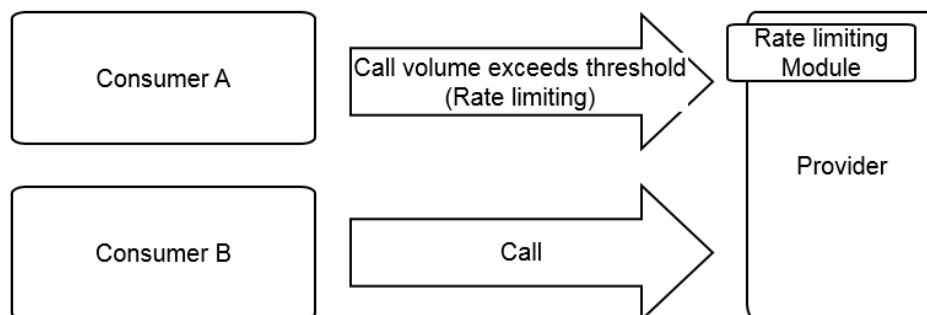
Rate limiting

This feature is used to control the traffic threshold or adjust the ratio. When a frontend website encounters high-traffic access, the traffic is controlled to prevent service unavailability and damage to

the backend core system. By adjusting the traffic threshold, the maximum traffic volume of the system is controlled to ensure secure and stable system running.

Basic principles

After a rate limiting module code is configured for a service provider and a rate limiting policy is configured on EDAS, the service provider can use the rate limiting function. When a service consumer calls the service provider, all access requests are calculated by the rate limiting module. If the call volume of the service consumer exceeds the preset threshold in a specific period, the rate limiting policy is triggered.

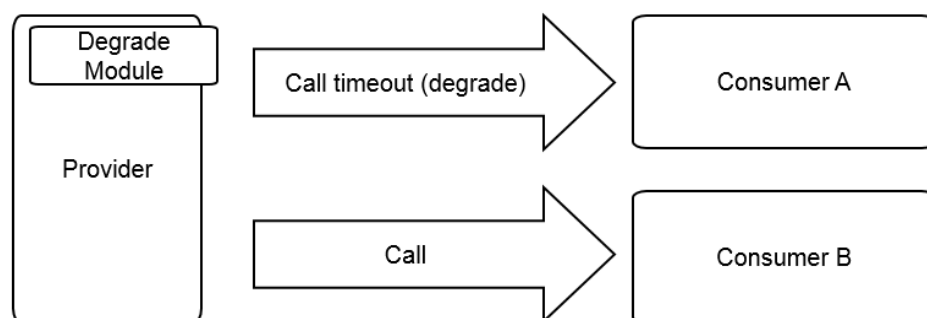


Degradation

Degradation is to lower the called priority of non-core service providers that are timed out to ensure the availability of core service consumers.

Basic principles

After a degradation module code is configured for a service consumer and a degradation policy is configured on EDAS, the degradation function is enabled for service consumers. When the service consumer calls a service provider, if the response time of the service provider exceeds the preset threshold, the degradation policy is triggered.



Container version management

An EDAS container consists of AliTomcat, Pandora, and custom Pandora plugins. In addition to the support for existing Apache Tomcat core functions, a class isolation mechanism, QoS, and Tomcat-Monitor are provided. Besides, highly customized plug-ins are added to EDAS containers to implement complex and advanced functions, such as container monitoring, service monitoring, and tracing. Applications deployed using EDAS must run in EDAS containers.

Basic concepts

AliTomcat

AliTomcat is a version of Apache Tomcat for internal use, which is developed by the Alibaba middleware team after performance optimization, bug fixing, and new feature development based on Apache Tomcat. AliTomcat is widely deployed and used in Alibaba Group, and offers enhanced performance, security, and stability compared with the community version.

Pandora and Pandora plugins

Pandora is a lightweight isolation container, which is taobao-hsf.sar. It is used to isolate dependence between web applications and middleware products and between middleware products so that they do not affect each other. Plug-ins implementing service discovery, configuration pushing, tracing, and other middleware products are integrated in EDAS Pandora. By using these plugins, you can monitor, process, track, analyze, maintain, and manage services of EDAS applications in all dimensions.

Container version

You must select the container version when creating an application on EDAS. EDAS containers are maintained and published by the EDAS development team. You can choose **Application Management > Software Version** to check the history and description of container publishing. Alternatively, you can view **Container version description**. Generally, a container of a higher version is superior to a container of a lower version in stability and functions variety.

The publishing of an EDAS container does not affect deployed applications. Once a new container is available, you can immediately upgrade the container to the latest version.

Upgrade or downgrade containers

In the left-side navigation pane of the EDAS console, choose **Application Management >**

Applications. On the **Applications** page, click the name of the target application.

On the **Application Details** page, click **Container Version** in the left-side navigation pane.

On the **Container Version** page, click **Upgrade to This Version** or **Downgrade to This Version** on the right side of the row corresponding to the container to upgrade or downgrade the container with a single click.

Configuration management

Application Configuration Management (ACM) has been integrated in EDAS as a component. You can use ACM in EDAS console to centrally manage and push configurations for applications, including Create configuration, View push status, Push track query and Query and rollback versions.

Microservice Management

Overview

Data-driven operations is an important set of functions within EDAS. The most important data-driven operations function is distributed link analysis.

Distributed link analysis analyzes every service distributed system call, all sent and received troubleshooting messages of interest, and all database access to help you precisely identify system bottlenecks and risks.

Data-driven operations provide these functions:

Trace query

By setting query conditions, you can accurately find businesses with poor performance or exceptions.

Trace details

Based on the trace query results, you can view detailed information for slow or erroneous businesses and reorganize their dependencies. This information allows you to identify frequent failures, performance bottlenecks, strong dependencies, and other problems. You can also evaluate business capacities based on link call ratios and peak QPS.

Service topology

The topology intuitively presents the calling between services and relevant performance data.

Service query

You can view the HSF, Spring Cloud and Service Mesh services in the specific region and namespace.

Service statistics

Service statistics shows the Total Calls in the Last 24 Hours, Average RT(ms) in the Last 24 Hours and Total Errors in the Last 24 Hours of current services in the specific region.

Trace query

By using the trace query function, you can view the status of the invocation trace in the system, especially for tasks that are slow or have encountered an error.

Log on to the EDAS console and Select **Microservice Management** > **Trace Query** in the left-side navigation pane.

Click **Show Advanced Options** in the upper-right corner of the **Trace Query** page to display more query conditions.

Specify the query conditions and click **Query**.

The screenshot shows the 'Trace Query' interface. At the top, there's a title bar with 'Trace Query' and a 'Hide Advanced Options' link. Below this, the interface is divided into several sections. On the left, there's a 'Time Range' section with a date selector set to '2018-01-15 09:50:09' and a dropdown for 'To 10 Minut'. Next to it is the 'Application Name' section with a dropdown menu showing 'Enter the application name to ... please inp...' and an 'Error' checkbox. Below these are three input fields: 'Client IP' (with placeholder 'Intranet IP Of Client'), 'Server IP' (with placeholder 'Intranet IP Of Server'), and 'Entrance IP' (with placeholder 'Internet IP or Intranet IP'). Further down is the 'Service Name' section with a text input field labeled 'Service Name or URL'. On the right side, there's a 'Results' dropdown set to '1~1000' and a 'Response Code' dropdown set to 'value'. At the bottom, there's a 'Call Type' dropdown set to 'Any', an 'Elapsed Time >' input field set to 'MS', a 'Request >' dropdown set to 'byte', a 'Response >' dropdown set to 'byte', and a 'Response Code' dropdown set to 'value'. A blue 'Query' button is located in the top right corner.

The descriptions for the parameters of the invocation traces (advanced query conditions) are as follows:

Time range: Click the time selector, set the query start time, and then select the end time. The options for the end time are “This Second” , “To 1 Minutes Later” , and “To 10 Minutes Later” . Therefore, the latest time periods are: last second, last one minute, and last ten minutes.

Application name: Select an application from the drop-down list. You can also enter a keyword to search for an application. Manual input of an application name is not supported.

Call type: Select the call type to query from the drop-down list. Options are HTTP, HSF provider, HSF consumer, MySQL, Redis cache, message sending, and message receiving.

Set the threshold values for time elapsed, request, or response for querying slow tasks in the system.

Select the **Error** check box in the upper-right corner to query the error cases only.

Specify other parameters as needed.

In the query result, click on a slow or erroneous task to view trace details.

For the procedure to view the trace details, see [Call trace details](#).

Trace details

The trace details function enables you to query by TraceId the details of a specific service invocation trace in a selected region.

The trace details page displays the trace of the RPC service calls, not including local method calls.

The trace details function is used mainly for tracking the consumed time and occurred exceptions at each point of the distributed service calls. Local methods are not the core content of the calls, so it is recommended that you use logs to track the consumed time and occurred exceptions for local methods. For example, the trace details page will not display the local trace of `methodA()` calling `localMethodB()` and `localMethodC()`. Therefore, it could happen that the elapsed time on a parent node is longer than the total elapsed time on all subnodes.

You can log on to the **EDAS console** and Select **Microservice Management > Trace Details** in the left-side navigation pane to view the details of a service invocation trace. However, a more typical scenario is to view the trace details of the slow or erroneous services. The following example demonstrates how to view the trace details entering from **Trace Query** on the left-side menu bar.

In the trace query result, find the HSF method, DB request, or other RPC service call that consumes the longest time.

For DB, Redis, MQ, or other simple calls, find out the reason why accesses to these nodes are slow and check whether they are caused by slow SQL or network congestion.

For HSF methods, further analyze the reason why the method consumes so much time.

Confirm the time consumed by a local method.

Hover the cursor over the time bar on the method row, and in the displayed page, view the elapsed time for the client to send the request, the elapsed time for the server to process the request, and elapsed time for the client to receive the response.

If it takes a long time for the server to process the request, analyze the tasks. Otherwise, conduct the analysis using the method that is used for analyzing call timeout.

Check whether the total time consumed on subnodes is close to that consumed on the method.

If the time difference is small, it indicates that most of the time is consumed on network calls. In this case, reduce network calls as many as possible to shorten the time consumed on each method.

The preceding figure shows that the same method is cyclically called. Instead, it could be just called once in batch.

If the time difference is large, for example, the time consumed on the parent node is 607 ms while the total time consumed on the subnodes does not reach 100 ms. Then it indicates most of the time is consumed on the task logic of the server itself, rather than the RPC service call.

Locate the time-consuming call.

By looking at the time bars to first locate the call before which much time is consumed. The time is purely consumed by the local logic, for which further troubleshooting is required.

After locating the time-consuming logic, review the codes or add logs to the codes to locate the errors.

If it is found that the codes do not consume so much time, perform the following step.

Check whether GC occurred at that time. Therefore, the gc.log file is important.

Locate the timeout error.

An timeout error occurs. Perform the following steps to evaluate the time.

The time is divided into three parts:

Client sends request (0 ms): indicates the time duration from the client sends the request to the server receives it. This process includes serialization, network transmission, and deserialization. If this process takes a long time, consider if a consumer GC should be triggered. It will take a long time if the object for serialization or deserialization is large, the network is under great transmission pressure, or the provider GC occurs.

Requests processed on server (10,077 ms): indicates the time duration from the server receives the request to the server returns the response to the client. The time is taken only by the server to process the request, not including other operations.

Client receives response (3,002 ms): indicates the time duration from the server sends the response to the client receives it. As the timeout time of 3s is set, the server directly returns timeout after 3s, but the server is still processing the request. If this process consumes much time, perform troubleshooting using the same method that is used for the client.

Service topology

The service topology function is used to view the real-time (last second) call topology between applications in the system.

Log on to the EDAS console, and Select **Microservice Management** > **Service Topology** in the left-side navigation pane.

View the service topology.

The Service topology shows the real-time (last second) call topology between all applications under the current account.

Hover your cursor over an application to view the call topology for this application.

Click on an application to view its call topology and traffic data.

Traffic data refers to the current application's QPS, including:

Source traffic: The QPS for calls from other applications to this application.

Call traffic: The QPS for calls from this application to other applications.

Redis tracing

Function overview

After Redis tracing support is added, whenever applications access and perform operations on Redis, the process is recorded in EagleEye trace logs and EDAS collects, analyzes the statistics of the logs. Then information about Redis calls is displayed on the tracing and call analysis page of the EDAS platform.

Supported scope

Due to the wide range of Redis database variants and the usability of Spring Data, Redis trace support is only available for Spring Data Redis of 1.7.4.RELEASE. If you use any other database (for example, Jedis) than Spring Data Redis, you cannot view relevant information on the EagleEye trace interface (which is accessible from **Digital Operations** > **Trace Details** on the left-side menu bar of the EDAS console).

Note: If you use Spring Data Redis later than 1.7.4.RELEASE and the version does not support the provided functions, open a ticket to consult with us.

Usage instructions

For applications on the EDAS platform, Redis trace support replaces Spring Data Redis and is used in the same way as Spring Data Redis. For usage instructions on Spring Data Redis, see the user guide. At the code level, EDAS is compatible with Spring Data Redis 1.7.4-RELEASE. To enable Redis tracing support, follow these steps:

Open the {user.home}/.m2/settings.xml file to configure the local Maven repository.

```
<profile>
<id>edas.oss.repo</id>
<repositories>
<repository>
<id>edas-oss-central</id>
<name>taobao mirror central</name>
<url>http://edas-public.oss-cn-hangzhou.aliyuncs.com/repository</url>
<snapshots>
<enabled>true</enabled>
</snapshots>
<releases>
<enabled>true</enabled>
</releases>
</repository>
</repositories>
<pluginRepositories>
<pluginRepository>
<id>edas-oss-plugin-central</id>
<url>http://edas-public.oss-cn-hangzhou.aliyuncs.com/repository</url>
<snapshots>
<enabled>true</enabled>
</snapshots>
<releases>
<enabled>true</enabled>
</releases>
</pluginRepository>
</pluginRepositories>
</profile>
</profiles>
```

Activate the corresponding profile:

```
<activeProfiles>
<activeProfile>edas.oss.repo</activeProfile>
</activeProfiles>
```

Add dependency to the pom.xml file in the Maven project.

```
<dependency>
<groupId>com.alibaba.middleware</groupId>
<artifactId>spring-data-redis</artifactId>
<version>1.7.4.RELEASE</version>
</dependency>
```

Redis command support

The following tables list the Redis commands supported by Spring Data Redis and the support for EagleEye trace logs.

Key-type operations

Data structure/Object	Operation	Spring Data Redis method	EDAS support for EagleEye tracing (Y/N)	Remarks
Key	DEL	RedisOperations.delete	Y	
	DUMP	RedisOperations.dump	Y	
	EXISTS	RedisOperations.hasKey	Y	
	EXPIRE	RedisOperations.expire	Y	
	EXPIREAT	RedisOperations.expireAt	Y	
	KEYS	RedisOperations.keys	Y	
	MIGRATE	N		
	MOVE	RedisOperations.move	Y	
	OBJECT	N		
	PERSIST	RedisOperations.persist	Y	

	PEXPIRE	RedisOperation s.expire	Y	
	PEXPIREAT	RedisOperation s.expireAt	Y	
	PTTL	RedisOperation s.getExpire	Y	
	RANDOMKEY	RedisOperation s.randomKey	Y	
	RENAME	RedisOperation s.rename	Y	key: oldKey : \${oldKey};newKey:\${newKey}
	RENAMENX	RedisOperation s.renameIfAbsent	Y	
	RESTORE	RedisOperation s.restore	Y	
	SORT	RedisKeyComm ands.sort	Y	key: query:\${SortQuery}
	TTL	RedisOperation s.getExpire	Y	
	TYPE	RedisOperation s.type	Y	
	SCAN	RedisKeyComm ands.scan	N	

String-type operations

Data structure/Objec t	Operation	Spring Data Redis method	EDAS support for EagleEye tracing (Y/N)	Remarks
String	APPEND	ValueOperation s.append	Y	
	BITCOUNT	N		
	BITOP	N		
	BITFIELD	N		
	DECR	ValueOperation s.increment	Y	
	DECRBY	ValueOperation s.increment	Y	
	GET	ValueOperation s.get	Y	

	GETBIT	ValueOperation s.getBit	Y	
	GETRANGE	ValueOperation s.get	Y	
	GETSET	ValueOperation s.getAndSet	Y	
	INCR	ValueOperation s.increment	Y	
	INCRBY	ValueOperation s.increment	Y	
	INCRBYFLOAT	ValueOperation s.increment	Y	
	MGET	ValueOperation s.multiGet	Y	
	MSET	ValueOperation s.multiSet	Y	
	MSETNX	ValueOperation s.multiSetIfAbs ent	Y	
	PSETEX	ValueOperation s.set	Y	
	SET	ValueOperation s.set	Y	
	SETBIT	ValueOperation s.setBit	Y	
	SETEX	ValueOperation s.set	Y	
	SETNX	ValueOperation s.setIfAbsent	Y	
	SETRANGE	ValueOperation s.set	Y	
	STRLEN	ValueOperation s.size	Y	

Hash-type operations

Data structure/Objec t	Operation	Spring Data Redis method	EDAS support for EagleEye tracing (Y/N)	Remarks
Hash	HDEL	HashOperation s.delete	Y	
	HEXISTS	HashOperation s.hasKey	Y	

	HGET	HashOperation s.get	Y	
	HGETALL	HashOperation s.entries	Y	
	HINCRBY	HashOperation s.increment	Y	
	HINCRBYFLOAT	HashOperation s.increment	Y	
	HKEYS	HashOperation s.keys	Y	
	HLEN	HashOperation s.size	Y	
	HMGET	HashOperation s.multiGet	Y	
	HMSET	HashOperation s.putAll	Y	
	HSET	HashOperation s.put	Y	
	HSETNX	HashOperation s.putIfAbsent	Y	
	HVALS	HashOperation s.values	Y	
	HSCAN	HashOperation s.scan	Y	
	HSTRLEN	N		

List-type operations

Data structure/Objec t	Operation	Spring Data Redis method	EDAS support for EagleEye tracing (Y/N)	Remarks
List	BLPOP	ListOperations.l eftPop	Y	
	BRPOP	ListOperations.r ightPop	Y	
	BRPOPLPUSH	ListOperations.r ightPopAndLeft Push	Y	key: sourceKey:\${so urceKey};destK ey:\${destKey}
	LINDEX	ListOperations.i ndex	Y	
	LINSERT	ListOperations.l eftPush	Y	

	LLEN	ListOperations.size	Y	
	LPOP	ListOperations.leftPop	Y	
	LPUSH	ListOperations.leftPush	Y	
	LPUSHX	ListOperations.leftPushIfPresent	Y	
	LRANGE	ListOperations.range	Y	
	LREM	ListOperations.remove	Y	
	LSET	ListOperations.set	Y	
	LTRIM	ListOperations.trim	Y	
	RPOP	ListOperations.rightPop	Y	
	RPOPLPUSH	ListOperations.rightPopAndLeftPush	Y	key: sourceKey:\${sourceKey};destKey:\${destKey}
	RPUSH	ListOperations.rightPush	Y	
	RPUSHX	ListOperations.rightPushIfPresent	Y	

Set-type operations

Data structure/Object	Operation	Spring Data Redis method	EDAS support for EagleEye tracing (Y/N)	Remarks
Set	SADD	SetOperations.add	Y	
	SCARD	SetOperations.size	Y	
	SDIFF	SetOperations.difference	Y	
	SDIFFSTORE	SetOperations.differenceAndStore	Y	
	SINTER	SetOperations.in	Y	

		tersect		
	SINTERSTORE	SetOperations.in tersectAndStore	Y	
	SISMEMBER	SetOperations.is Member	Y	
	SMEMBERS	SetOperations.m embers	Y	
	SMOVE	SetOperations.m ove	Y	
	SPOP	SetOperations.p op	Y	
	SRANDMEMBER	SetOperations.ra ndomMember randomMembers distinctRandom Members	Y	
	SREM	SetOperations.re move	Y	
	SUNION	SetOperations.u nion	Y	
	SUNIONSTORE	SetOperations.u nionAndStore	Y	
	SSCAN	SetOperations.sc an	Y	

SortedSet-type operations

Data structure/Object	Operation	Spring Data Redis method	EDAS support for EagleEye tracing (Y/N)	Remarks
SortedSet	ZADD	ZSetOperations .add	Y	
	ZCARD	ZSetOperations .size/zCard	Y	
	ZCOUNT	ZSetOperations .count	Y	
	ZINCRBY	ZSetOperations .incrementScore	Y	
	ZRANGE	ZSetOperations .range rangeWithScores	Y	

	ZRANGEBYSCORE	ZSetOperations .rangeByScore rangeByScoreWithScores	Y	
	ZRANK	ZSetOperations .rank	Y	
	ZREM	ZSetOperations .remove	Y	
	ZREMRANGEBYRANK	ZSetOperations .removeRange	Y	
	ZREMRANGEBYSCORE	ZSetOperations .removeRangeByScore	Y	
	ZREVRANGE	ZSetOperations .reverseRange reverseRangeWithScores	Y	
	ZREVRANGEBYSCORE	ZSetOperations .reverseRangeByScore reverseRangeByScoreWithScores	Y	
	ZREVRANK	ZSetOperations .reverseRank	Y	
	ZSCORE	ZSetOperations .score	Y	
	ZUNIONSTORE	ZSetOperations .unionAndStore	Y	
	ZINTERSTORE	ZSetOperations .intersectAndStore	Y	
	ZSCAN	ZSetOperations .scan	Y	
	ZRANGEBYLEX	ZSetOperations .rangeByLex	Y	
	ZLEXCOUNT	N		
	ZREMRANGEBYLEX	N		

HyperLogLog operations

Data structure/Object	Operation	Spring Data Redis method	EDAS support for EagleEye tracing (Y/N)	Remarks
-----------------------	-----------	--------------------------	---	---------

HyperLogLog	PFADD	HyperLogLogOperations.add	Y	
	PFCOUNT	HyperLogLogOperations.size	Y	
	PFMERGE	HyperLogLogOperations.union	Y	key: dest:\${destination}

Pub/Sub (publish/subscribe) operations

Data structure/Object	Operation	Spring Data Redis method	EDAS support for EagleEye tracing (Y/N)	Remarks
Pub/Sub	PSUBSCRIBE	N		
	PUBLISH	RedisOperations.convertAndSend	Y	key: msg:\${msg}
	PUBSUB	RedisMessageListenerContainer. setMessageListeners addMessageListener	N	
	PUNSUBSCRIBE	N		
	UNSUBSCRIBE	N		

Transaction operations

Data structure/Object	Operation	Spring Data Redis method	EDAS support for EagleEye tracing (Y/N)	Remarks
Transaction	DISCARD	RedisOperations.discard	Y	
	EXEC	RedisOperations.exec	Y	key: execRaw
	MULTI	RedisOperations.multi	Y	
	UNWATCH	RedisOperations.unwatch	Y	
	WATCH	RedisOperations.watch	Y	

Script operations

Data structure/Object	Operation	Spring Data Redis method	EDAS support for EagleEye tracing (Y/N)	Remarks
Script	EVAL	ScriptExecutor.execute	Y	key: Null
	EVALSHA	ScriptExecutor.execute	Y	key: Null
	SCRIPT EXISTS	RedisScriptingCommands.scriptExists	N	
	SCRIPT FLUSH	RedisScriptingCommands.scriptFlush	N	
	SCRIPT KILL	RedisScriptingCommands.scriptKill	N	
	SCRIPT LOAD	RedisScriptingCommands.scriptLoad	N	

Component Center

Scheduler

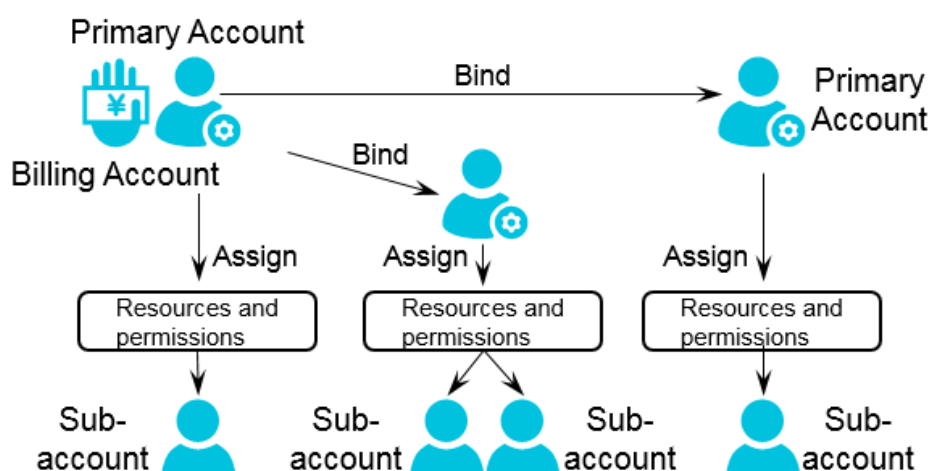
System management

Account system introduction

EDAS provides a comprehensive primary and sub-account management system to help you achieve enterprise-level account management and improve enterprise information security. A primary account can assign permissions and resources to multiple sub-accounts on demand in accordance with the minimum permission principle, which lowers the risks of enterprise information security and reduces the load of the primary account.

Before adopting the account system of Alibaba Cloud Resource Access Management (RAM), EDAS is developed with a strict primary and sub-account system to implement separation between users and permissions. After being upgraded on July 2016, EDAS also supports the RAM primary and subaccount system.

The following figure shows the EDAS account system.



The billing account is a primary account used to buy EDAS service. If multiple departments of an enterprise need to use EDAS, a user can create a billing account to buy the EDAS product and then binds it with multiple primary accounts to give other primary account users access to EDAS. This helps customers maximize their benefits.

Note: A billing account can be bound with a maximum of five primary accounts.

Billing account and primary account:

All billing accounts are primary accounts, but not all primary accounts are billing accounts.

Each primary account is an independent account that owns all resources bought with the account and has full permissions on EDAS except that it cannot bind other primary accounts.

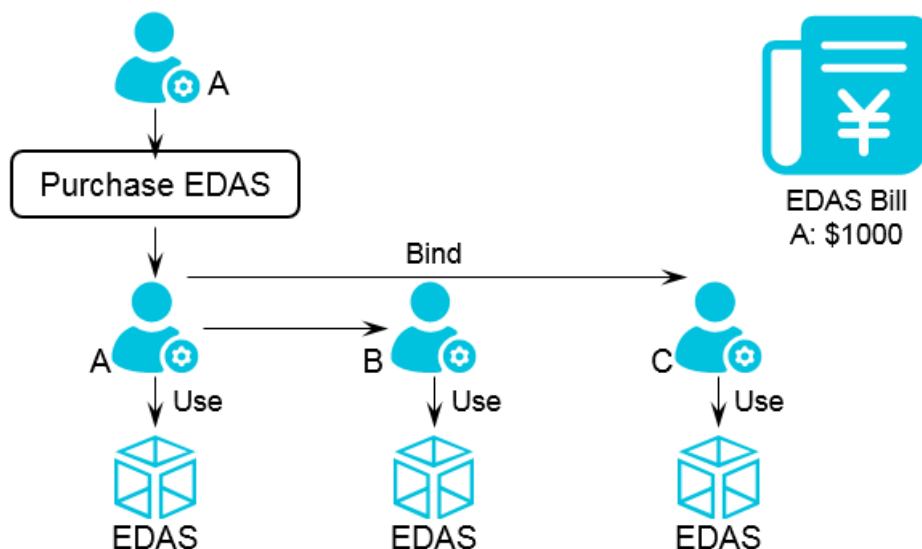
A billing account and a primary account are two independent accounts of Alibaba Cloud. The payment binding relationship between billing account and primary account is only effective for the purpose of EDAS purchase. The billing account cannot be used to buy any other resources than EDAS on behalf of the primary account. A primary account should still buy resources such as ECS and SLB by itself even if it is bound to a billing account for EDAS.

purchase. (For details about specific resources, see [Resource management](#).)

The following describes three use cases of the EDAS account system.

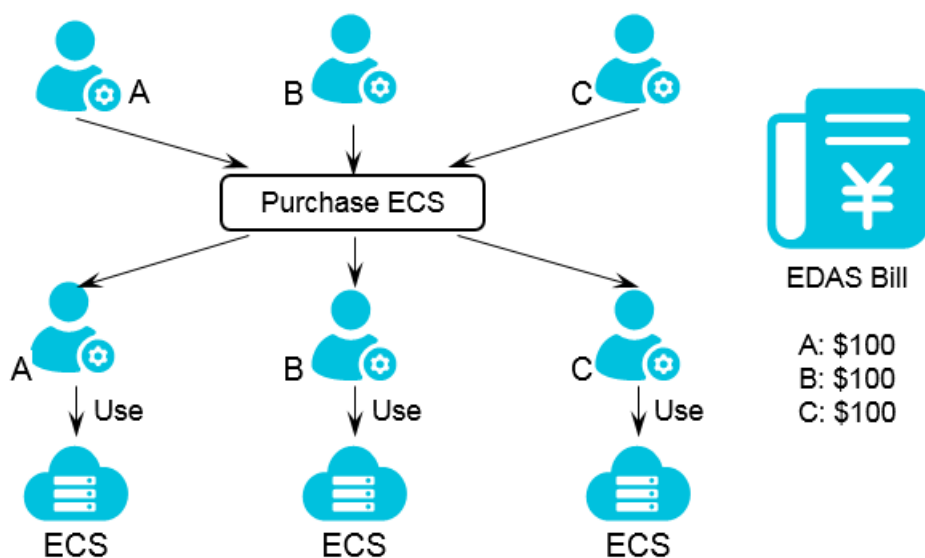
Scenario 1

A company uses Account A to buy EDAS. Account A is a billing account and also a primary account. The company binds this billing account with the primary accounts (Account B and Account C) of two departments to enable the departments to access EDAS without purchasing EDAS again. See the following figure.



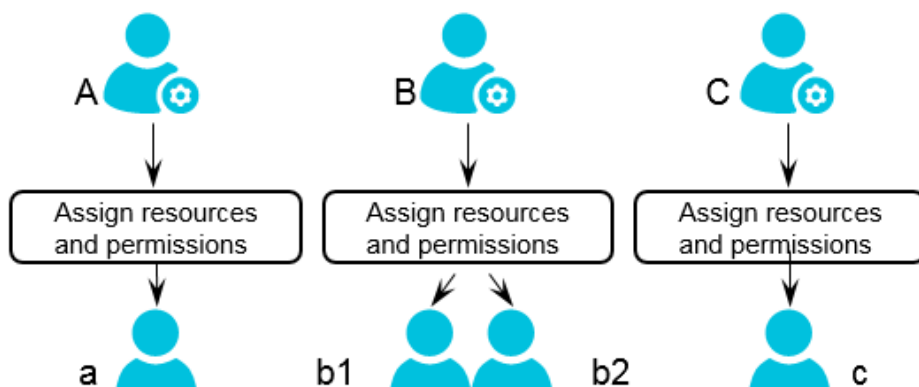
Scenario 2

If users of Account B and Account C require the full functions of EDAS, for example, to create or run applications, the two accounts rather than Account A must be used to buy resources such as ECS, as shown in the following figure.



Scenario 3

After resources are prepared, sub-accounts are created under the three primary accounts and used to allocate and manage permissions and resources. Sub-account a is created under Account A and assigned all ECS resources and permissions. Two roles, application administrator and operation administrator, are created under Account B and allocated to Sub-account b1 and Sub-account b2, respectively. A role for application query is created under Account C and allocated to Sub-account c.



Primary accounts

In EDAS, a primary account owns all resources under the account and has full operation permissions on EDAS. The primary account used to buy the EDAS product is also the billing account. After a company buys the EDAS service, it can bind the billing account with other primary accounts of the company to give the primary account users access to EDAS without secondary purchase. This helps customers lower resource costs.

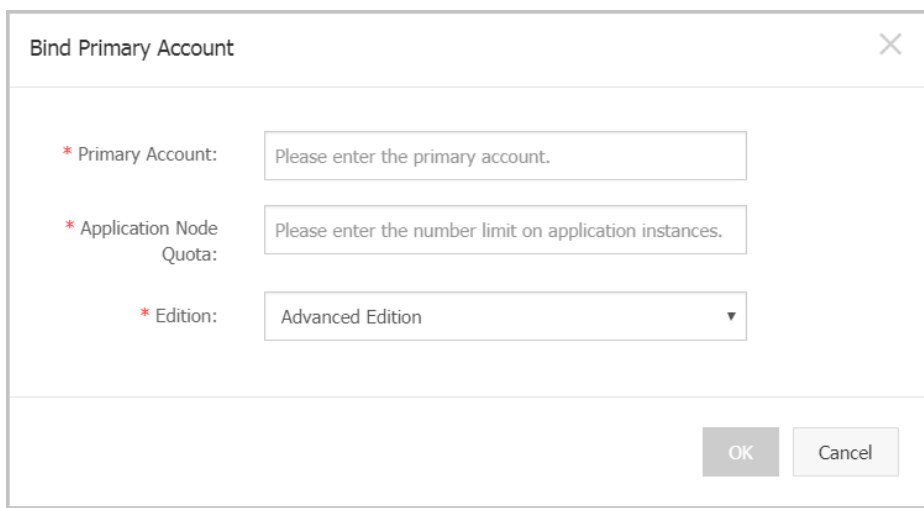
Bind a primary account

To bind a primary account, follow these steps:

In the EDAS console, select **Accounts** > **Primary Accounts** in the left-side navigation pane.

Click **Bind Primary Account** in the upper-right corner.

Enter a primary account name, set the maximum number of applications allowed for this account, select a product edition, and click **OK**.



The dialog box titled "Bind Primary Account" contains three required fields, each marked with a red asterisk:

- * Primary Account:** A text input field with the placeholder text "Please enter the primary account."
- * Application Node Quota:** A text input field with the placeholder text "Please enter the number limit on application instances."
- * Edition:** A dropdown menu currently showing "Advanced Edition".

At the bottom right of the dialog are "OK" and "Cancel" buttons.

Primary Account: It must be a valid Alibaba Cloud account that has never been used to buy the EDAS service.

Application Quota: Maximum number of applications that can be owned by the primary account and its sub-accounts. When the billing account allocates a quota to each primary account, the sum of the quotas of all primary accounts bound to it cannot be greater than the total application quota of the billing account.

Product Edition: The product edition that the billing account allocates to each bound primary account must be the same as that of the billing account.

Note:

- A billing account can be bound with a maximum of five primary accounts.
- Open a ticket to unbind primary accounts from the billing account, .

For bound primary accounts:

- The number of actual applications you have should not exceed your application instance quota.

The **actual number of applications** of the **billing account** plus the sum of the application **quotas** of **other primary accounts** bound to the billing account should not exceed the application quota of the billing account.

Primary Accounts

Bind Primary Account

Instances with Applications Deployed: 125

Primary Account	Application Node Quota	Instances with Applications Deployed	Product Series
edas_test1@aliyun-test.com (Billing Account)	1000	125	Professional Edition
dd101616	1	0	Professional Edition

Note: One billing account can bind 5 primary accounts. If you need to unbind an account, please open a ticket.

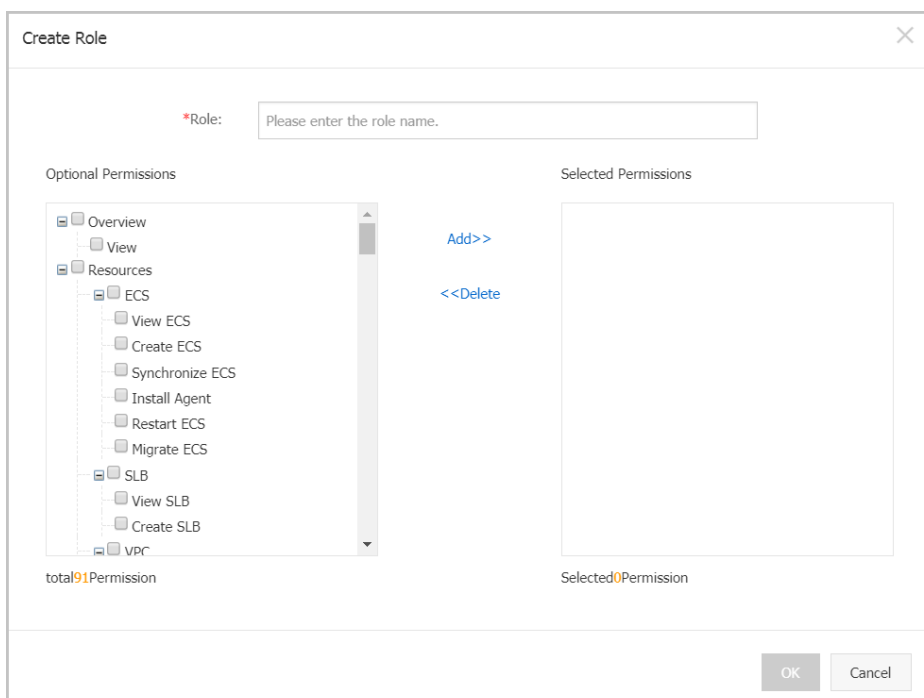
Manage roles

A primary account can define different operation permissions for its sub-accounts by creating different roles.

In the EDAS console, select **Roles** in the left-side navigation pane.

Click **Create Role** in the upper-right corner.

Enter a role name, select the permissions from the left-side field and add to the right, and click **OK**.



You can view, manage Permissions or Delete roles on the **Roles** page.

View all permissions

You can view all permissions of the EDAS system in the console.

In the EDAS console, select **Accounts > All Permissions** in the left-side navigation pane, and unfold the lists to display specific information.

All Permissions	
Permission	Description
+ Overview	Overview
+ Resources	Resources
+ Applications	Applications
+ Services	Services
+ Global Configuration	Global Configuration
+ Application Configuration Management	Application Configuration Management
+ Continuous Integration	Continuous Integration
+ Service Groups	Service Groups
+ Digital Operations	Digital Operations

Sub-accounts

EDAS supports the account system of Alibaba Cloud Resource Access Management (RAM). You can create RAM sub-accounts under your primary account to avoid sharing your account key with other users. You can also assign minimum permissions to sub-accounts as needed to separate responsibilities and conduct efficient enterprise management. This topic introduces the following subjects:

- Introduction to RAM sub-accounts
- Create a RAM sub-account
- Use a RAM sub-account for EDAS logon
- Authorize a RAM account
- Unbind a RAM account

Introduction to RAM sub-accounts

When you use your primary account in EDAS, you can allocate different roles and resources to the sub-accounts under the primary account so as to complete different types of jobs with different user identities, such as application administrator (with the permissions to create, start, query, and delete applications) and operation administrator (with the permissions to view resources, check application monitoring, and manage alarm policies, throttling and degradation rules). The primary and sub-account permission mode is similar to the classification of system user and common user in Linux. A system user can grant and revoke permissions to/from common users.

RAM sub-accounts:

- A RAM account is created by a primary account in the RAM system. Validity check is not required for the RAM account, but the account name must be unique under the primary account.

- A dedicated logon portal is available for RAM accounts. For details about the logon portal, see relevant description in the RAM console.

Create a RAM sub-account

Follow these steps:

In the EDAS console, choose **Accounts** > **Sub-Account** in the left-side navigation pane.

Click **Bind Sub-Account** in the upper-right corner to go to the RAM console. After you create a RAM user in the RAM console, by default, the RAM user is also a sub-account under your primary account in EDAS.

Click **Users** in the RAM console.

Click **Create User** in the upper-right corner. In the **Create User** dialog box that appears, enter your logon name and other information, and click **OK**. The user management page shows a new username, indicating that a RAM user is successfully created.

Note: The logon name must be unique under the primary account.

Click the **logon name/displayed name** link of the RAM user to go to the user information page.

Click **Enable Console Logon** in **Web Console Logon Management**. The password setting dialog box appears.

1. Enter a **new password**, and select **Require to reset the password upon next logon** as needed.

After the preceding steps are complete, a RAM user with the console logon permission is successfully created.

Use a RAM sub-account for EDAS logon

Following these steps:

Click the **RAM User Logon Link** on the **Dashboard** page of the RAM console.

Note: The RAM user logon link varies depending on different primary accounts.

Enter the sub-account name and password on the RAM user logon page, and click **Logon** to go to the RAM console.

Note:

Enterprise Alias: Already exists in the logon link of the sub-account.

Sub-account Name: Logon name that the primary account sets when creating the RAM user.

Sub-account Password: Password that the primary account sets when enabling console logon for the RAM account. If **Require to reset the password upon next logon** is selected, the RAM account is required to reset the password after initial logon to the console. The new password is used for future logons.

Click **Products & Services** in the top navigation bar of the RAM console, and click **Enterprise Distributed Application Service (EDAS)** under the Middleware category to go to the EDAS console.

Authorize a RAM account

Two authorization methods are available:

- RAM authorization
- EDAS authorization

The two authorization methods are mutually exclusive. After you authorize a RAM account in RAM, you cannot authorize the same account in EDAS. You must revoke RAM authorization in the RAM console before you can perform EDAS authorization in the EDAS console. To authorize a RAM account in EDAS, ensure that the account is not authorized in RAM.

RAM authorization is performed at the EDAS service level, indicating that a RAM-authorized account has full permissions on EDAS. RAM authorization and revocation must be performed in the RAM console.

The procedure of RAM authorization is as follows:

In the RAM console, click **Users** in the left-side navigation pane, select the user to be authorized, and click **Authorize** in the "Action" field on the right.

Enter **EDAS** in the search box in the left part of the dialog box, select **AliyunEDASFullAccess** and add this option to **Selected Authorization Policy Name** on the right, and click **OK** to grant full EDAS permissions to the account.

After authorization is complete, use the primary account to log on to EDAS and select **Accounts > Sub-Accounts** on the left-side menu bar. Then the page lists the permissions, resources, and applications granted to the RAM account. The authorization function is disabled for the RAM account in the EDAS console.

To revoke RAM authorization, follow these steps:

In the RAM console, click **Users** in the left-side navigation pane, select the user, and click **Authorize** in the **Actions** field.

Move the **AliyunEDASFullAccess** option in the right-side field to the left and click **OK**.

After authorization is revoked, use the primary account to log on to EDAS and select **Accounts > Sub-Accounts** in the left-side navigation pane. Then the page shows that all resources and permissions of the RAM account are revoked. The authorization function is enabled for the RAM account on EDAS.

RAM users that are not authorized for EDAS can manage roles and resource groups, authorize applications, and perform other operations in EDAS, but cannot perform the unbind operation. Supported operations:

Manage roles

A primary account can assign a role to a sub-account to grant the role-specific permissions to this sub-account. The procedure of role management is as follows:

In the EDAS console, choose **Accounts > Sub-Accounts** on the left-side menu bar. Find the sub-account to be authorized and select **Manage Roles** in the "Actions" field on the right.

Select roles on the left side and click > to add the roles to the right side, then click **OK**.

Select **Accounts > Roles** on the left-side menu bar. The role name is displayed in the **Roles** page.

Authorize an application

A primary account can assign an application to a sub-account to grant the application ownership to this sub-account. The procedure of application authorization is the same as that of role management.

Note: Application authorization only grants the application ownership to the sub-account. To

grant application operation permissions (to start or delete the application, for example) to the sub-account, you need to assign a role to the sub-account. Therefore, application authorization is typically followed by role authorization.

Authorize a resource group

A primary account can assign a resource group to a sub-account to give the sub-account access to resources in the resource group. For details about the concept of resource group, see [Resource group](#). The procedure of resource group authorization is the same as that of role management.

Unbind a RAM account

Follow these steps:

Log on to the RAM console.

Click **Users** in the left-side navigation pane, find the account to be unbound, and click **Delete** in the “Actions” field on the right.