

Alibaba Cloud Enterprise Distributed Application Service

k8s User Guide

Issue: 20200618









Legal disclaimer

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

- 1.** You shall download and obtain this document from the Alibaba Cloud website or other Alibaba Cloud-authorized channels, and use this document for your own legal business activities only. The content of this document is considered confidential information of Alibaba Cloud. You shall strictly abide by the confidentiality obligations. No part of this document shall be disclosed or provided to any third party for use without the prior written consent of Alibaba Cloud.
- 2.** No part of this document shall be excerpted, translated, reproduced, transmitted, or disseminated by any organization, company, or individual in any form or by any means without the prior written consent of Alibaba Cloud.
- 3.** The content of this document may be changed due to product version upgrades, adjustments, or other reasons. Alibaba Cloud reserves the right to modify the content of this document without notice and the updated versions of this document will be occasionally released through Alibaba Cloud-authorized channels. You shall pay attention to the version changes of this document as they occur and download and obtain the most up-to-date version of this document from Alibaba Cloud-authorized channels.
- 4.** This document serves only as a reference guide for your use of Alibaba Cloud products and services. Alibaba Cloud provides the document in the context that Alibaba Cloud products and services are provided on an "as is", "with all faults" and "as available" basis. Alibaba Cloud makes every effort to provide relevant operational guidance based on existing technologies. However, Alibaba Cloud hereby makes a clear statement that it in no way guarantees the accuracy, integrity, applicability, and reliability of the content of this document, either explicitly or implicitly. Alibaba Cloud shall not bear any liability for any errors or financial losses incurred by any organizations, companies, or individuals arising from their download, use, or trust in this document. Alibaba Cloud shall not, under any circumstances, bear responsibility for any indirect, consequential, exemplary, incidental, special, or punitive damages, including lost profits arising from the use or trust in this document, even if Alibaba Cloud has been notified of the possibility of such a loss.

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

Document conventions

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

Style	Description	Example
{ } or {a b}	This format is used for a required value, where only one item can be selected.	switch {active stand}

Contents

Legal disclaimer	I
Document conventions	I
1 Comparison between the new and previous versions of console	1
2 Upgrade application management	4
3 Create and import Container Service Kubernetes clusters	6
4 Deploy application	8
4.1 Create Docker images for applications.....	8
4.2 Application deployment overview (Kubernetes clusters).....	16
4.3 Application Deployments (Console).....	17
4.3.1 Use an image to deploy an application in a Container Service Kubernetes cluster (new version).....	17
4.3.2 Use a JAR or WAR package to deploy an application (new version).....	25
4.4 Deploy applications with tools.....	33
4.5 Canary release (Kubernetes clusters).....	33
4.6 Batch release (applicable to Kubernetes clusters).....	37
5 User Guide (Old Version)	41
5.1 Application monitoring.....	41
5.1.1 Application overview.....	41
5.1.2 Application details.....	42
5.1.2.1 JVM monitoring.....	42
5.1.2.2 Host monitoring.....	44
5.1.3 API call monitoring.....	46
5.1.4 Advanced monitoring.....	47
5.1.5 Alerting.....	47
5.1.5.1 Create an alert.....	47
5.1.5.2 Manage alerts.....	52
5.1.5.3 Create contacts.....	53
5.1.5.4 Create contact groups.....	54
5.1.5.5 Enable DingTalk robot alerts.....	56
5.2 Log diagnosis.....	56

1 Comparison between the new and previous versions of console

We upgraded the underlying architecture of Enterprise Distributed Application Service (EDAS) at 24:00 on February 20, 2020. EDAS provides lightweight O&M capabilities for applications in Kubernetes clusters that were imported after this time but has no function changes for applications in Kubernetes clusters that were imported earlier. Read this topic when performing application management.

Scenarios

- New version: For applications in Kubernetes clusters that were imported after 24:00 on February 20, 2020, EDAS provides the new Application Management page. For more information, see [User guide \(new\)](#).
- Previous version: For applications in Kubernetes clusters that were imported before 24:00 on February 20, 2020, EDAS provides the old Application Details page for management. For more information, see [User guide \(previous\)](#).



Note:

After the architecture of EDAS Kubernetes cluster was automatically updated at 24:00 on February 20, 2020, the applications created in the Kubernetes cluster are provided with the new Application Management page for management. We will provide the **One-click Upgrade** function for you to upgrade the management page for existing applications in Kubernetes clusters that were imported before this time.

Benefits

The new application management feature provides one-stop lightweight O&M capabilities. When a Kubernetes cluster is imported to EDAS, EDAS monitors the cluster and its applications in real time, such as the service details, instance details, and change records. EDAS also provides operation reports and corresponding solutions in real time.

For example, the Change List. In the new version, EDAS displays the real-time information of application metrics during a change process, such as the average number of errors, response time (RT), and the number of single-instance requests.

Feature comparison

Feature	New version	Old version	Difference
Application overview	Enhanced	Supported	Compared with the previous version, the new application overview contains not only the basic information but also the diagnosis report, solution, overview, and topology of the application. For more information, see #unique_4 .
Lifecycle management	Supported	Supported	The same as the previous version
Scale-in and scale-out	Enhanced	Supported	Compared with the previous version, manual scaling and automatic scaling are now next to each other, which is easy to operate. For more information, see #unique_6 .
Server Load Balancer (SLB)	Enhanced	Supported	Compared with the previous version, you can add HTTPS listeners to SLB instances. For more information, see #unique_7 .
Application changes	Enhanced	Supported	Compared with the previous version, the new version adds metrics monitoring during a change process, such as application metrics monitoring, exception monitoring, system monitoring, and pod execution logs. For more information, see #unique_8
Instance details	Supported	Supported	The same as the previous version
Service details	Supported	Supported	The same as the previous version
Application diagnosis	New	Not supported	Compared with the previous version, the new version adds real-time diagnosis and thread analysis. For more information, see Application diagnosis .
Application monitoring	Supported	Supported	Compared with the previous version, the new version focuses on one-stop real-time monitoring to simplify O&M.
Alerting	Supported	Supported	The same as the previous version
Log management	Supported	Supported	The same as the previous version

Feature	New version	Old version	Difference
Events	Enhanced	Supported	Compared with the previous version, the new version displays more event information, such as application alerts and diagnosis reports. For more information, see View application events .
Microservice governance	Supported	Supported	The same as the previous version

2 Upgrade application management

If you want to use the new version of all-in-one lightweight application management, you must upgrade the old application management version. This topic describes how to upgrade application management from an earlier version to the new version in one-click upgrade or silent mode.

One-click upgrade

Enterprise Distributed Application Service (EDAS) provides all-in-one lightweight application management capabilities for you to easily upgrade application management.

1. Log on to the [EDAS console](#).
2. In the left-side navigation pane, choose **Resources > Clusters**. On the Cluster Details page, click the application name in **Applications**.

You can also choose **Application Management > Applications** from the left-side navigation pane, and then click the name of the Container Service Kubernetes application on the **Applications** page.

3. On the **Basic Information** page, click **New Version** in the upper-right corner.
4. In the **Upgrade New Application Management** dialog box, select **Redeploy Application (Release for All Instances, Rolling Upgrade)** and immediately upgrade the new application management. Then click **Upgrade New Version Now (Redeploy)**.

After the deployment is complete, your Application Management page is upgraded to the new version. For more information, see [User guide \(new version\)](#).

If you are used to the old version, click **Old Version** in the upper-left corner of the **Basic Information** page for your application.



Note:

- After the Application Management page is upgraded, if you need to configure a Server Load Balancer (SLB) instance for your application, you must configure this instance on the upgraded Application Management page. Do not return to the old version to configure the SLB instance.
- If you have configured an SLB instance before the Application Management page is upgraded, the original SLB instance configuration is synchronized to the new version after the upgrade.

Silent upgrade

In addition to the one-click upgrade, EDAS also supports silent upgrade (applicable to application deployment for instances all at once or in batches) to upgrade the Application Management page. For more information, see [Batch release \(applicable to Kubernetes clusters\)](#). After your application is deployed, the new Application Management page is available to you.

3 Create and import Container Service Kubernetes clusters

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

Prerequisites

- You have [#unique_14](#).
- You have activated Container Service for Kubernetes and completed [#unique_15](#).



Note:

We recommend that you use the same Alibaba Cloud account or RAM user account for Container Service and EDAS. If you use different RAM user accounts, make sure they belong to the same Alibaba Cloud account and have been authorized.

Step 1: Create Container Service Kubernetes clusters in the Container Service for Kubernetes console

Log on to the [Container Service console](#), [#unique_16](#)

Step 2: Import Container Service Kubernetes clusters into the EDAS console

1. Log on to the [EDAS console](#).
2. In the left-side navigation pane, choose **Resources > Clusters**.
3. On the **Clusters** page, click **Container Service K8s Cluster**. In the cluster list, find 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 been imported successfully to EDAS.

View and manage Container Service Kubernetes clusters

Click the cluster ID of the target Container Service Kubernetes cluster. On the Cluster Details page, you can view details of the cluster, ECS instances, applications, and service mesh installation information.

- **Cluster Information:** this section displays the basic information of the cluster, including the cluster ID, csClusterID, cluster name, namespace, cluster type, Virtual Private Cloud (VPC) VPC ID, network type, cluster status, and cluster description.
- **ECS instance:** this section displays the ECS instances in the cluster, overview of the ECS instances, and actions that you can perform on the ECS instances, such as removal, and billing method change.
 - You can select the check boxes on the left of the ECS instances to be removed, click **Bulk removal**, and then follow the instructions to complete the subsequent operations.
 - You can select the check boxes on the left of the ECS instances to be subcontracted and click **Pay-per-volume subcontracting year-month**, then follow the instructions to complete the subsequent operations.
- **Applications:** this section lists the applications in a cluster. You can view the application name, JDK version, application runtime environment, total number of instances, number of running instances, and application owner. You can click the name of an application to go to the Application Details page.

4 Deploy application

4.1 Create Docker images for applications

You can use commands in local development tools to package an application into a WAR or a JAR package for direct deployment. Alternatively, you can create an image based on a WAR or a JAR package, and then upload the image to the Alibaba Cloud image repository for deployment. This topic describes how to create the Dockerfiles for images of applications with different frameworks and how to upload the images to the Alibaba Cloud image repository.

Prerequisites

Before creating an application image, read [Regulations for creating an image](#). Follow the instructions and steps below to create an image for an Enterprise Distributed Application Service (EDAS) application.

Create a standard Dockerfile

A [Dockerfile](#) is a configuration file in text format. You can use the Dockerfile to quickly create an image.

You can use the Dockerfile to create images for High-speed Service Framework (HSF), Spring Cloud, or Dubbo applications. The following examples describe how to create Dockerfiles for applications with different frameworks.

An EDAS standard [Dockerfile](#) describes all instructions for creating application runtime environments in EDAS, including downloading, installing, and starting OpenJDK, Tomcat, and the WAR and the JAR packages. You can modify the Dockerfile to replace the OpenJDK version, modify the Tomcat configuration, and change the runtime environment. For more information, see [Customize the Dockerfile](#).

The following examples show how to create Dockerfiles for images of applications with different frameworks. Note that the CATALINA_OPTS parameter needs to be included in the Dockerfile of a JAR package application. Otherwise, errors might occur in application service registration and configuration reading.

- [Dockerfile example of an HSF application \(based on a WAR package\)](#)
- [Dockerfile example of HSF applications \(based on a JAR package\)](#)
- [Dockerfile example of Spring Cloud or Dubbo applications \(based on a WAR package\)](#)

- [Dockerfile example of Spring Cloud or Dubbo applications \(based on a JAR package\)](#)

Dockerfile example of an HSF application (based on a WAR package)

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

# Install the required software for packaging.
RUN yum install -y wget unzip telnet lsof net-tools bind-utils

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

# Set the version of EDAS Container or Pandora.
ENV EDAS_CONTAINER_VERSION V3.5.4
LABEL pandora V3.5.4

# Download and install OpenJDK.
RUN yum -y install java-1.8.0-openjdk-devel

# Create a JAVA_HOME symbolic link.
RUN if [ ! -L "${JAVA_HOME}" ]; then mkdir -p 'dirname ${JAVA_HOME}' && ln -s 'readlink -f /usr/lib/jvm/java' ${JAVA_HOME}; fi

# Download and install Ali-Tomcat 7.0.92 to the directory: /home/admin/taobao-tomcat
RUN wget http://edas-hz.oss-cn-hangzhou.aliyuncs.com/edas-container/7.0.92/taobao-tomcat-production-7.0.92.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 of the corresponding version 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 WAR package of the EDAS application demo.
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 the run mode, and the catalina log is displayed in the standard command line.
WORKDIR ${CATALINA_HOME}
CMD ["catalina.sh", "run"]
```

Dockerfile example of HSF applications (based on a JAR package)

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

# Install the required software for packaging.
RUN yum install -y wget unzip telnet lsof net-tools bind-utils
```

```

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

# Set the version of EDAS Container or Pandora.
ENV EDAS_CONTAINER_VERSION V3.5.4
LABEL pandora V3.5.4

# Download and install OpenJDK.
RUN yum -y install java-1.8.0-openjdk-devel

# Create a JAVA_HOME symbolic link.
RUN if [ ! -L "${JAVA_HOME}" ]; then mkdir -p 'dirname ${JAVA_HOME}' && ln -s 'readlink -f /usr/lib/jvm/java' ${JAVA_HOME}; fi

# Download and install EDAS Container or Pandora of the corresponding version 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 JAR package of the EDAS application demo.
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=${CATALINA_H
OME}/deploy/taobao-hsf.sar "${ADMIN_HOME}/app/provider.jar" --server.context-path
=/ --server.port=8080 --server.tomcat.uri-encoding=ISO-8859-1 --server.tomcat.max-
threads=400' > ${ADMIN_HOME}/start.sh && chmod +x ${ADMIN_HOME}/start.sh

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

```

Dockerfile example of Spring Cloud or Dubbo applications (based on a WAR package)

```

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

# Install the required software for packaging.
RUN yum install -y wget unzip telnet lsof net-tools bind-utils

# Prepare the JDK or Tomcat system variables.
ENV JAVA_HOME /usr/java/latest
ENV CATALINA_HOME /home/admin/apache-tomcat-7.0.91
ENV ADMIN_HOME /home/admin
ENV PATH ${JAVA_HOME}/bin:${CATALINA_HOME}/bin:${PATH}

RUN mkdir -p ${ADMIN_HOME}

# Download and install OpenJDK.
RUN yum -y install java-1.8.0-openjdk-devel

# Create a JAVA_HOME symbolic link.
RUN if [ ! -L "${JAVA_HOME}" ]; then mkdir -p 'dirname ${JAVA_HOME}' && ln -s 'readlink -f /usr/lib/jvm/java' ${JAVA_HOME}; fi

```

```

# Download and install Tomcat.
RUN wget http://edas-hz.oss-cn-hangzhou.aliyuncs.com/apache-tomcat-7.0.91.tar.gz -
O /tmp/apache-tomcat-7.0.91.tar.gz && \
  tar -xvf /tmp/apache-tomcat-7.0.91.tar.gz -C ${ADMIN_HOME} && \
  rm /tmp/apache-tomcat-7.0.91.tar.gz && \
  chmod +x ${CATALINA_HOME}/bin/*sh

RUN mkdir -p ${CATALINA_HOME}/deploy/

# Add support for the Chinese language in containers.
ENV LANG="en_US.UTF-8"

# Enhance the web shell use experience.
ENV TERM=xterm

# Download and deploy the WAR package of the EDAS application demo.
RUN wget http://edas-hz.oss-cn-hangzhou.aliyuncs.com/demo/1.0/hello-edas.war -O /
tmp/ROOT.war && \
  rm -rf ${CATALINA_HOME}/webapps/ROOT && \
  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
the run mode, and the catalina log is displayed in the standard command line.
WORKDIR ${ADMIN_HOME}
CMD ["catalina.sh", "run"]

```

Dockerfile example of Spring Cloud or Dubbo applications (based on a JAR package)

```

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

# Install the required software for packaging.
RUN yum install -y wget unzip telnet lsof net-tools bind-utils

# Prepare the JDK or Tomcat system variables.
ENV JAVA_HOME /usr/java/latest
ENV PATH ${JAVA_HOME}/bin:$PATH
ENV ADMIN_HOME /home/admin

# Download and install OpenJDK.
RUN yum -y install java-1.8.0-openjdk-devel

# Create a JAVA_HOME symbolic link.
RUN if [ ! -L "${JAVA_HOME}" ]; then mkdir -p 'dirname ${JAVA_HOME}' && ln -s 'readlink -f /
usr/lib/jvm/java' ${JAVA_HOME}; fi

# Download and deploy the JAR package of the EDAS application demo.
RUN mkdir -p ${ADMIN_HOME}/app && \
  wget http://edas-hz.oss-cn-hangzhou.aliyuncs.com/demo/1.0/hello-edas-0.0.1-
SNAPSHOT.jar -O ${ADMIN_HOME}/app/hello-edas-0.0.1-SNAPSHOT.jar

# Add support for the Chinese language in containers.
ENV LANG="en_US.UTF-8"

# Enhance the web shell use experience.
ENV TERM=xterm

# Include the startup command in the startup script start.sh.
RUN mkdir -p ${ADMIN_HOME}

```

```
RUN echo '${JAVA_HOME}/bin/java -jar ${CATALINA_OPTS} ${ADMIN_HOME}/app/hello-
edas-0.0.1-SNAPSHOT.jar'> ${ADMIN_HOME}/start.sh && chmod +x ${ADMIN_HOME}/start
.sh

WORKDIR ${ADMIN_HOME}

CMD ["/bin/bash", "/home/admin/start.sh"]
```

Customize the Dockerfile

You can customize settings in the created standard Dockerfile as needed.

Upgrade OpenJDK: you can download and install a later version of OpenJDK in the Dockerfile. The following examples show how to download and install OpenJDK 9:

```
# Download and install OpenJDK 9.
RUN yum -y install java-1.9.0-openjdk-devel
```

Upgrade the version of EDAS Container (for HSF applications only)

For HSF applications deployed in Container Service Kubernetes clusters, you can upgrade the EDAS Container version to use new middleware features or to fix the existing issues. The upgrade procedure is as follows:

1. Log on to the [EDAS console](#). In the left-side navigation pane, choose **Application Management > Applications**. In the upper-right corner of the **Applications** page, click **Create Application**. On the **Application Information** page, check **Application Runtime Environment** and the latest EDAS Container version (3.X.X).
2. Choose an EDAS Container or Pandora version based on [#unique_19](#).
3. Replace the version in the Dockerfile, such as 3.5.4.

```
# Set the version of EDAS Container or Pandora.
ENV EDAS_CONTAINER_VERSION V3.5.4
```

4. Recreate and publish an application image.

Add the EDAS runtime environment to the Tomcat startup parameters.

EDAS provides the JVM environment variable `EDAS_CATALINA_OPTS`, which contains the minimum parameters required during runtime. Tomcat provides the customized JVM parameter configuration option **JAVA_OPTS** for setting parameters such as **Xmx** and **Xms**.

```
# 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"
```

Build a local image

Find the directory where the Dockerfile is located from the local command line. Run the docker build command to build an image:

```
docker build -t [Label name, preferably application name]:[version].  
or  
docker build -t [Label name, preferably application name]:[version] -f /path/to/  
custom_dockerfile_name. # If the created Dockerfile is in another location or its name is  
not Dockerfile, use the latter command.
```

Example:

```
docker build -t hsf-provider:1.0.0.
```

Then use the `docker images | grep<Image label name>` command to view the packed local images.

Upload the image to the image repository

Application images created and built locally can be uploaded to Container Registry provided by Alibaba Cloud. Log on to the and select the region where your application is located. In the repository list, select an existing repository or create a new one to store your images. If you create a new repository, name it after your application for easy identification

To upload the local image to the repository, run the commands of pushing images to registry provided on the Basic Information tab of the specified repository.

```
docker login --username=[current account name] registry.[region_i d].aliyuncs.com ... #  
Instead of entering your Alibaba Cloud account password, enter the fixed or temporary  
password that is used as the access credential for the default instance on Container  
Registry.  
docker tag [ID of the local application image] registry.[region_i d].aliyuncs.com/[  
namespace name]/[repository name]:[image version]  
docker push registry.[region_i d].aliyuncs.com/[namespace name]/[repository name]:[  
image version]
```

Example:

```
docker login --username=tdy218@gmail.com registry.cn-hangzhou.aliyuncs.com  
docker tag 2b64f63601a7 registry.cn-hangzhou.aliyuncs.com/webapps/hsf-provider:1.0  
.0  
docker push registry.cn-hangzhou.aliyuncs.com/webapps/hsf-provider:1.0.0
```

Regulations for creating an image

Follow the specifications and restrictions below when using Dockerfile to create a custom image:

- Tenant and encryption information

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

- Resources

Resource type	Resource name	Description
Secret	edas-certs	An encryption dictionary that stores the EDAS tenant information.

- Environment variables

Environment variable key	Resource name	Description
tenantId	String	The ID of an EDAS tenant.
accessKey	String	The AccessKey ID for authentication.
secretKey	String	The AccessKey secret for authentication.

- Local files

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

- Service information

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

- Resources

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

- Environment variables

Environment variable key	Type	Description
EDAS_ADDRESS_SERVER_DOMAIN	String	The service domain name or IP address of the configuration center.
EDAS_ADDRESS_SERVER_PORT	String	The service port of the configuration center.
EDAS_CONFIGSERVER_CLIENT_PORT	String	The port of Config Server

- Environment variables during application runtime (required)

The following environment variables are provided during the 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	The ID of the EDAS application.
EDAS_ECC_ID	String	EDAS ECC ID
EDAS_PROJECT_NAME	String	The same as EDAS_APP_ID , used for trace parsing.
EDAS_JM_CONTAINER_ID	String	The same as EDAS_ECC_ID, used for trace parsing.
EDAS_CATALINA_OPTS	String	The CATALINA_OPTS parameter required during the middleware runtime.

Environment variable key	Type	Description
CATALINA_OPTS	String	The same as EDAS_CATALINA_OPTS, which is a default startup parameter of Tomcat.

4.2 Application deployment overview (Kubernetes clusters)

After developing a Spring Cloud, Dubbo, or High-speed Service Framework (HSF) application, you can use a WAR or a JAR package to deploy the application to a Kubernetes cluster. To do so, you can use the Enterprise Distributed Application Service (EDAS) console, plug-ins, or Apsara DevOps to deploy the application to a Kubernetes cluster.

Application runtime environment

When deploying the application to a Kubernetes cluster, you need to select the corresponding application runtime environment if a different deployment method is selected.

- Select the relevant **Apache Tomcat** version when using a WAR package to deploy Spring Cloud, Dubbo, common Java web, or Spring model-view-controller (MVC) applications where Pandora or Pandora Boot has not been used.
- Select the **standard Java runtime environment** when using a JAR package to deploy Spring Cloud, Dubbo, common Java web, or Spring MVC applications where Pandora or Pandora Boot has not been used.
- Select the relevant **EDAS Container** version when using a WAR or JAR package to deploy HSF applications, including Spring Cloud applications developed with Pandora Boot and applications converted from Dubbo to HSF.

Deploy applications in Kubernetes clusters

The following tutorials help you develop applications in the EDAS console. We recommend that you use Google Chrome to perform operations in the console.

- Console deployment:
 - [#unique_21](#)
 - [#unique_22](#)

- Tool deployment:
 - [#unique_23](#)
 - [#unique_24](#)
 -

4.3 Application Deployments (Console)

4.3.1 Use an image to deploy an application in a Container Service Kubernetes cluster (new version)

Container Service for Kubernetes provides enterprise-level high-performance and flexible management for Kubernetes containerized applications throughout the application lifecycle. This service simplifies cluster creation and scale-out and integrates capabilities of Alibaba Cloud in virtualization, storage, networking, and security. Therefore, this service provides an improved runtime environment for Kubernetes containerized applications.

Prerequisites

You can deploy applications in Container Service Kubernetes clusters by using images. For this purpose, you must prepare images in advance, create a Container Service Kubernetes cluster in the Container Service for Kubernetes console, and import this cluster to the Enterprise Distributed Application Service (EDAS) console. Then, you can create and deploy applications in this cluster.

- Your Alibaba Cloud account has activated EDAS. For more information, see [#unique_14](#). Meanwhile, your account has activated [Container Service for Kubernetes](#).
- You have completed role authorization in the Container Service for Kubernetes console. For more information, see [#unique_15](#).
- [You have prepared application images for the Container Service Kubernetes cluster to which you want to deploy an application.](#)

Step 1: Create a Container Service Kubernetes cluster

1. Log on to the [Container Service for Kubernetes console](#).

2. In the left-side navigation pane, choose **Resources > Clusters**. On the Clusters page, click **Create Kubernetes Cluster** in the upper-right corner.

In Container Service for Kubernetes, you can create **Kubernetes**, **Managed Kubernetes**, and **multi-zone Kubernetes** clusters. Select a type as needed.

- **#unique_27**: Three of the instances that you purchase and add must serve as master nodes. You cannot deploy applications on these three instances. You can only deploy applications on other instances (workers).
- **#unique_28**: All the instances that you purchase and add must serve as worker nodes. You can deploy applications on these instances.

**Note:**

If you have created a public-facing Server Load Balancer (SLB) instance and configured a whitelist when you create a Container Service Kubernetes cluster, you must add the 10.X.X.X/X CIDR block and 11.193.253.0/24 to this whitelist. This ensures that EDAS can access resources in this Container Service Kubernetes cluster.

Step 2: Import the Container Service Kubernetes cluster to the EDAS console

1. Log on to the [EDAS console](#).
2. In the left-side navigation pane, choose **Resources > Clusters**.
3. On the **Clusters** page, click **Container Service K8s Cluster**. On the Clusters page, select the Container Service Kubernetes cluster you created. In the **Actions** column, click **Import**. In the **Import Kubernetes Cluster** dialog box, click **Import**.
 - To view the import log, click **Log** in the **Actions** column.
 - To cancel the import, click **Cancel Import** in the **Actions** column.
 - If the import fails, click **Retry** in the **Actions** column.

When **Import Status** for the imported Container Service Kubernetes cluster appears as **Import Success** and the cluster status is **Running**, the cluster is imported to EDAS.

Step 3: Create an application in the Container Service Kubernetes cluster

1. Configure the basic information for an application.
 - a) Log on to the [EDAS console](#).
 - b) In the upper-left corner of the console, select a **region**.
 - c) In the left-side navigation pane, choose **Application Management > Applications**. On the **Applications** page, click **Create Application**.
 - d) On the **Application Information** tab, set the basic information and parameters for the application, and click **Next**.
 - Select **Kubernetes Cluster** for **Cluster Type**.
 - Select **Custom** for **Application Runtime Environment**.
 - **Application Name**: Enter the name of the application.
 - **Application Description**: Enter the basic information of the application.
2. Configure application deployment information.
 - **Namespace**: Select a namespace from the right-side drop-down list.

If you do not select a namespace, the default setting is **default**.
 - **Cluster Type**: Select a cluster from the right-side drop-down list.
 - **K8S Namespace**: Internal system objects are allocated to different namespaces to form logically isolated projects, groups, or user groups. Therefore, different groups can share resources of the whole cluster while being separately managed.
 - **default**: The default namespace. When an object is not set with a namespace, default is used.
 - **kube-system**: The namespace used by objects that are created by the system.
 - **kube-public**: The namespace that is automatically created by the system. This namespace can be read by all users, including users who are not authenticated.
 - **Application Name**: Enter the name of the application.
 - **Application Description**: Enter the summary information of the application.

3. Set the application image.

- Use a custom image.

a. Select **Configure Image** for **Image Type**.

b. Configure the image information as prompted.

Alibaba Cloud Image Service: The options includes **Current Account** and **Other Alibaba Cloud Accounts**.

- If you select **Current Account**, you must set **Region of the Image**, **Container Mirroring Service**, **Mirror Warehouse Namespace**, and **Mirror Warehouse Name**, and select an image version.

- If you select **Other Alibaba Cloud Accounts**:

■ When your image is stored in a public repository, only set **Full Mirror Address**.

■ When your image is stored in a private repository, you must use a password-free plug-in to pull the container image. For more information, see [#unique_29/unique_29_Connect_42_section_nc4_6af_h8y](#).

- Use the Demo image.

a. Select **Mirror Warehouse Namespace**, such as edas-demo-image.

b. Select **Mirror Warehouse Name**.

c. Select an image version.

4. Set pod parameters.

Pods are the smallest units for application deployment. An application can contain multiple pods. When an SLB instance is used, a request is randomly allocated to a pod for processing.

a) Set **Total Pods**.

When a pod fails to run or encounters a fault, this pod can automatically restart or services on this pod seamlessly fail over to other pods. This ensures high availability for applications. For stateful applications that use persistent storage, instance data is retained when these applications are redeployed. For stateless applications, instance

data is not retained when these applications are redeployed. You can set Total Pods to a maximum value of 50.

b) Set **Single Pod Resource Quota**.

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

5. Optional: Set the startup command and parameters.



Note:

If you do not know the [CMD](#) and [ENTRYPOINT](#) content of the original Dockerfile image, do not modify the custom startup command and parameters. Otherwise, you cannot create applications due to an incorrect custom command.

- **Startup Command:** Enter the startup command. To run the CMD `["/usr/sbin/sshd",-D"]` command, enter `/usr/sbin/sshd -D` in the field.
- **Startup Parameters:** Enter one parameter per line. For example, `args:["-c"; "while sleep 2"; "do echo date"; "done"]` contains four parameters. You must enter these parameters in four lines.

6. Optional: Set environment variables.

When you create the application, inject the environment variables you have entered to the container that will be generated. This saves you from repeatedly adding common environment variables.

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

- `MYSQL_ROOT_PASSWORD` (required): allows you to set a MySQL root password.
- `MYSQL_USER` and `MYSQL_PASSWORD` (optional): allows you to add an account in addition to the root account and set a password.
- `MYSQL_DATABASE` (optional): allows you to set the database that you want to create when the container is generated.

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

7. Optional: Set persistent storage.



Note:

Before you enable persistent storage, ensure that you have activated [NAS](#) for your Alibaba Cloud account that uses the EDAS service. The [billing method](#) of NAS is pay-as-

you-go. Ensure that your account has a sufficient balance or is billed on a pay-as-you-go basis.

- **Storage Type:** The default value is NAS, which cannot be changed.

Storage Service Type: Currently, only SSD (Performance-Type) is supported, which cannot be changed.

- **Select NAS:**

- **Buy New NAS:** Select an NAS mount directory and a local mount directory. A single region supports up to 10 NAS instances. When this number in a single region exceeds 10, your attempt to create more NAS instances fails. If you need to create more NAS instances, submit a ticket.
- **Use Existing NAS:** Select an existing NAS instance. You can create up to two mount points. Only compliant NAS instances appear in the drop-down list.

- **Mount Directory:** Set the mount directory command.

8. Optional: Set local storage.

You can map part of the file system of the host to the container as needed. Before you use this function, read [hostpath](#) and consider the rationality of the solution.

File types:

Parameter	Value	Description
Default	Null string	Indicates that the file is directly mounted without the file type check.
(New) File directory	DirectoryOrCreate	The file directory. If no file directory exists, a new directory is created.
File directory	Directory	The file directory. If the file directory does not exist, container startup fails.
(New) File	FileOrCreate	The file type. If no file exists under this file type, a new file is created.
File	File	The file. If the file does not exist, container startup fails.

Parameter	Value	Description
Socket	Socket	The standard UNIX Socket file. If the file does not exist , container startup fails.
CharDevice	CharDevice	The character device file. If the file does not exist, container startup fails.
BlockDevice	BlockDevice	The block storage device file. If the file does not exist , container startup fails.

**Note:**

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

9. Optional: Configure application lifecycle management.

Container Service Kubernetes clusters support stateless and stateful applications.

- **Stateless:** A stateless application supports multi-replica deployment. When a stateless application is redeployed, the instance data is not saved. A stateless application can be one of the following applications:
 - A web application that does not retain instance data during upgrade or migration.
 - An application that can be scaled out to address sudden traffic changes.
- **Stateful:** A stateful application stores data that requires persistent storage and retains instance data during upgrade or migration. A stateful application can be one of the following applications:
 - An application that frequently operates on containers by using Secure Socket Shell (SSH).
 - An application that requires persistent data storage (such as 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.

Lifecycle management scripts:

- **Poststart:** a container hook, which is triggered immediately after a container is created to notify the container of its creation. The hook does not pass any parameters to the corresponding hook handler. If the corresponding hook handler fails to run, the

container is killed and the restart policy of the container is used to determine whether to restart the container. For more information, see [Container Lifecycle Hooks](#).

- **PreStop**: a container hook, which is triggered before a container is deleted. The corresponding hook handler must be executed before the request to delete the container is sent to the Docker daemon. The Docker daemon sends an SGTERN semaphore to itself to delete the container, regardless of the execution result of the corresponding hook handler. For more information, see [Container Lifecycle Hooks](#).
- **Liveness**: a container status probe, which monitors the health status of applications. If an application is unhealthy, its corresponding container is deleted and recreated. For more information, see [Pod Lifecycle](#).
- **Readiness**: a container status probe, which monitors whether applications have started and are running properly. If an application is not running, the container status is updated. For more information, see [Pod Lifecycle](#).

10.Optional: Configure log collection

You can activate Log Service (SLS). Then, EDAS can store business logs and container standard output (stdout) and standard error (stderr) logs to SLS. In this way, you can view logs without limits on the number of lines and performs automatic aggregation and analysis on these logs. SLS is billed on a pay-as-you-go basis.

a) In the Log Collection Settings section, click **Enable Log Collection to SLS**.

b) Select **File Log** and **Container Standard Output Log**.

The log paths must be added for file logs.

c) After the configuration is complete, click **Create**.

Creating an application may take up to several minutes. During the creation, you can track the creation process based on the [#unique_31](#). An application in a Container Service Kubernetes cluster is deployed upon creation. After an application is created, go to the Basic Information page to view the pod status on the Instance Information tab. If the pod status is **Running**, the application is successfully deployed.

Upgrade the application in the Container Service Kubernetes cluster

An application in a Container Service Kubernetes cluster is deployed upon creation. Therefore, you can use the application deployment feature to upgrade this application. You can update the image version and reset advanced settings such as environment variables to upgrade this application.

1. On the Applications page, click the name of the created Container Service Kubernetes application to go to the Basic Information page.
2. On the page that appears, click **Deploy Application** in the upper-right corner.
3. On the **Deploy Application** page, you can modify the value of **Configure Image** and configure other advanced setting options, such as **Startup Command**, **Environment Variables**, **Persistent Storage**, **Local Storage**, and **Application Life Cycle Management**. After you complete the parameter settings, click **OK**.
4. After the application is upgraded, its image version on the Basic Information page is changed to the new version. On the **Instance Information** tab, check the status of the application. If the status is **Running**, the application is upgraded.

What to do next

After the application is created, add a public-facing SLB instance to it for Internet access. Adding an internal SLB instance allows all nodes in the same Virtual Private Cloud (VPC) to access your application. For more information, see [#unique_32](#).

Feedback

If you have any questions about using Container Service Kubernetes clusters, scan the following QR code to join the DingTalk group and give feedback.

4.3.2 Use a JAR or WAR package to deploy an application (new version)

Container Service for Kubernetes provides enterprise-level high-performance and flexible management for Kubernetes containerized applications throughout the application lifecycle. This service simplifies cluster creation and scale-out and integrates capabilities of Alibaba Cloud in virtualization, storage, networking, and security. Therefore, this service provides an improved runtime environment for Kubernetes containerized applications.

Prerequisites

- Before you use a WAR or JAR package to deploy an application in a Container Service Kubernetes cluster, you must create a Container Service Kubernetes cluster in the Container Service for Kubernetes console and import this cluster to the Enterprise Distributed Application Service (EDAS) console. Alternatively, you can use an image to deploy an application in the Container Service Kubernetes cluster. For more information, see [#unique_22](#).
- Your Alibaba Cloud account has activated [EDAS](#) and [Container Service for Kubernetes](#).

- You have completed [#unique_15](#) in the Container Service for Kubernetes console.
- [Create Docker images for applications](#)

Step 1: Create a Container Service Kubernetes cluster

Log on to the [Container Service for Kubernetes console](#), [#unique_16](#).

Step 2: Import the Container Service Kubernetes cluster to the EDAS console

1. Log on to the [EDAS console](#).
2. In the left-side navigation pane, choose **Resources > Clusters**.
3. On the **Clusters** page, click **Container Service K8s Cluster**. On the Clusters page, select the Container Service Kubernetes cluster you created. In the **Actions** column, click **Import**. In the **Import Kubernetes Cluster** dialog box, click **Import**.

- To view the import log, click **Log** in the **Actions** column.
- To cancel the import, click **Cancel Import** in the **Actions** column.
- If the import fails, click **Retry** in the **Actions** column.

When **Import Status** for the imported Container Service Kubernetes cluster appears as **Import Success** and the cluster status is **Running**, the cluster is imported to EDAS.

4. On the **Clusters** page, click **Container Service K8S Cluster**. On the Clusters page, select the Container Service Kubernetes cluster you created. In the **Actions** column, click **Import**. In the **Import Kubernetes Cluster** dialog box, click **Import**.

When the option button in the **Actions** column changes to **Delete** and the cluster status is **Running**, the cluster is imported to EDAS.

Step 3: Deploy an application in the Container Service Kubernetes cluster



Note:

The deployment procedure by using a WAR package is the same as the deployment procedure by using a JAR package. This topic uses JAR as an example to describe how to deploy an application.

1. Configure the basic information for an application.

- a) Log on to the [EDAS console](#).
- b) In the upper-left corner of the console, select a **region**.
- c) In the left-side navigation pane, choose **Application Management > Applications**. On the **Applications** page, click **Create Application**.
- d) On the **Application Information** tab, set the basic information and parameters for the application, and click **Next**.

- Select **Kubernetes Cluster** for **Cluster Type**.
- **Application runtime environment**
 - Deployment by using a JAR package: Select **Java** and configure the Java environment.
 - Deployment by using a WAR package:

To create a Dubbo or Spring Boot application, select **Tomcat**, and configure the Java environment and container version.

To create a High-Speed Service Framework (HSF) application, select **EDAS-Container(HSF)**, and configure the Java environment, container version, Pandora version, and Ali-Tomcat version.



Note:

Currently, the Java environment supports Open JDK 8 and Open JDK 7. The container version supports Apache Tomcat 8.5.42 and Apache Tomcat 7.0.91.

- **Application Name:** Enter the name of the application.
- **Application Description:** Enter the basic information of the application.

2. On the **Application Configuration** tab, configure the deployment information and set parameters.

- **Namespace:** Select a region from the left-side drop-down list. Select a namespace from the right-side drop-down list. If you do not select a namespace, the default setting is **default**.
- **Cluster:** Select a cluster from the right-side drop-down list.
- **K8S Namespace:** Internal system objects are allocated to different namespaces to form logically isolated projects, groups, or user groups. Therefore, different groups can share resources of the whole cluster while being separately managed.
 - **default:** The default namespace. When an object is not set with a namespace, default is used.
 - **kube-system:** The namespace used by objects that are created by the system.
 - **kube-public:** The namespace that is automatically created by the system. This namespace can be read by all users, including users that are not authenticated.

3. Configure the deployment package information.

- **Customer Code.**

File Uploading Method:

- **Upload JAR Package:** Select and upload the downloaded JAR package.
- **JAR Package Location:** Enter the address of the demo package.

- **Official Demo.**

EDAS provides the following demo types: **Spring Cloud Server Application, Spring Cloud Client Application, Dubbo Server Application, and Dubbo Client Application.**

Select a demo type as needed.



Note:

After you create an application by using the official demo program, you can experience the microservice calling, monitoring, and O&M features.

4. Configure the application version, time zone, and pod information, and then click **Next**.

- **Version:** Enter the version of the application.
- **Time:** Set the time zone of the application.
- **Total Pods:** Enter the number of pods required by the application.
- **Single Pod Resource Quota:** Set the CPU cores and memory for individual pods.

5. Optional: Set the startup command and parameters.



Note:

If you do not know the [CMD](#) and [ENTRYPOINT](#) content of the original Dockerfile image, do not modify the custom startup command and parameters. Otherwise, you cannot create applications due to an incorrect custom command.

- **Startup Command:** Enter the startup command. To run the CMD ["/usr/sbin/sshd",-D"] command, enter /usr/sbin/sshd -D in the field.
- **Startup Parameters:** Enter one parameter per line. For example, args:["-c"; "while sleep 2"; "do echo date"; "done"] contains four parameters. You must enter these parameters in four lines.

6. Optional: Set environment variables.

When you create the application, inject the environment variables you have entered to the container that will be generated. This saves you from repeatedly adding common environment variables.

If you need to set parameters such as the JVM heap memory, JVM property parameters, and javaagent, you can add the relevant parameters when you set environment variables:

Variable name: **CATALINA_OPTS**, variable value: **[Parameters to be added]**
\$(EDAS_CATALINA_OPTS)

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

- **MYSQL_ROOT_PASSWORD** (required): allows you to set a MySQL root password.
- **MYSQL_USER** and **MYSQL_PASSWORD** (optional): allows you to add an account in addition to the root account and set a password.
- **MYSQL_DATABASE** (optional): allows you to set the database that you want to create when the container is generated.

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

7. Optional: Set persistent storage.

In Container Service Kubernetes clusters of Alibaba Cloud, the physical storage of a native volume object is not persistent. That is, the volume object is a temporary storage object and has the same lifecycle as Kubernetes pods. You can use the [Network Attached Storage \(NAS\)](#), persistent storage service provided by Alibaba Cloud to store instance

data persistently. When an instance is upgraded or migrated, the instance data is retained.



Note:

Before you configure persistent storage, ensure that you have activated [NAS](#) for your Alibaba Cloud account that uses the EDAS service. The [#unique_30](#) of NAS is pay-as-you-go. Ensure that your account has a sufficient balance or is billed on a pay-as-you-go basis.

- **Storage Type:** The default value is NAS, which cannot be changed.
- **Storage Service Type:** Currently, only SSD (Performance-Type) is supported, which cannot be changed.
- **Select NAS:**
 - **Buy New NAS:** Select an NAS mount directory and a local mount directory. A single region supports up to 10 NAS instances. When this number in a single region exceeds 10, your attempt to create more NAS instances fails. If you need to create more NAS instances, submit a ticket.
 - **Use Existing NAS:** Select an existing NAS instance. You can create up to two mount points. Only compliant NAS instances appear in the drop-down list.
- **Mount Directory:** Set the mount directory command.

8. Optional: Set local storage.

You can map part of the file system of the host to the container as needed. Before you use this function, read [hostpath](#) and consider the rationality of the solution.

File types:

Parameter	Value	Description
Default	Null string	Indicates that the file is directly mounted without the file type check.
(New) File directory	DirectoryOrCreate	The file directory. If no file directory exists, a new directory is created.
File directory	Directory	The file directory. If the file directory does not exist, container startup fails.

Parameter	Value	Description
(New) File	FileOrCreate	The file type. If no file exists under this file type, a new file is created.
File	File	The file. If the file does not exist, container startup fails .
Socket	Socket	The standard UNIX Socket file. If the file does not exist , container startup fails.
CharDevice	CharDevice	The character device file. If the file does not exist, container startup fails.
BlockDevice	BlockDevice	The block storage device file. If the file does not exist , container startup fails.

**Note:**

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

9. Optional: Configure application lifecycle management.

Container Service Kubernetes clusters support stateless and stateful applications.

- **Stateless:** A stateless application supports multi-replica deployment. When a stateless application is redeployed, the instance data is not saved. A stateless application can be one of the following applications:
 - A web application that does not retain instance data during upgrade or migration.
 - An application that can be scaled out to address sudden traffic changes.
- **Stateful:** A stateful application stores data that requires persistent storage and retains instance data during upgrade or migration. A stateful application can be one of the following applications:
 - An application that frequently operates on containers by using Secure Socket Shell (SSH).
 - An application that requires persistent data storage (such as 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.

Lifecycle management scripts:

- **Poststart:** a container hook, which is triggered immediately after a container is created to notify the container of its creation. The hook does not pass any parameters to the corresponding hook handler. If the corresponding hook handler fails to run, the container is killed and the restart policy of the container is used to determine whether to restart the container. For more information, see [Container Lifecycle Hooks](#).
- **PreStop:** a container hook, which is triggered before a container is deleted. The corresponding hook handler must be executed before the request to delete the container is sent to the Docker daemon. The Docker daemon sends an SGTERN semaphore to itself to delete the container, regardless of the execution result of the corresponding hook handler. For more information, see [Container Lifecycle Hooks](#).
- **Liveness:** a container status probe, which monitors the health status of applications. If an application is unhealthy, its corresponding container is deleted and recreated. For more information, see [Pod Lifecycle](#).
- **Readiness:** a container status probe, which monitors whether applications have started and are running properly. If an application is not running, the container status is updated. For more information, see [Pod Lifecycle](#).

10.Optional: Configure log collection.

You can activate Log Service (SLS). Then, EDAS can store business logs and container standard output (stdout) and standard error (stderr) logs to SLS. In this way, you can view logs without limits on the number of lines and performs automatic aggregation and analysis on these logs. SLS is billed on a pay-as-you-go basis.

- a) In the Log Collection Settings section, click **Enable Log Collection to SLS**.
- b) Select **File Log** and **Container Standard Output Log**.

The log paths must be added for file logs.

After the configuration is complete, click **Create**.

Creating an application may take up to several minutes. During the creation, you can use an [#unique_8](#) to track the creation progress. An application in a Container Service Kubernetes cluster is deployed upon creation. After an application is created, go to the Basic Information page to view the pod status on the Instance Information tab. If the pod status is **Running**, the application is successfully deployed.

What's next

After the application is created, add a public-facing SLB instance to it for Internet access. Adding an internal SLB instance allows all nodes in the same Virtual Private Cloud (VPC) to access your application. For more information, see [#unique_32](#).

If you have any questions about using Container Service Kubernetes clusters, scan the following QR code to join the DingTalk group and give feedback.

4.4 Deploy applications with tools

4.5 Canary release (Kubernetes clusters)

To upgrade Spring Cloud or Dubbo microservice-oriented applications deployed in a Container Service Kubernetes cluster, you can use canary releases to perform a small-scale validation first, and then perform a full upgrade if the validation succeeds.

Limits

- Dubbo applications have no restrictions when canary release is used.

- Spring Cloud applications have the following restrictions when canary release is used:
 - Netflix Zuul and Spring Cloud Gateway are not supported at present.
 - Ingress applications (the first application node that receives external traffic) do not support canary release.

Ingress applications directly receive external traffic, and because such traffic cannot be intercepted or routed, canary traffic control is temporarily unavailable to ingress applications. An ideal mode is to build a microservice gateway as the "ingress application" to receive external traffic and put all business applications behind it, so as to implement canary traffic control for ingress applications.

- However, do not use canary release if the application contains the following native features or configurations:
 - Horizontal Pod Autoscaler (HPA)
 - Rancher
 - Istio
 - Features and configurations that are dependent on `Deployment.Metadata.Name` or `Deployment.Metadata.Uid`

Procedure

1. Log on to the [EDAS console](#).
2. In the left-side navigation pane, choose **Application Management > Applications**.
3. On the **Applications** page, select a **region** and **Namespaces** and then click the name of the target application.
4. On the application details page, click **Deploy Application** in the upper-right corner.
5. On the **Select Deployment Mode** page, click **Start Deployment** in the upper-right corner of the **Canary Release (Grayscale)** section.

6. On the **Canary Release** page, upload the deployment package of the new application version, set the canary release policy and rules, and then click **Confirm**.

- a) Upload the deployment package for the new application version.
- b) In the **Release Strategy** section, configure release policy parameters.

The right-side **Release Strategy Configuration Information** section shows the canary release procedure according to the configuration.

The following describes the release policy parameters:

- **First Grayscale Quantity:** the number of application instances in a canary release. The current number of instances for the application is displayed on the right side. To ensure application stability, the number of instances in a canary release cannot exceed 50% of the total number of application instances.
- **Remaining Batch:** After a canary release, the remaining application instances are released in specified number of batches.
- **Batch Mode:** This parameter is required when **Remaining Batch** is set to 2 or a larger value. Valid values include Automatic and Manual.
 - Automatic: automatically releases in batches according to **Interval**. **Interval:** the release interval for the remaining batches. Unit: minute.
 - Manual: manually triggers the release of the next batch.
- **Batch in Deployment Interval:** the deployment interval (unit: second) between application instances if the number of application instances is greater than one in each batch.

- c) Set canary release rules.

Two canary release rules are provided: **Grayscale by Content** and **Proportional Grayscale**.

- **Grayscale by Content:** Click **Create New Entrance Flow Rules** and set a new rule for inbound traffic.



Note:

You can create multiple inbound traffic rules.

Parameters of canary release by content:

- **Protocol Type:** select Spring Cloud or Dubbo according to the actual situation of the application.
 - Spring Cloud: **Path** is required.
 - Dubbo: **Select Service** and **Method** are required.
- **Scene Rules:** valid values include **At the same time meet the following conditions** and **Meet any of the following conditions**.
- **Condition List:** conditions for Spring Cloud and Dubbo protocol are different. Three methods are available: Cookie, Header and Parameter. Set the parameters as needed.
 - Spring Cloud: valid values include Cookie, Header and Parameter. Set as required.
 - Dubbo: set **Parameters** and **Parameter value gets the expression** based on the actual use of your application.
- **Proportional Grayscale:** set **Traffic Ratio**. Traffic will be forwarded to the current canary instance group according to this value.

After a canary release is started, deploy the new application version to the specified canary instance group. The **Basic Information** page prompts that **A change process is being executed for this application. Status: {0}**. Click **View Details**. On the **Change Details** page, view the deployment progress and status.

Terminate the change: **The application is in a canary release and the change has been terminated. Please roll back the app before doing anything else.**

7. Monitor whether the volume of traffic to the canary instance group meets expectations. For more information, see [#unique_36](#).
8. After the traffic verification is completed, click **Start the Next Batch** on the **Change Details** page. Complete the subsequent batch release.

If any issues are found during the verification process, you can click **Rollback** in the upper-right corner of the **Change Details** page. In the **Rollback** dialog box, confirm the impact of rollback, and then click **Rollback**.

Verify the result

After a canary release, check whether **deployment package** is the new version on the **Basic Information** page. On the **Instance Information** page, view whether the value of **Status** is **Running**.

4.6 Batch release (applicable to Kubernetes clusters)

In application release and product iteration, batch release is often used to control release risks.

Introduction

Batch release is the process in which only certain instances of an application are upgraded based on batches. If an error occurs during batch release, you can terminate the upgrade process and roll back the instances. You can deploy the new version of the application to the instances again after the error is fixed.

When an application is released in batches in a Container Service Kubernetes cluster, the instances of the application are evenly distributed to each batch for deployment. If the instances cannot be evenly distributed, a small number of instances are allocated for the first batches, and a large number of instances are allocated for the remaining batches.

Scenario

An application contains 10 instances. The deployed application version of each instance is V1. Now, the application in each instance needs to be upgraded to V2.

Assume that the new version of the application is released in all the instances in three batches. The following figure shows the release process based on the batch release policy.

Instructions

- When you use batch release for an application in a Container Service Kubernetes cluster , you need to create a Deployment.

- However, do not use batch release if the application contains the following native features or configurations:
 - Horizontal Pod Autoscaler (HPA)
 - Rancher
 - Istio
 - Features and configurations that are dependent on Deployment.Metadata.Name or Deployment.Metadata.Uid

Procedure

1. Log on to the [EDAS console](#).
2. In the left-side navigation pane, choose **Application Management > Applications**.
3. On the **Applications** page, select a **region** and **Namespaces** and then click the name of the target application.
4. On the Application Details page, click **Deploy Application** in the upper-right corner.
5. On the **Deployment Mode Selection** page, click **Start Deployment** in the **Batch Release** section.
6. On the **Batch Release** page, upload the application deployment package of the new version.

- WAR and JAR

WAR and JAR packages differ only in the deployment method, but have the same parameters.

- Images

You only need to select a new image version and do not need to set other parameters

.

7. Set Release Strategy.

Parameter description:

- **Release Batch**: the number of batches for releasing the new version of the application in the remaining instances after the canary release is completed.
- **Batch Mode**: the mode for releasing the new version of the application in the remaining instances. Valid values are Automatic and Manual.
 - Automatic: automatically release in batches according to **Interval**. **Interval**: the release interval for the remaining batches. Unit: minute.
 - Manual: manually triggers the release of the next batch.
- **Batch in Deployment Interval**: the deployment interval (unit: second) between application instances if the number of application instances is greater than one in each batch.

8. Optional: You can determine whether to configure **Startup Command, Environment Variables, Persistent Storage, Local Storage, Application Life Cycle Management**, and **Log Collection Settings** based on your requirements.

9. After you set the parameters, click **OK**.

Verify the result

Go to the Application Details page to view the status of batch release. If all batches are successfully released, the batch release is successful.

On the Application Details page, view the instance deployment information. When the instance version is V2 and the status of all instances is **Running**, the release is successful.

Roll back an application

During a batch release, if at least one application instance is not upgraded to the new version, the release is considered in progress. When monitoring an upgrade, if you notice that any application instance in the first batch stops responding, you can go to the Change Details page and click **Roll Back Immediately** to roll back the released instance to the previous service package and configuration.

If an exception occurs during batch release, see [#unique_37](#) for troubleshooting.

- Exceptions such as unavailable deployment package or health check failure may lead to failures of application upgrades. The current application changes are automatically terminated and the application is rolled back.

- During the upgrade, the maximum timeout period for a single batch is 30 minutes. If the change process is suspended due to timeout, you must go to the **Change Details** page to manually terminate the release process and roll back the application.

5 User Guide (Old Version)

5.1 Application monitoring

5.1.1 Application overview

When connected to Application Real-Time Monitoring Service (ARMS), your application is monitored by ARMS in all rounds. You can view the key health metrics of your application on the **Application Overview** page. The topology allows you to preview the upstream and downstream dependent components of the component, and the 3D topology allows you to view the health status of your application, services, and hosts.

Portal

1. Log on to the [ARMS console](#). In the left-side navigation pane, choose **Application Monitoring > Applications**.
2. On the **Applications** page, click the name of the target application to access the **Application Overview**.

At the top of the **Application Overview** page, you can click tabs **Overview Analysis**, **Topology Graph**, and **3D Topo (Probation)** to view the corresponding information.

Features

- **Key application metrics**

The **Overview Analysis** tab shows these key metrics:

- Total number of requests, average response time, error count, real-time number of instances, number of full garbage collection (GC) activities, number of slow

SQLs, number of exceptions within the specified time range, and how they change compared with the previous day and previous week.

- Application Support Services: time sequence curves of the application support service requests and average response time.
 - Application Dependent Services: time sequence curves of the application dependent service requests, average response time, and number of application instances, and statistics of the HTTP status code.
 - System info: time sequence curves of CPU, memory, and load.
 - Slow call: time sequence curves and details of slow calls.
 - Statistical analysis: analysis on slow API calls and exception types.
- **Application topology**

On the **Topology Graph** tab, you can get a better view of the upstream and downstream components of your application and their call relations, allowing you to effectively identify the bottlenecks of your application.

- **3D Topo (Probation)**

The **3D Topo (Probation)** tab provides a comprehensive view of the health conditions for your application, service, and host, and the upstream and downstream dependencies of the application. With the 3D topology, you can quickly identify the services that caused failures, applications influenced by the failures, and associated hosts. This enables you to thoroughly investigate the root cause of the failures, and then fix the issues quickly.

For more information about the 3D topology function, see [#unique_41](#).

5.1.2 Application details

5.1.2.1 JVM monitoring

The application monitoring function of Application Real-Time Monitoring Service (ARMS) provides the Java Virtual Machine (JVM) monitoring function. It monitors heap metrics, non-heap metrics, direct buffer metrics, memory-mapped buffer metrics, garbage collection (GC) details, and the number of JVM threads. This topic describes the JVM monitoring feature and how to monitor JVM metrics.

Features

ARMS' JVM monitoring feature can help you monitor the following metrics.

- Heap memory
 - heap_init: initial bytes of the heap memory
 - heap_max: maximum bytes of the heap memory
 - heap_committed: submitted bytes of the heap memory
 - heap_used: used bytes of the heap memory
- Non-heap memory
 - non_heap_init: initial bytes of the non-heap memory
 - non_heap_max: maximum bytes of the non-heap memory
 - non_heap_committed: submitted bytes of the non-heap memory
 - non_heap_used: used bytes of the non-heap memory
- Direct buffer
 - direct_capacity: total size of direct buffer (in bytes)
 - direct_used: used size of the direct buffer (in bytes)
- Memory-mapped buffer
 - mapped_capacity: total size of memory-mapped buffer (in bytes)
 - mapped_used: used size of memory-mapped buffer (in bytes)
- GC details
 - GcPsMarkSweepCount: GC Parallel Scavenge MarkSweep amount
 - GcPsScavengeCount: GC Parallel Scavenge amount
 - GcPsMarkSweepTime: GC Parallel Scavenge MarkSweep time
 - GcPsScavengeTime: GC Parallel Scavenge time
- Number of JVM threads
 - ThreadCount: total number of threads
 - ThreadDeadLockCount: number of deadlocked threads
 - ThreadNewCount: number of new threads
 - ThreadBlockedCount: number of blocked threads
 - ThreadRunnableCount: number of threads that can run
 - ThreadTerminatedCount: number of terminated threads
 - ThreadTimedWaitCount: number of threads in timed waiting
 - ThreadWaitCount: number of waiting threads

Procedure

1. Log on to the [ARMS console](#). In the left-side navigation pane, choose **Application Monitoring > Applications**.
2. On the **Applications** page, click the name of the target application.
3. On the **Application Details** page, select the target node and click the **JVM Monitoring** tab on the right side of the page.

The **JVM Monitoring** tab shows the time sequence curves of the instantaneous times of GC, instantaneous time consumption of GC, heap memory details, non-heap memory details, and number of JVM threads.

- Click **Instantaneous** and **Accumulated** in the upper-right corner of **Instantaneous Count/Min** and **Instantaneous Duration/Min** panels. You can alternatively view the time sequence curves of the instantaneous or accumulated times and time consumption of GC. By default, the instantaneous values are displayed.
- Click a metric name (for example, the number of instantaneous GC times) on each monitoring panel to enable or disable the visibility of this metric in the chart.



Note:

Each chart must have at least one visible metric.

5.1.2.2 Host monitoring

The application monitoring function of Application Real-Time Monitoring Service (ARMS) provides the host monitoring function. It monitors metrics such as the CPU, memory, disk, load, network traffic, and network data packets. This topic describes the host monitoring feature and how to view host monitoring metrics.

Features

Host monitoring helps you monitor the following metrics.

- CPU
 - SystemCpuIdle: idle CPU usage in the last five seconds
 - SystemCpuSystem: system CPU usage in the last five seconds
 - SystemCpuUser: user CPU usage in the last five seconds
 - SystemCpuIOWait: I/O await CPU usage in the last five minutes

- Memory
 - SystemMemFree: idle system memory (in KB)
 - SystemMemTotal: total system memory (in KB)
 - SystemMemUsed: used system memory (in KB)
 - SystemMemBuffers: current memory in the cache of system buffer (in KB)
 - SystemMemCached: current memory in the cache of system page (in KB)
 - SystemMemSwapFree: idle memory of system swap (in KB)
 - SystemMemSwapTotal: total memory of system swap (in KB)
- Disk
 - SystemDiskTotal: total bytes of the system disk
 - SystemDiskFree: idle bytes of the system disk
 - SystemDiskUsedRatio: usage of system disk
- Load
 - SystemLoad: load on the system
- Network traffic
 - SystemNetInBytes: average bytes per second received through the network in the last 30 seconds
 - SystemNetInErrs: average errors per second received through the network in the last 30 seconds
 - SystemNetOutBytes: average bytes per second sent through the network in the last 30 seconds
 - SystemNetOutErrs: average errors per second sent through the network in the last 30 seconds
- Network packets
 - SystemNetInPackets: average messages per second received through the network in the last 30 seconds
 - SystemNetOutPackets: average messages per second sent through the network in the last 30 seconds

Procedure

1. Log on to the [ARMS console](#). In the left-side navigation pane, choose **Application Monitoring > Applications**.
2. On the **Applications** page, click the name of the target application.

3. On the **Application Details** page, select the target node and click the **Host Monitoring** tab on the right side of the page.

The **Host Monitoring** tab shows the time sequence curves of the CPU, memory, disk, load, network traffic, and network packets.

- You can enable or disable the visibility of a metric in the chart by clicking its name (for example, the system CPU usage) on the monitoring panel.

**Note:**

Each chart must have at least one visible metric.

- Click the two alert icons in the upper-right corner of the monitoring panel. You can view existing alert points and create new alerts. For how to create alerts, see [#unique_45](#).

5.1.3 API call monitoring

This topic explains the API call monitoring function of application monitoring.

Features

On the **Interface Invocation** page of application monitoring, you can view the detailed statistics of the APIs of the application. Application Real-Time Monitoring Service (ARMS) can automatically discover and monitor the APIs provided in the following web frameworks and RPC frameworks:

- Tomcat 7+
- Jetty 8+
- Resin 3.0+
- Undertow 1.3+
- WebLogic 11.0+
- SpringBoot 1.3.0+
- HSF 2.0+
- Dubbo 2.5+

API overview

The **Overview** tab of the **Interface Invocation** page shows all APIs detected by ARMS. You can sort the list by response time, request count, or error count. Select a service to view its detailed topology and the line charts of request count, response time, and error count on the **Overview** tab.

SQL analysis

The **SQL Analysis** tab shows the list of SQL requests initiated within the code snippets of the selected service on the left side. Through this tab, you can identify which SQL request is the cause for the slow response of a service. You can also click **Interface Snapshot** of an SQL request to view the complete trace where the SQL is executed.

Exception analysis

The **Exception Analysis** tab shows the exceptions thrown within the code snippets of the selected service on the left side. You can also click **Interface Snapshot** of an exception to view the complete trace where the exception stack is located.

Interface snapshot

In the service trace snapshot, you can view the call stack of a single call, details of the executed SQLs, details of the thrown exceptions, and the parameters of the API.

5.1.4 Advanced monitoring

Application Real-Time Monitoring Service (ARMS) is an application performance management (APM) product of Alibaba Cloud. Enterprise Distributed Application Service (EDAS) can seamlessly interoperate with ARMS. You can enable advanced monitoring to use the APM feature of ARMS and manage your applications in EDAS.

For more information about how to enable advanced monitoring, see [#unique_48](#).

5.1.5 Alerting

5.1.5.1 Create an alert

By creating alerts, you can set alert rules for specific monitored objects. When a rule is triggered, the system will send an alert message to the specified contact group in the specified alerting mode. This reminds you to take necessary actions to solve the problem.

Prerequisites

- You have created contacts. You can only set a contact group as the notification receiver of an alert.

Context

Default behaviors of alert notifications:

- To prevent you from receiving a large number of alert notifications in a short period of time, the system only sends one message for repeated alerts within 24 hours.

- If no duplicate alerts are generated within five minutes, Application Real-Time Monitoring Service (ARMS) sends a recovery email to notify you that the alert has been cleared.
- After a recovery email is sent, the alert status is reset. If this alert arises again, it is deemed as a new one.

An alert widget is essentially a data display method of datasets. When you create an alert widget, a dataset is created to store the underlying data of the alert widget.

**Note:**

New alerts take effect within 10 minutes. The alert check may have a delay of 1 to 3 minutes.

Create an alert

To create an alert for an application monitoring job on Java Virtual Machine-Garbage Collection (JVM-GC) times in corresponding-period comparison, perform the following steps :

1. Log on to the console. Click the target application in **Applications**. In the left-side navigation pane, choose **Alerts > Alert Policies**.
2. On the **Alert Policies** page, click **Create Alert** in the upper-right corner.
3. In the **Create Alert** dialog box, enter all required information and click **Save**.
 - a) Enter **Alert Name**, for example, alert on JVM-GC times in corresponding-period comparison.
 - b) Select an application for **Application Site** and an application group for **Application Group**.
 - c) In the **Type** drop-down list, select the type of the monitoring metrics, for example, **JVM_Monitoring**.
 - d) Set Dimension to **Traverse**.
 - e) Set alert rules.
 - A. Select Meet All of the Following Criteria.**
 - B. Edit the alert rule.** For example, an alert is triggered when the value of N is 5 and the average value of JVM_FullGC increases by 100% compared with that in the previous hour.

**Note:**

To add another alert rule, click **+** on the right of **Alert Rules**.

- f) Set Notification Mode. For example, select Email.
- g) Set Notification Receiver. In the **Contact Groups** box, click the name of a contact group. If the contact group appears in the **Selected Groups** box, the setting is successful.

Description of basic fields

The following table describes the basic fields of the **Create Alert** dialog box.

Field	Description	Remarks
Application Site	The monitoring job that has been created.	Select a value from the drop-down list.
Type	The type of the metric.	<p>The types for the three alerts are different:</p> <ul style="list-style-type: none"> • Application monitoring alerts: This displays application entry calls, the statistics of application call types, database metrics, JVM monitoring, host monitoring, and abnormal interface calls. • Browser monitoring alerts: This shows page metrics, interface metrics, custom metrics, and page interface metrics. • Custom monitoring alerts: This creates alerts based on existing drilled-down datasets and existing general datasets.
Dimension	The dimensions for alert metrics (datasets). You can select None,"=", or Traverse.	<ul style="list-style-type: none"> • When it is set to None, the alert content shows the sum of all values of this dimension. • When it is set to "=", you need to enter the specific content. • When it is set to Traverse, the alert content shows the dimension content that actually triggers the alert.
Last N Minutes	The system checks whether the data results in the last N minutes meet the trigger condition.	Range of N: 3 to 3600 minutes.

Field	Description	Remarks
Notification Mode	Email, SMS, and DingTalk chatbot are supported.	You can select multiple modes. If you want to set DingTalk chatbot alert, see Enable DingTalk chatbot alert.
Alert Quiet Period	You can enable or disable Alert Quiet Period. By default, it is enabled.	<ul style="list-style-type: none"> When it is enabled: if data remains in the triggered state, the second alert message will only be sent 24 hours after the first alert is triggered. When data recovers, you will receive a data recovery notification and the alert will be cleared . If the data triggers the alert one more time, the alert message is sent again. When it is disabled: if the alert is continually triggered, the system sends the alert message every minute.
Alert Severity	Valid values include Warn , Error and Fatal.	None
Notification Time	The time when the alert was sent. No alert notification is sent out of this time period, but alert events are recorded.	For more information about alert event history, see Alert management.
Notification Content	The custom content of the alert.	You can edit the default template. In the template, the four variables, \$AlertName, \$AlertFilter, \$AlertTime, and \$AlertContent, are preset. (Other preset variables are not supported currently.) The rest of the content can be customized.

Description of complex general fields: Period-over-period comparison

- N-minute-on-N-minute comparison: Assume that β is the data (optionally average, sum , maximum, or minimum) in the last N minutes, and α is the N-minute data starting from 2N minutes ago. The N-minute-on-N-minute comparison is the percentage increase or decrease of β as compared to α .
- N-minute-on-N-minute hourly comparison: Assume that β is the data (optionally average, sum, maximum or minimum) in the last N minutes, and α is the N-minute data from an hour ago. The N-minute-on-N-minute hourly comparison is the percentage increase or decrease of β as compared to α .

- N-minute-on-N-minute daily comparison: Assume that β is the data (optionally average, sum, maximum or minimum) in the last N minutes, and α is the N-minute data a day ago. The N-minute-on-N-minute daily comparison is the percentage increase or decrease of β as compared to α .

Description of complex general fields: Alert data revision strategy

You can select "Zero fill", "One fill", or "Zero fill null" (default). This feature is generally used to fix anomalies in data, including no data, abnormal composite metrics, or abnormal period-on-period comparison.

- Zero fill: fixes the value checked to 0.
- One fill: fixes the value checked to 1.
- Zero fill null: does not trigger the alert.

Scenarios:

- Anomaly 1: No data

User A wants to use the alert feature to monitor the page views. When creating the alert, user A selects Browser Monitoring Alert. User A sets the alert rule as follows: N is 5 and the sum of the page views is at most 10. If the page is not hit, no data is reported and no alert is sent. To solve this problem, you can select "Zero fill" as the alert data revision policy. If you do not receive any data, it is considered that zero data is received. This meets the alert rule and an alert is sent.

- Anomaly 2: abnormal composite metrics

User B wants to use the alert feature to monitor the real-time unit price of a product. When creating the alert, user B selects Custom Monitoring Alert. User B sets the dataset of variable a to the current total price, and the dataset of variable b to the current total items. User B also sets the alert rule as follows: N is 3 and the minimum value of current total price divided by current total items is at most 10. If the current total of items is 0, the value of the composite metric, current total price divided by current total items, does not exist. No alert will be sent. To solve this problem, you can select "Zero fill" as the alert data revision policy. The value of the composite metric, current total price divided by current total items, is now considered as 0. This meets the alert rule and an alert will be sent.

- Anomaly 3: Abnormal period-on-period comparisons

User C wants to use the alert function to monitor the CPU utilization of the node machine . When user C creates the alert, C selects Application Monitoring Alert, and sets the alert rule as follows: N is 3 and the average user CPU utilization of the node machine decreases by 100% compared with the previous monitoring period. If the user's CPU fails to work in the last N minutes, the α cannot be obtained. This means the period-on-period result does not exist. No alert is sent. To solve this problem, you can select the alert data revision strategy as "One fill", and consider the period-on-period comparison result as a decrease of 100%. This meets the alert rule and an alert will be sent.

What's next

You can query and delete alert records in alert management.

5.1.5.2 Manage alerts

On the alert Policies page, you can manage all the alert rules under your account and query the history of alert events and alert posts.

Manage alert rules

1. Log on to the console. Click the target application in **Applications**. In the left-side navigation pane, choose **Alerts > Alert Policies**.
2. Optional: On the **Alert Rules** tab, enter the alert name in the search box, then click **Search**.



Note:

You can enter part of an alert name in the search box to perform a fuzzy search.

3. In the **Actions** column, you may take actions on the filtered alert rules, as needed:
 - To edit an alert rule, click **Edit**. In the **Edit Alert** dialog box, edit the alert rule, and click **Save**.
 - To delete an alert rule, click **Delete**. In the **Delete** dialog box, click **Delete**.
 - To start a stopped alert rule, click **Start**, and in **OK** dialog box, click **Start**.
 - To stop a running alert rule, click **Stop**, and then click **OK** in the **Stop** dialog box.
 - To view the alert event history and alert post history, click **View Alert Detail**, and view related records on the **Alert Event History** tab.

Query alert history

You can view historical records about when and why an alert rule was triggered, and about the alert notification records sent to specified alert contacts on the **Alert History** tab.

1. On the **Alert Policies** page, click the **Alert History** tab.
2. On the **Alert History** tab, select or enter the **Alert Type**, **Trigger State** and **Alert Name**, and then click **Search**.

The line charts and bar charts on the tab show the relationship between alert data and alert trigger events, also the alert trigger details. The line chart represents the alert data and the bar chart represents the alert events.

3. Scroll down to the bottom, view the history of alert events on the **Alert Event History** tab.



Note:

Alert notifications are sent only when the trigger state is **triggered**. (**Trigger** column contains a red dot.)

4. Click the **Alert Post History** tab to view the records of alert notifications (such as SMS and email) that were sent for triggered alerts.

5.1.5.3 Create contacts

When an alert rule is triggered, notifications are sent to the contact group that you specified. Before you create a contact group, you must create contacts. When creating a contact, you can specify the mobile phone number and email address of the contact to receive notifications. You can also provide a DingTalk chatbot webhook URL used to automatically send alert notifications.

Prerequisites

To add a DingTalk chatbot as a contact, you must obtain its webhook URL first. For more information, see [Enable DingTalk chatbot alert](#).

Procedure

1. Log on to the console. Click the target application in **Applications**. In the left-side navigation pane, choose **Alerts > Alert Policies**.
2. On the **Alert Policies** page, click **Create Alert** in the upper-right corner.
3. On the **Contacts** tab, click **Create Contact** in the upper-right corner.

4. In **Create Contact** dialog box, edit contact information.

- To add a contact, enter the **Name**, **Phone Number** and **Email**.

**Note:**

The phone number and email address cannot be blank at the same time. Each phone number or email address must be used for only one contact. You can create a maximum of 100 contacts.

- To add a DingTalk chatbot, enter the name and the webhook URL of the chatbot.

**Note:**

For more information about how to obtain the webhook URL of the DingTalk chatbot, see [Enable DingTalk chatbot alert](#).

What to do next

- To search for contacts, on the **Contacts** tab, select **Name**, **Phone Number**, or **Email** in the drop-down list, then enter the entire or a part of the selected name, phone number or email in the search box, and click **Search**.
- To edit a contact, click **Edit** in the **Actions** column of the contact, edit the information in the **Update Contact** dialog box, then click **OK**.
- To delete a single contact, click **Delete** in the **Actions** column of the contact, then click **Delete** in the **Delete** dialog box.
- To delete multiple contacts, select the target contacts, click **Batch Delete Contacts**, then click **OK** in the **Note** dialog box.

5.1.5.4 Create contact groups

When creating an alert rule, you can specify a contact group as the alert notifications receiver. When the alert rule is triggered, Application Real-Time Monitoring Service (ARMS) sends alert notifications to the contacts in this contact group. This topic describes how to create contact groups.

Prerequisites

You have created contacts.

Procedure

1. Log on to the console. Click the target application in **Applications**. In the left-side navigation pane, choose **Alerts > Contact Management**.
2. On the **Contact Groups** tab page, click **New Contact Group** in the upper-right corner.

3. In **Create Contact Group** dialog box, enter **Group Name**, select **Contact Members**, and click **OK**.

**Note:**

If there are no options in the **Contact Members** list, you need to first [Create a contact](#).

What to do next

- To search for a contact group, go to the **Contact Groups** tab, enter all or some characters of the contact group name in the search box, then click **Search**.

**Notice:**

English keywords are case-sensitive.

- To edit a contact group, click the pencil icon on the right side of the contact group, and edit the information in the **Edit Contact Group** dialog box.
- To show the contacts under a contact group, click the downward arrow on the right side of a contact group to expand the group.

**Note:**

You can remove one or more contacts from an expanded contact group. To remove a contact, click **Delete** in the **Actions** column of the target contact.

- To delete a contact group, click the X icon on the right side of a contact group.

**Notice:**

Before deleting a contact group, make sure that no monitoring job is running. Otherwise, alerting and other functions may be ineffective.

5.1.5.5 Enable DingTalk robot alerts

ARMS allows you to receive alert notifications from DingTalk groups. After enabling the DingTalk robot alert function, you can receive alert notifications from DingTalk groups. This topic describes how to enable the DingTalk robot alert function.

1. Obtain the address of the DingTalk robot.
 - a. Run the DingTalk client on a PC, click to enter the DingTalk group to which you want to add an alert robot, and click the Group Settings icon in the upper-right corner.
 - b. In the Group Settings dialog box that appears, choose **ChatBot**.
 - c. On the ChatBot page that appears, click **+** in the **Add Robot** section, and then click **Custom**.
 - d. In the Add Robot dialog box that appears, edit the robot avatar and name, and click **Finish**.
 - e. In the **Add Robot** dialog box that appears, copy the address that the system generates for the robot.
2. In the ARMS console, add the DingTalk robot as the contact. For more information about how to add a contact, see [Create contacts](#).
3. Create a contact group, and add the contact that you created in the previous step as the alert contact. For more information about how to create a contact group, see [Create contact groups](#).
4. Set alert rules.
 - If you have not created an alert job yet, [Create an alert](#), set the notification mode to **DingTalk Robot**, and set the notification receiver to the contact group that you created in [Step 3](#).
 - If you have created an alert job, click [Modify Alert Rules](#), set the notification mode to DingTalk Robot, and set the notification receiver to the contact group that you created in [Step 3](#).

Now, you have enabled the DingTalk robot alert function. When an alert is triggered, you can receive the alert notification in the specified DingTalk group, for example:

5.2 Log diagnosis