ALIBABA CLOUD

Alibaba Cloud

Application Real-time Monitoring Service

Application monitoring

Document Version: 20210308

Alibaba Cloud

Legal disclaimer

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

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

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

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

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

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

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

Document conventions

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

Table of Contents

1.Overview

Application Real-Time Monitoring Service (ARMS) is an application performance management (APM) service. To monitor an application, you only need to install the ARMS agent. You do not need to modify the code of the application. The ARMS agent helps you identify abnormal and slow API operations, view request parameters, and detect system bottlenecks. This improves the efficiency of online troubleshooting.

Automatically discover the application topology

The ARMS agent can automatically identify the upstream and downstream dependencies of applications. The ARMS agent can capture, compute, and display the traces that are formed by different applications in a remote procedure call (RPC) framework, such as Dubbo, HTTP, and HSF. You can identify performance bottlenecks and abnormal calls in the system by using the application topology.

View the 3D topology

A 3D topology shows the health status of applications, services, and hosts. It also shows the upstream and downstream dependencies. The 3D topology helps you detect abnormal services, affected applications, and related hosts. You can then identify the causes of issues, perform troubleshooting, and resume services in a timely manner.

Capture abnormal and slow transactions

You can obtain the stack analysis reports of slow SQL queries, accumulated Message Queue (MQ) messages, or exceptions, and conduct more detailed analysis.

Automatically discover and monitor API requests

ARMS can automatically discover and monitor common web frameworks and RPC frameworks in application code. ARMS can also automatically collect statistics on metrics such as the number of calls, response time, and number of abnormal web API requests and RPC API requests.

Perform real-time diagnosis

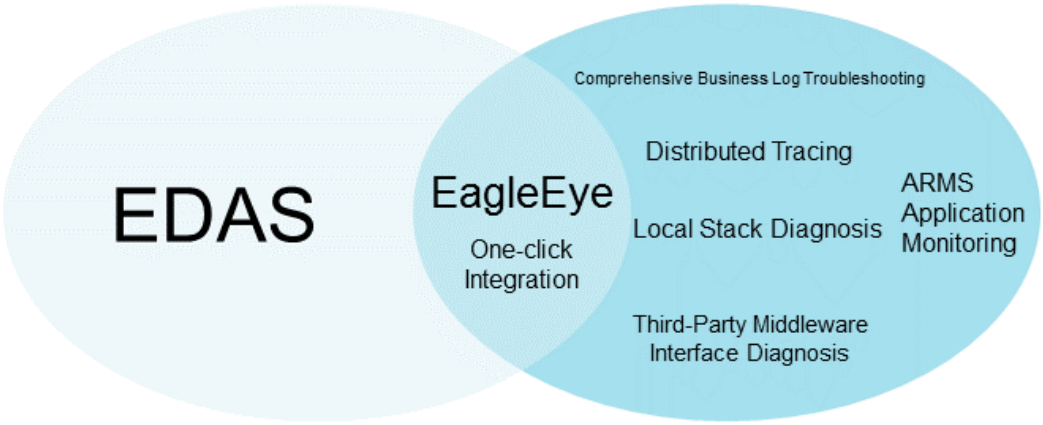
If you need to monitor the application performance for a short period of time, for example, when you release an application or perform stress tests on the application, you can use the real-time diagnosis feature. After real-time diagnosis is enabled for an application, ARMS monitors the application for 5 minutes and reports all the trace data that is generated during this period. If an error occurs on a trace, you can use call stack waterfall charts and the thread profiling feature to identify the cause of the error.

Perform multi-dimensional troubleshooting

You can view the details of distributed and local call stacks, and perform analysis based on multiple dimensions, such as application, IP address, and time consumption. You can also use the comprehensive troubleshooting feature of ARMS custom monitoring to troubleshoot transactions and tickets.

Integrate with the Alibaba Cloud PaaS platform

ARMS application monitoring can be integrated with the Alibaba Cloud PaaS platform named Enterprise Distributed Application Service (EDAS). You can monitor applications that run on EDAS with improved efficiency.

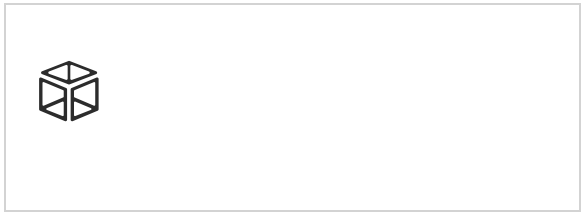

EDAS
EagleEye
One-click Integration
Comprehensive Business Log Troubleshooting
Distributed Tracing
Local Stack Diagnosis
ARMS Application Monitoring
Third-Party Middleware Interface Diagnosis

2.Quick start

2.1. Overview

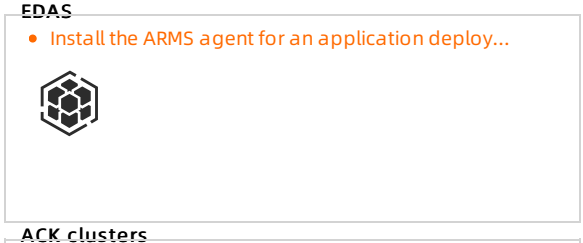
Before you use Application Monitoring of Application Real-Time Monitoring Service (ARMS) to monitor applications, you must install the ARMS agent. Then, you can view a wide variety of metrics in the ARMS console. This topic lists all the topics about how to install the ARMS agent for different applications. The applications can be classified from two dimensions: deployment environment and programming language.

Monitor applications deployed in different environments



EDAS

- Install the ARMS agent for an application deploy...

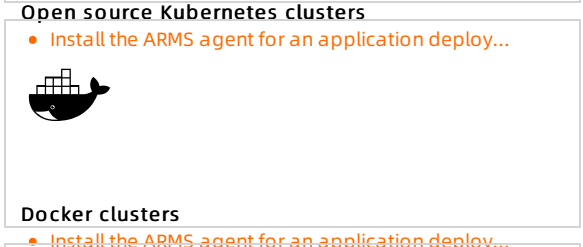


ACK clusters

- Install the ARMS agent for a Java application dep..
- Install the ARMS agent for a PHP application dep...

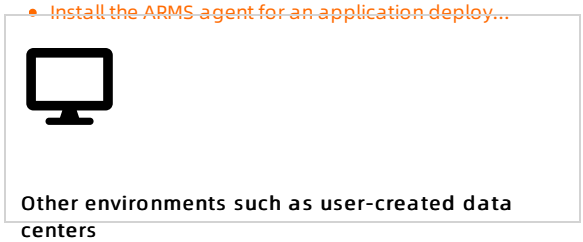
Open source Kubernetes clusters

- Install the ARMS agent for an application deploy...



Docker clusters

- Install the ARMS agent for an application deploy...



Other environments such as user-created data centers

- Manually install the ARMS agent for a Java application
- Automatically install the ARMS agent for a Java application
- Install the ARMS agent for a PHP application
- Install the ARMS agent for a PHP application deployed on multiple servers in standalone mode

Monitor applications developed in different languages

Note The asterisk (*) indicates that Tracing Analysis must be activated.

Monitor Java applications

- Manually install the ARMS agent for a Java application
- Automatically install the ARMS agent for a Java application
- Install the ARMS agent for an application deployed in EDAS
- Install the ARMS agent for an application deployed in a Docker cluster
- Install the ARMS agent for a Java application deployed in an ACK cluster

Monitor PHP applications

- Install the ARMS agent for a PHP application
- Install the ARMS agent for a PHP application deployed on multiple servers in standalone mode
- Install the ARMS agent for a PHP application deployed in an ACK cluster

Monitor Go applications *

- Use Jaeger to report Go application data
- Use Zipkin to report Go application data

Monitor Node.js applications *

- Use Jaeger to report Node.js application data

Monitor .NET applications *

- Use Jaeger to report .NET application data
- Use Zipkin to report .NET application data

Monitor C++ applications *

- Use Jaeger to report C++ application data

2.2. Monitor Java applications

2.2.1. Manually install the ARMS agent for a Java application

After you install the Application Real-Time Monitoring Service (ARMS) agent for a Java application, ARMS starts to monitor the Java application. Then, you can view the monitoring data of the Java application, such as the application topology, traces, abnormal transactions, slow transactions, and SQL analysis. You can install the ARMS agent manually or by using scripts. This topic shows you how to manually install the ARMS agent for a Java application.

Prerequisites

- Make sure that the security group of your ECS instance has opened the TCP outbound permission of

ports 8442, 8443, and 8883. For more information about how to open outbound permissions for ECS, see Add security group rules.

> **Note** ARMS can be connected to not only applications on ECS, but also applications on other servers that can access the Internet.

- Make sure that the third-party components or frameworks you use are within the scope of application monitoring compatibility lists. See ARMS-compatible components and frameworks.
- If the JDK Version is 1.8.0_25 or 1.8.0_31, you may fail to install the arms Agent. In this case, upgrade the JDK version to the latest version, which is 1.8.X.

Procedure

1. Log on to the ARMS console .
2. In the left-side navigation pane, choose **Application Monitoring > Applications**. In the top navigation bar, select a region.
3. On the **Applications** page, click **Add Application** in the upper-right corner.
4. On the **Add Application** page, perform the following operations:
   i. Select **Java** as the **programming language of your application**.
   ii. Select **Default** as the **environment where your application is deployed**.
   iii. Select **Install Manually** as the **method to install the ARMS agent**.
5. Download the ARMS agent and click **Next** in the **Download Agent** step on the **Add Application** page. You can use one of the following methods to download the ARMS agent:
   - Method 1: Manually download the ARMS agent. In the **Download Agent** step, click **Download Agent**.
   - Method 2: Run the **wget** command. Download the installation package based on your region.

     > **Note** Use the public endpoint. If the download fails, use the VPC endpoint.

Region	Download link for the Internet	Download link for VPC
China (Hangzhou)	wget "http://arms-apm-hangzhou.oss-cn-hangzhou.aliyuncs.com/ArmsAgent.zip" -O ArmsAgent.zip	wget "http://arms-apm-hangzhou.oss-cn-hangzhou-internal.aliyuncs.com/ArmsAgent.zip" -O ArmsAgent.zip
China (Shanghai)	wget "http://arms-apm-shanghai.oss-cn-shanghai.aliyuncs.com/ArmsAgent.zip" -O ArmsAgent.zip	wget "http://arms-apm-shanghai.oss-cn-shanghai-internal.aliyuncs.com/ArmsAgent.zip" -O ArmsAgent.zip

Region	Download link for the Internet	Download link for VPC
China (Qingdao)	wget "http://arms-apm-qingdao.oss-cn-qingdao.aliyuncs.com/ArmsAgent.zip" -O ArmsAgent.zip	wget "http://arms-apm-qingdao.oss-cn-qingdao-internal.aliyuncs.com/ArmsAgent.zip" -O ArmsAgent.zip
China (Beijing)	wget "http://arms-apm-beijing.oss-cn-beijing.aliyuncs.com/ArmsAgent.zip" -O ArmsAgent.zip	wget "http://arms-apm-beijing.oss-cn-beijing-internal.aliyuncs.com/ArmsAgent.zip" -O ArmsAgent.zip
China (Zhangjiakou)	wget "http://arms-apm-zhangjiakou.oss-cn-zhangjiakou.aliyuncs.com/ArmsAgent.zip" -O ArmsAgent.zip	wget "http://arms-apm-zhangjiakou.oss-cn-zhangjiakou-internal.aliyuncs.com/ArmsAgent.zip" -O ArmsAgent.zip
China (Shenzhen)	wget "http://arms-apm-shenzhen.oss-cn-shenzhen.aliyuncs.com/ArmsAgent.zip" -O ArmsAgent.zip	wget "http://arms-apm-shenzhen.oss-cn-shenzhen-internal.aliyuncs.com/ArmsAgent.zip" -O ArmsAgent.zip
China (Hong Kong)	wget "http://arms-apm-hongkong.oss-cn-hongkong.aliyuncs.com/ArmsAgent.zip" -O ArmsAgent.zip	wget "http://arms-apm-hongkong.oss-cn-hongkong-internal.aliyuncs.com/ArmsAgent.zip" -O ArmsAgent.zip
Singapore (Singapore)	wget "http://arms-apm-ap-southeast.oss-ap-southeast-1.aliyuncs.com/cloud_ap-southeast-1/ArmsAgent.zip" -O ArmsAgent.zip	wget "http://arms-apm-ap-southeast.oss-ap-southeast-1-internal.aliyuncs.com/cloud_ap-southeast-1/ArmsAgent.zip" -O ArmsAgent.zip

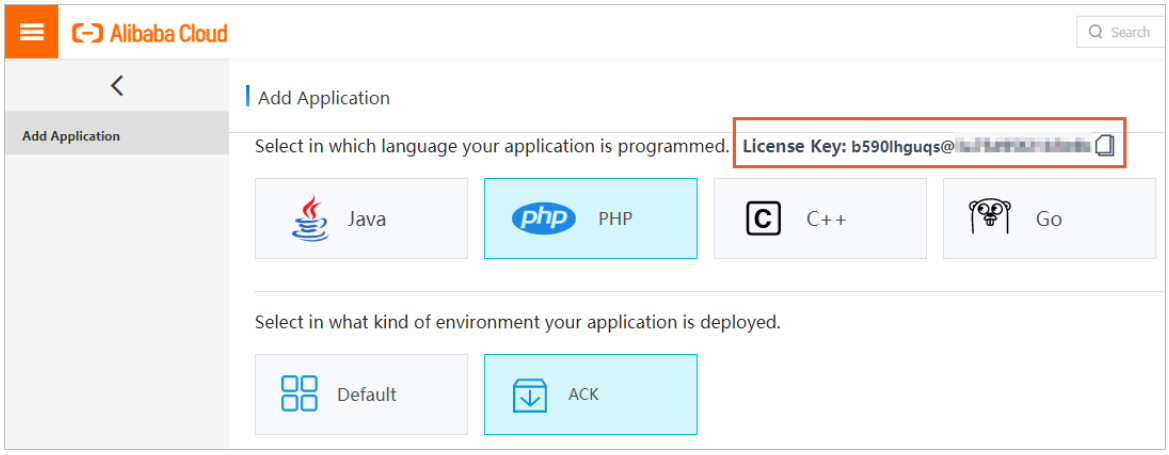
Region	Download link for the Internet	Download link for VPC
Japan (Tokyo)	wget "http://arms-apm-japan.oss-ap-northeast-1.aliyuncs.com/ArmsAgent.zip" -O ArmsAgent.zip	wget "http://arms-apm-japan.oss-ap-northeast-1-internal.aliyuncs.com/ArmsAgent.zip" -O ArmsAgent.zip
US (Silicon Valley)	wget "http://arms-apm-usw.oss-us-west-1.aliyuncs.com/ArmsAgent.zip" -O ArmsAgent.zip	wget "http://arms-apm-usw.oss-us-west-1-internal.aliyuncs.com/ArmsAgent.zip" -O ArmsAgent.zip
China East 1 Finance	wget "http://arms-apm-hangzhou.oss-cn-hangzhou.aliyuncs.com/finance/ArmsAgent.zip" -O ArmsAgent.zip	wget "http://arms-apm-hangzhou.oss-cn-hangzhou-internal.aliyuncs.com/finance/ArmsAgent.zip" -O ArmsAgent.zip
China East 2 Finance	wget "http://arms-apm-sh-finance-1.oss-cn-shanghai-finance-1.aliyuncs.com/ArmsAgent.zip" -O ArmsAgent.zip	wget "http://arms-apm-sh-finance-1.oss-cn-shanghai-finance-1-internal.aliyuncs.com/ArmsAgent.zip" -O ArmsAgent.zip
China South 1 Finance	wget "http://arms-apm-sz-finance.oss-cn-shenzhen-finance-1.aliyuncs.com/ArmsAgent.zip" -O ArmsAgent.zip	wget "https://arms-apm-sz-finance.oss-cn-shenzhen-finance-1-internal.aliyuncs.com/ArmsAgent.zip" -O ArmsAgent.zip

6. Go to the directory of the installation package. Run the following command to decompress the installation package to a working directory:

```
unzip ArmsAgent.zip -d /{user.workspace}/
```

> **Note** *{user.workspace}* is a sample directory. Replace it with an actual directory.

7. Copy the license key in the upper part of the **Add Application** page.



8. Use one of the following methods to add the AppName and LicenseKey parameters:

> **Note** Replace `{LicenseKey}` in the sample code with your license key. Replace `{AppName}` with your application name. The application name cannot contain Chinese characters. Replace `{user.workspace}` with the directory where the ARMS agent is decompressed. Replace demoApp.jar with the path to the actual JAR package.

   - Method 1: Edit the JVM parameters based on the runtime environment of your application.

Runtime environment	Procedure
Tomcat on Linux or macOS	Append the following configurations to the *{TOMCAT_HOME}/bin/setenv.sh* file: `JAVA_OPTS="$JAVA_OPTS -javaagent:/{user.workspace}/ArmsAgent/arms-bootstrap-1.7.0-SNAPSHOT.jar -Darms.licenseKey={LicenseKey} -Darms.appName={AppName}"` If your Tomcat does not contain the *setenv.sh* configuration file, open the *{TOMCAT_HOME}/bin/catalina.sh* file and append the preceding configurations to the JAVA_OPTS parameter. For more information, see Row 256 in the catalina.sh file.

Tomcat on Windows	Append the following configurations to the *{TOMCAT_HOME}/bin/catalina.bat* file:
	`set "JAVA_OPTS=%JAVA_OPTS% -javaagent:/{user.workspace}/ArmsAgent/arms-bootstrap-1.7.0-SNAPSHOT.jar -Darms.licenseKey={LicenseKey} -Darms.appName={AppName}"`
	If the preceding setting does not take effect, append the following configurations to the *{TOMCAT_HOME}/bin/catalina.bat* file:
	`set "CATALINA_OPTS=-javaagent:/{user.workspace}/arms-bootstrap-1.7.0-SNAPSHOT.jar -Darms.licenseKey={LicenseKey} -Darms.appName={AppName}"`

Runtime environment	Procedure

Jetty	Append the following configurations to the *{JETTY_HOME}/start.ini* configuration file:
	`--exec`
	`-javaagent:/{user.workspace}/ArmsAgent/arms-bootstrap-1.7.0-SNAPSHOT.jar`
	`-Darms.licenseKey={LicenseKey}`
	`-Darms.appName={AppName}`

Spring Boot	When you start the Spring Boot process, append the -javaagent parameter to the startup command: `java -javaagent:/{user.workspace}/ArmsAgent/arms-bootstrap-1.7.0-SNAPSHOT.jar -Darms.licenseKey={LicenseKey} -Darms.appName={AppName} -jar demoApp.jar`

Resin	When you start the Resin process, append the following tag to the *conf/resion.xml* configuration file:
	`<server-default>`
	`  <jvm-arg>-javaagent:{user.workspace}/ArmsAgent/arms-bootstrap-1.7.0-SNAPSHOT.jar</jvm-arg>`
	`  <jvm-arg>-Darms.licenseKey={LicenseKey}</jvm-arg>`
	`  <jvm-arg>-Darms.appName={AppName}</jvm-arg>`
	`</server-default>`
	Append the following tag to the *conf/app-default.xml* file:
	`<library-loader path="{user.workspace}/ArmsAgent/plugin"/>`

Windows	When you run a CMD command to start the Java process, use a backslash (\) as the delimiter in the mount path of the ARMS agent. `java -javaagent:\{user.workspace}\ArmsAgent\arms-bootstrap-1.7.0-SNAPSHOT.jar -Darms.licenseKey={LicenseKey} -Darms.appName={AppName} -jar {user.workspace}\demoApp.jar`

To deploy multiple instances of the same application on a server, you can differentiate the JVM processes by setting the -Darms.agentId parameter to a logical number. Example:

```
java -javaagent:/{user.workspace}/ArmsAgent/arms-bootstrap-1.7.0-SNAPSHOT.jar -Darms.licenseKey={LicenseKey} -Darms.appName={AppName} -Darms.agentId=001 -jar demoApp.jar
```

- Method 2:
  a. Append the following configurations to the *arms-agent.config* file:

```
arms.licenseKey={LicenseKey} arms.appName={AppName}
```

b. Append the following parameter to the startup script of the Java application:

```
-javaagent:/{user.workspace}/ArmsAgent/arms-bootstrap-1.7.0-SNAPSHOT.jar
```

9. Restart the Java application.

Verification

After about 1 minute, if your application is displayed in the application list and some data records are sent, your application is monitored by ARMS.

Uninstall the ARMS agent

If you no longer need ARMS to monitor your Java application, perform the following steps to uninstall the ARMS agent.

1. Delete all parameters that you added in Step 8, such as {AppName} and {LicenseKey}.
2. Restart the Java application.

Change the application name

If you forget to change the sample name Java-Demo to a custom name, you can change the application name by performing a few operations. You do not need to restart the application or reinstall the ARMS agent. For more information, see How do I modify the name of a common Java application on which the ARMS agent is manually installed?.

Related information

- FAQ

2.2.2. Install the ARMS agent for a Java application by using scripts

You can use scripts to install the Application Real-Time Monitoring Service (ARMS) agent for a Java application. To monitor the application, you do not need to restart the application. We recommend that you choose this installation method if you are using ARMS for the first time. If you restart the Java application, ARMS automatically loads the ARMS agent and monitors the application.

Prerequisites

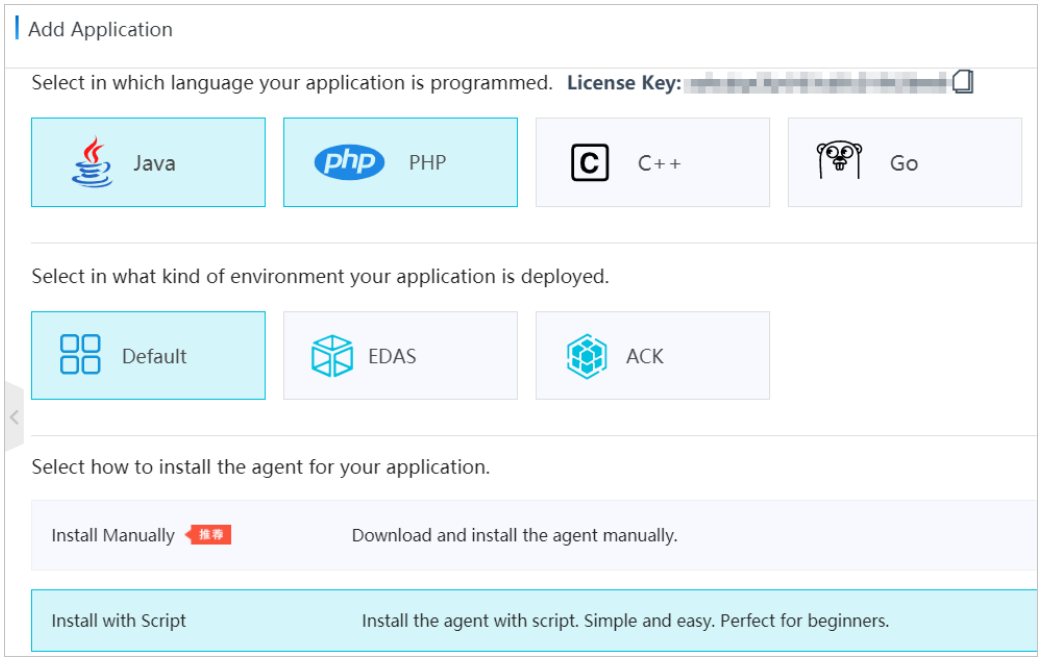
- Make sure that the security group of your ECS instance has opened the TCP outbound permission of ports 8442, 8443, and 8883. For more information about how to open outbound permissions for ECS, see Add security group rules.

  > **Note** ARMS can be connected to not only applications on ECS, but also applications on other servers that can access the Internet.

- Make sure that the third-party components or frameworks you use are within the scope of application monitoring compatibility lists. See ARMS-compatible components and frameworks.
- If you have manually installed the ARMS agent for the Java application, you must uninstall the ARMS agent before you can use scripts to install the ARMS agent. For more information, see Manually install the ARMS agent for a Java application.
- If the JDK Version is 1.8.0_25 or 1.8.0_31, you may fail to install the arms Agent. In this case, upgrade the JDK version to the latest version, which is 1.8.X.

Procedure

1. Log on to the ARMS console .
2. In the left-side navigation pane, choose **Application Monitoring > Applications**.
3. On the **Applications** page, select a region in the top navigation bar, and click **Add Application** in the upper-right corner.
4. On the **Add Application** page, perform the following operations:
   i. Select **Java** as the **programming language of your application**.
   ii. Select **Default** as the **environment where your application is deployed**.
   iii. Select **Install with Script** as the **method to install the ARMS agent**.



5. Copy the license key at the top of the Add Application page.
6. Run the installation script that corresponds to your region.

Region	Installation script
China (Hangzhou)	wget -O- http://arms-apm-hangzhou.oss-cn-hangzhou.aliyuncs.com/install.sh | sh && ~/.arms/supervisor/cli.sh <licenseKey> Java-Demo
China (Shanghai)	wget -O- http://arms-apm-shanghai.oss-cn-shanghai.aliyuncs.com/install.sh | sh && ~/.arms/supervisor/cli.sh <licenseKey> Java-Demo
China (Qingdao)	wget -O- http://arms-apm-qingdao.oss-cn-qingdao.aliyuncs.com/install.sh | sh && ~/.arms/supervisor/cli.sh <licenseKey> Java-Demo

Region	Installation script
China (Beijing)	wget -O- http://arms-apm-beijing.oss-cn-beijing.aliyuncs.com/install.sh | sh && ~/.arms/supervisor/cli.sh <licenseKey> Java-Demo
China (Shenzhen)	wget -O- http://arms-apm-shenzhen.oss-cn-shenzhen.aliyuncs.com/install.sh | sh && ~/.arms/supervisor/cli.sh <licenseKey> Java-Demo
China (Hong Kong)	wget -O- http://arms-apm-hongkong.oss-cn-hongkong.aliyuncs.com/install.sh | sh && ~/.arms/supervisor/cli.sh <licenseKey> Java-Demo
Singapore (Singapore)	wget -O- http://arms-apm-ap-southeast.oss-ap-southeast-1.aliyuncs.com/cloud_ap-southeast-1/install.sh | sh && ~/.arms/supervisor/cli.sh <licenseKey> Java-Demo
Japan (Tokyo)	wget -O- http://arms-apm-japan.oss-ap-northeast-1.aliyuncs.com/install.sh | sh && ~/.arms/supervisor/cli.sh <licenseKey> Java-Demo
US (Silicon Valley)	wget -O- http://arms-apm-usw.oss-us-west-1.aliyuncs.com/install.sh | sh && ~/.arms/supervisor/cli.sh <licenseKey> Java-Demo
China East 1 Finance	wget -O- http://arms-apm-hangzhou.oss-cn-hangzhou.aliyuncs.com/finance/install.sh | sh && ~/.arms/supervisor/cli.sh <licenseKey> Java-Demo

Note

- Replace `<licenseKey>` with your license key.
- Replace `Java-Demo` with the name of your application. The application name cannot contain Chinese characters.
- After you run the installation script, it automatically downloads the latest ARMS agent.
- If your server has only one Java process, the installation script installs the ARMS agent on this process by default. If your server has multiple Java processes, select a process to install the ARMS agent.

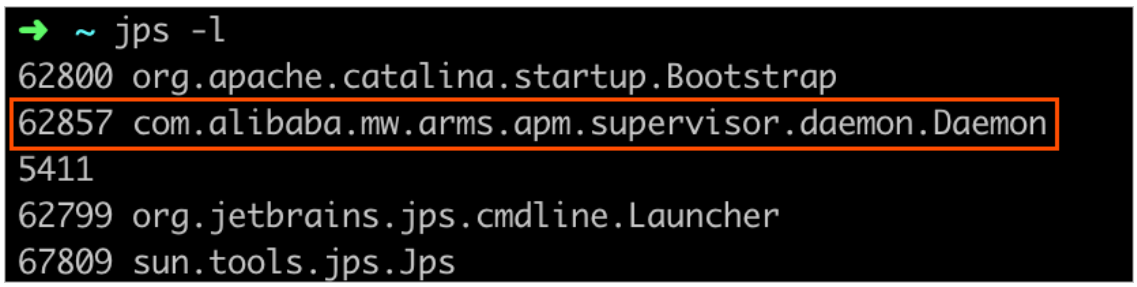
Verify the result

After about 1 minute, if your application is displayed in the application list and some data records are sent, it indicates that your application is monitored by ARMS.

Uninstall the ARMS Agent

1. If you no longer want to use ARMS to monitor your Java applications, run the `jps -l` command to view all processes and find the process ID of `com.alibaba.mw.arms.apm.supervisor.daemon.Daemon` in the returned results.

   In this example, the process ID of `com.alibaba.mw.arms.apm.supervisor.daemon.Daemon` is 62857.

2. Run the `kill -9 <process ID>` command.

   Example: `kill -9 62857` .

3. Run the `rm -rf /.arms /root/.arms` command.
4. Restart your application.

Change the application name

If you forget to change the sample name Java-Demo to a custom name, you can change the application name by performing a few operations. You do not need to restart the application or reinstall the ARMS agent. For more information, see How can I change the name of a Java application after I use scripts to install the ARMS agent?

FAQ

1. How can I handle the following getcwd error when I run the installation script to install the ARMS agent?

   ```
   shell-init: error retrieving current directory: getcwd: cannot access parent directories: No such file or directory Error occurred during initialization of VM java.lang.Error: Properties init: Could not determine current working directory. at java.lang.System.initProperties(Native Method) at java.lang.System.initializeSystemClass(System.java:1119)
   ```

   The possible cause is that the current directory is deleted by mistake when you run the installation script. To solve this issue, run the `cd` command and then run the installation script again.

2. Where do I view logs after I run the installation script to install the ARMS agent?
   The default directory of logs is */root/.arms/supervisor/logs/arms-supervisor.log*. If no logs are available in this directory, run the `ps -ef |grep arms` command to view the directory where logs are stored.

Related information

- FAQ

2.2.3. Enable ARMS to monitor an EDAS application

You can use Application Real-Time Monitoring Service (ARMS) to monitor applications that are deployed in Enterprise Distributed Application Service (EDAS). ARMS allows you to monitor applications based on various performance metrics, such as topology, API requests, abnormal transactions, slow transactions, and SQL analysis. To enable ARMS to monitor an EDAS application, you only need to perform a few operations in the EDAS console.

Prerequisites

If the JDK Version is 1.8.0_25 or 1.8.0_31, you may fail to install the arms Agent. In this case, upgrade the JDK version to the latest version, which is 1.8.X.

Enable ARMS to monitor an application in the EDAS console

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 in the top navigation bar and click the name of the application to be monitored by ARMS.
4. In the left-side navigation pane, choose **Application Monitoring > Advanced Monitoring** or **Monitoring > Advanced Monitoring**. Then, click **Enable Advanced Application Monitoring**.
5. In the **Confirm** dialog box, click **Confirm**.

View the monitoring data of the EDAS application in ARMS

After you enable ARMS to monitor the application, click **Go to ARMS Application Monitoring**. The **Applications** page of the ARMS console appears. On the **Applications** page, click the name of the application to view the detailed monitoring data.

After you enable ARMS to monitor the EDAS application, you can use multiple application monitoring capabilities, as shown in the following examples.

For more information about ARMS application monitoring, see Overview.

2.2.4. Install the ARMS agent for a Java application deployed in Container Service for Kubernetes

After you install the Application Real-Time Monitoring Service (ARMS) agent for a Java application that is deployed in Container Service for Kubernetes, ARMS starts to monitor the Java application. You can view the monitoring data of application topology, API requests, abnormal transactions, slow transactions, and SQL analysis. This topic describes how to install the ARMS agent for a Java application that is deployed in Container Service for Kubernetes.

Prerequisites

- Create a dedicated Kubernetes cluster
- Create a namespace: The namespace in this example is *arms-demo*.
- If the JDK Version is 1.8.0_25 or 1.8.0_31, you may fail to install the arms Agent. In this case, upgrade the JDK version to the latest version, which is 1.8.X.

Install the ARMS application monitoring agent

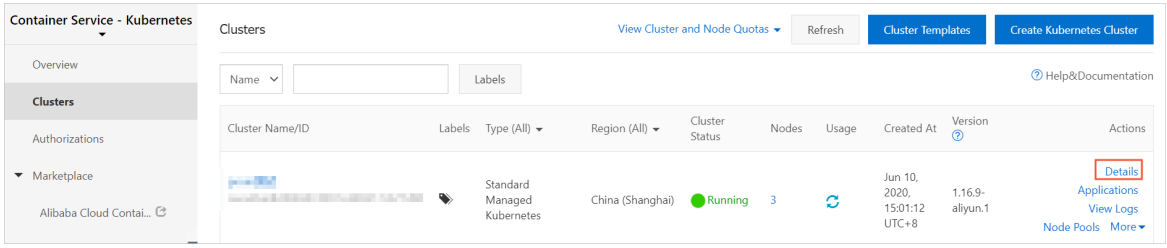
Install the ARMS application monitoring components ack-arms-pilot.

1. Log on to the Alibaba Cloud Container Service for Kubernetes console .
2. In the left-side navigation pane, click **Latest version > App Catalog** . On the right page, click **ack-arms-pilot** .
3. Log on to the **App Catalog-ack-arms-pilot** On the page, on the right **Creation** Panel, select the cluster created in prerequisites, and click **Creation** .

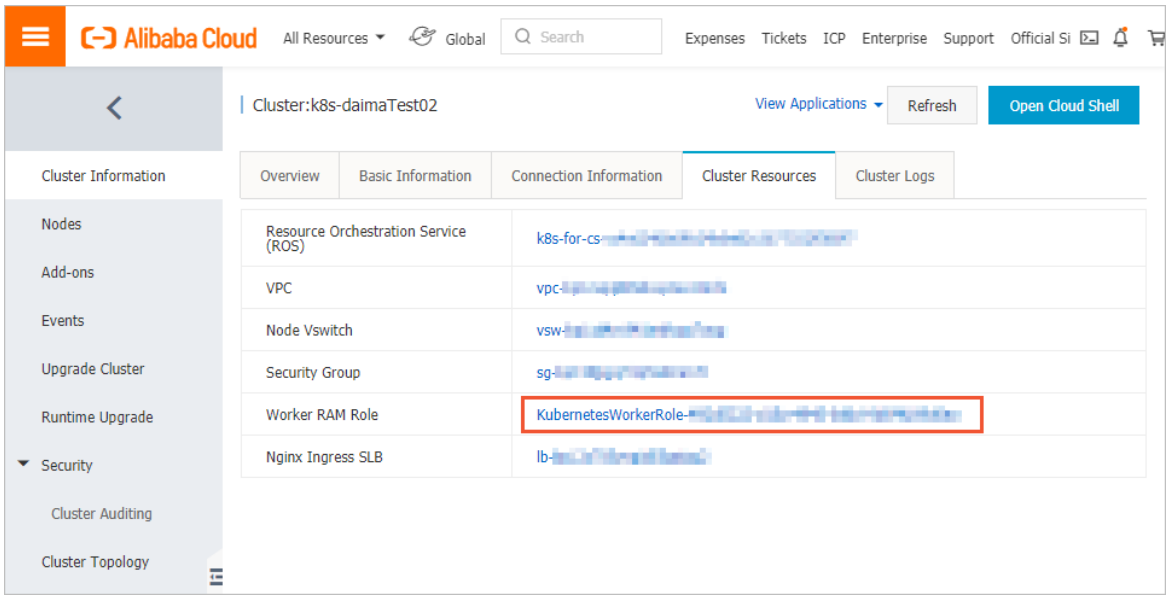
Authorize ACK to access ARMS

Grant Alibaba Cloud Container Service for Kubernetes access permissions on ARMS resources.

1. Log on to the Log on to the Container Service console. .
2. In the left-side navigation pane, click **cluster** , in **Clusters** Page on the right of the target cluster **Action** Column click **View Details** .

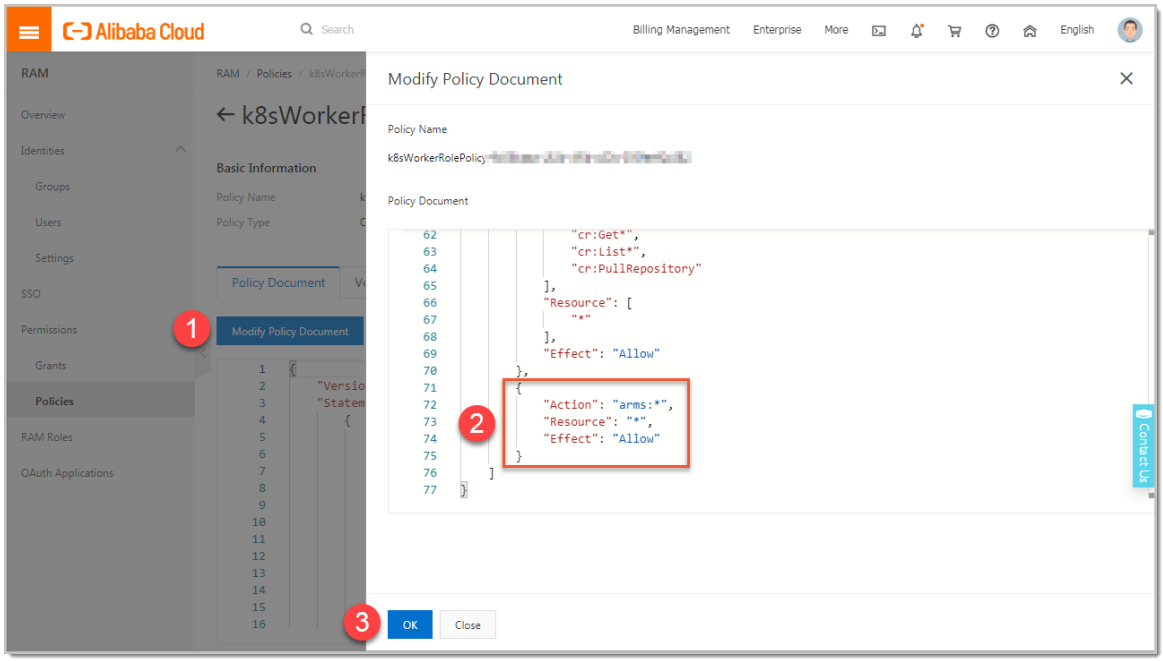


3. In the destination cluster, **Create CDH Cluster Configuration** On the page, click **Cluster Resources** Tab, and then click **Worker RAM Role** Link on the right.



4. In the Resource Access Management RAM console, **RAM Roles** Page, click **Permission management** The name of the permission policy on the tab.
5. Log on to the **Policy Document** On the tab, click **Modify Policy Document** , and add the following to the **Policy Document** In, finally click **OK** .

```
{"Action": "arms:*", "Resource": "*", "Effect": "Allow"}
```



Enable ARMS application monitoring for Java applications

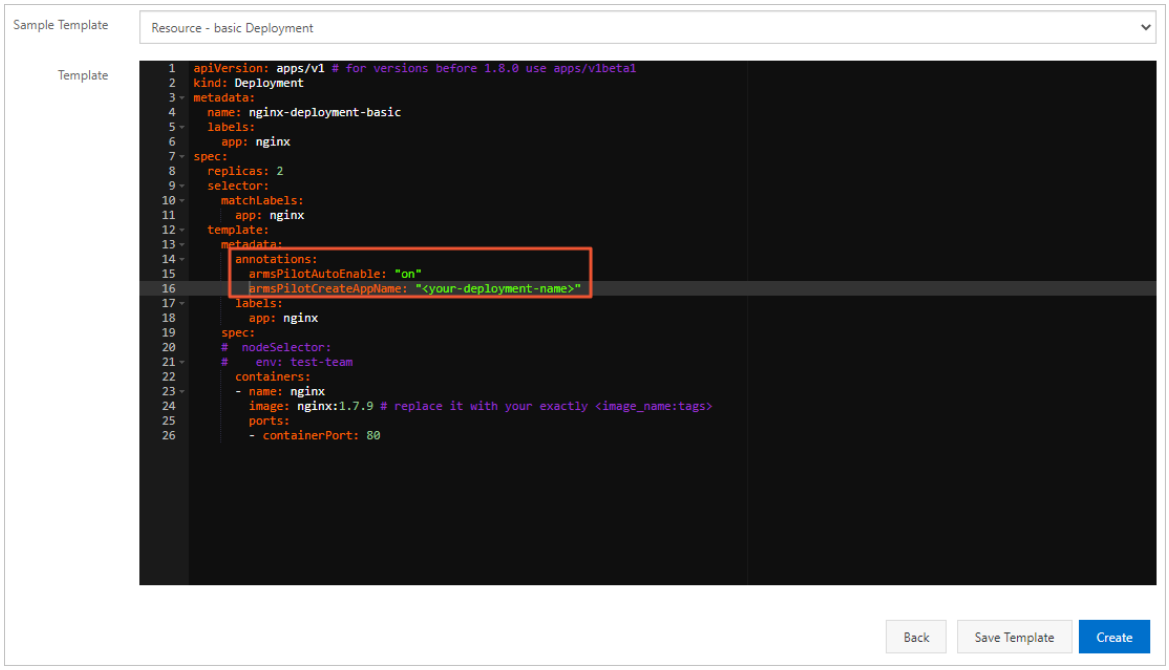
The following steps describe how to enable ARMS application monitoring for a newly created or existing application.

To enable ARMS application monitoring when you create an application, perform the following steps.

1. Log on to the Alibaba Cloud Container Service for Kubernetes console In the left-side navigation pane, choose **cluster** , in **Clusters** Page on the right of the target cluster **Action** Column click **Application management** .
2. Log on to the **Deployment** In the upper-right corner of the page, click **Use a template to create** .
3. Log on to the **Use a template to create** Selection on page **Sample Template** , and in **template** (YAML format) the following `annotations` Add to *spec > template > metadata* Under the level.

> **Note** Please *<your-deployment-name>* With the name of your application.

```
annotations: armsPilotAutoEnable: "on" armsPilotCreateAppName: "<your-deployment-name>"
```

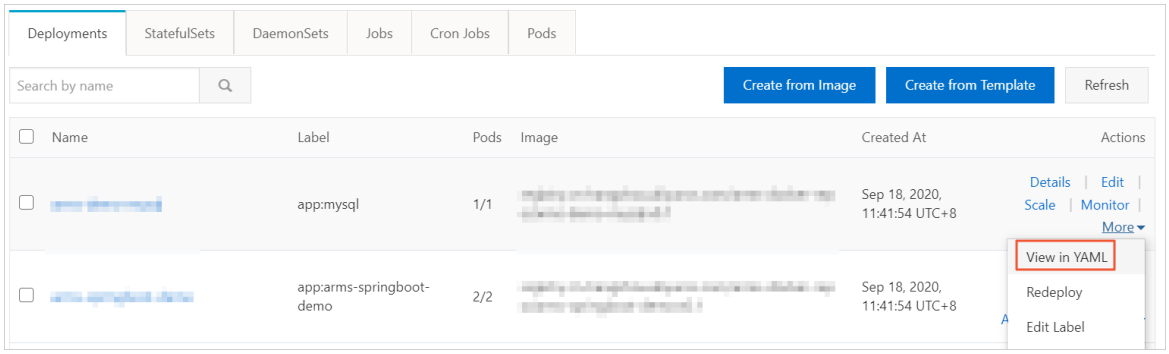


The following YAML template shows how to create a stateless application and enable ARMS for the application.

Show the complete YAML file (Java)

To enable ARMS application monitoring for an existing application, perform the following steps.

1. Log on to the Alibaba Cloud Container Service for Kubernetes console In the left-side navigation pane, choose **cluster** , in **Clusters** Page on the right of the target cluster **Action** Column click **Application management** .
2. Log on to the **Deployment** Or **StatefulSet** Right side of target application on page **Action** Column selection **The Hide/Show icon > View Yaml** .



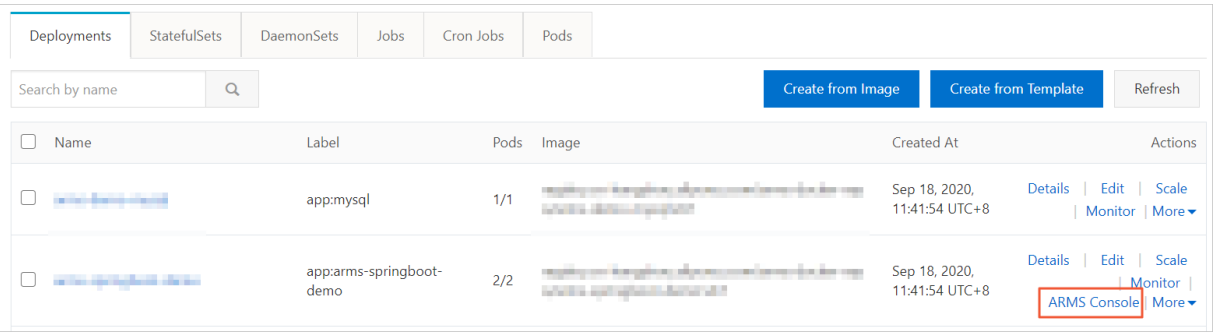
3. Log on to the **Edit YAML** In the dialog box, the following `annotations` Add to *spec > template > metadata* Level, and click **Update** .

> **Note** Please *<your-deployment-name>* With the name of your application.

```
annotations: armsPilotAutoEnable: "on" armsPilotCreateAppName: "<your-deployment-name>"
```

Execution result

On the **Deployments** or **StatefulSets** tab, **ARMS Console** appears in the **Actions** column of the application.



Note If you cannot find **ARMS Console** in the **Actions** column, check whether you have authorized Container Service to access ARMS.

Uninstall the ARMS agent

1. Log on to the Alibaba Cloud Container Service for Kubernetes console .
2. In the left-side navigation pane, click **Clusters**. On the **Clusters** page, click **Applications** in the **Actions** column corresponding to the cluster that contains the Java application from which you want to uninstall the ARMS agent.
3. In the left-side navigation pane, select **Releases**.
4. On the **Helm** tab, select the release name **arms-pilot** of the ARMS agent, and click **Delete** in the **Actions** column.
5. In the **Delete** dialog box, click **OK**.
6. Restart your business pod.

Change the application name

You can change the application name without restarting the application or reinstalling the agent. For example, if you forget to change the sample name Java-Demo to a custom name, you can edit the armsPilotCreateAppName parameter in the Deployment application and then restart the pod. For more information, see How can I change the name of a Java application that is deployed in a Container Service for Kubernetes cluster?

Related information

- Create a dedicated Kubernetes cluster
- Create a namespace
- Install the ARMS agent for a PHP application deployed in Container Service for Kubernetes

2.2.5. Install the ARMS agent for Java applications in Function Compute

After you install the Application Real-Time Monitoring Service (ARMS) agent, you can use it to monitor Java applications in Function Compute. You can view the monitoring data of application topology, API requests, abnormal transactions, and slow transactions. This topic describes how to install the ARMS agent for Java applications in Function Compute.

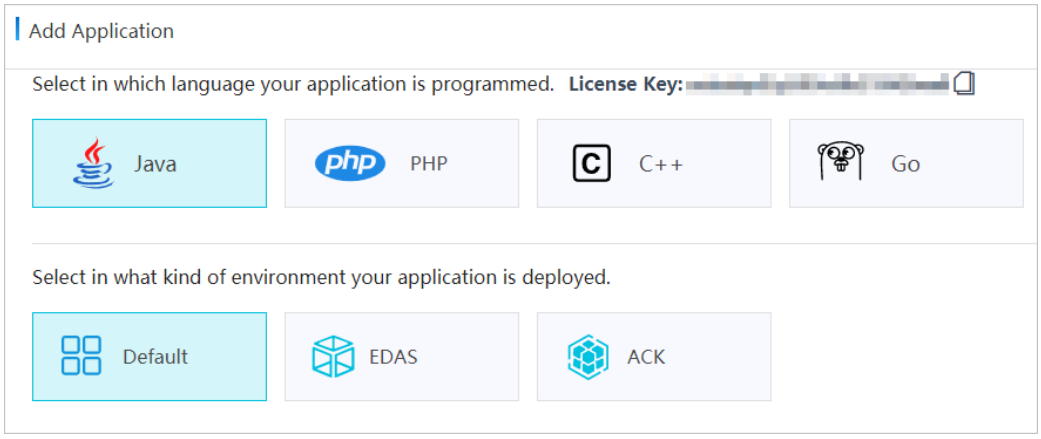
Prerequisites

Prerequisites

Create a function in the Function Compute console

Obtain a License Key

1. Log on to the ARMS console. In the left-side navigation pane, choose **Application Monitoring > Applications**.
2. On the **Applications** page, select the destination region in the top navigation bar, and click **Add Application** in the upper-right corner.
3. At the top of the **Add Application** page, click the copy icon to the right of the License Key and save the Key.



Procedure

1. Log on to the Function Compute console.
2. In the top navigation bar, select a region.
3. In the left-side navigation pane, click **Service/Function**. On the **Service/Function** page, find the function, and click **Modify Configurations** in the Actions column.
4. On the **Modify Configurations** page, set Environment Variables to Key value.
5. Set the key to *FC_EXTENSIONS_ARMS_LICENSE_KEY*, set the value to the License Key that is obtained in Obtain a License Key, and then click **Submit**.

Verify the result

1. Log on to the ARMS console. In the left-side navigation pane, choose **Application Monitoring > Applications**.
2. At the top of the **Applications** page, select the destination region.

If your Java application is displayed in the application list and data is imported after you call the function multiple times in the Function Compute console, the Java application is connected to ARMS.

Related information

- Create a function in the Function Compute console

2.2.6. Install the ARMS agent for an application deployed in an open source Kubernetes environment

You can use Application Real-Time Monitoring Service (ARMS) to monitor applications that are deployed in open source Kubernetes environments. ARMS allows you to monitor applications based on various performance metrics, such as topology, API requests, abnormal transactions, slow transactions, and SQL analysis. This topic describes how to enable ARMS to monitor an application that is deployed in an open source Kubernetes environment.

Prerequisites

- ARMS is activated. For more information, see Activate ARMS.
- Helm is installed. For more information, see Installing Helm.
- The version of your Kubernetes api-server is V1.10 or later.
- Your cluster is accessible over the Internet.
- If the JDK Version is 1.8.0_25 or 1.8.0_31, you may fail to install the arms Agent. In this case, upgrade the JDK version to the latest version, which is 1.8.X.

ARMS can monitor only the following two types of applications: Deployment and StatefulSet. To enable ARMS to monitor a Deployment application that is deployed in an open source Kubernetes environment, perform the following steps:

Step 1: Install the ARMS agent

1. Download the arms-pilot installation package by using one of the following methods:
   - Method 1: Download arms-pilot installation package.
   - Method 2: Run the following `wget` command to download the arms-pilot installation package:

     ```
     wget 'http://arms-apm-hangzhou.oss-cn-hangzhou.aliyuncs.com/arms-pilot/arms-pilot-0.1.1-community.tgz' -O arms-pilot-0.1.1.tgz
     ```
2. Run the following command to decompress the arms-pilot installation package:

   ```
   tar zxvf arms-pilot-0.1.1.tgz
   ```
3. Run the following command to install arms-pilot:

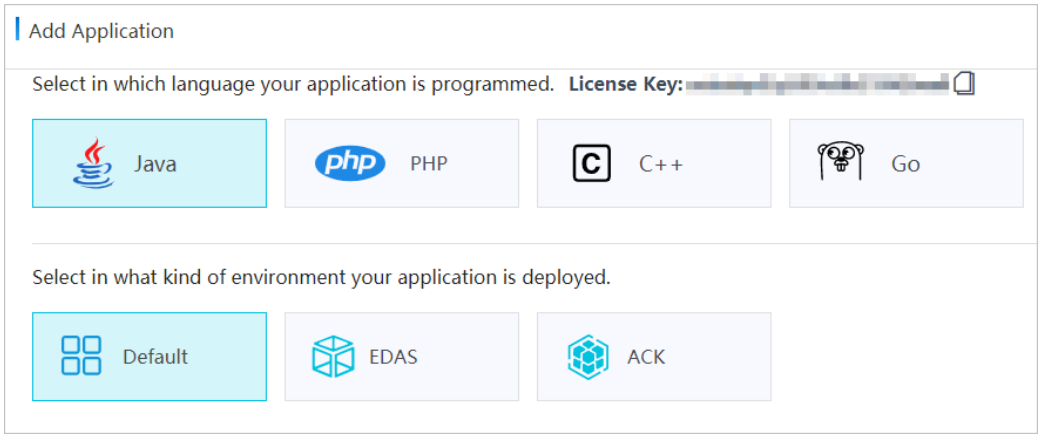
   ```
   helm install ./arms-pilot --namespace arms-pilot-system
   ```

Step 2: Obtain the license key

1. Log on to the ARMS console. In the left-side navigation pane, choose **Application Monitoring > Applications**.
2. On the **Applications** page, select China (Hangzhou) in the top navigation bar, and click **Add Application** in the upper-right corner.

> **Notice** By default, applications that are deployed in open source Kubernetes environments reside in the China (Hangzhou) region. Therefore, you must obtain the license key for the China (Hangzhou) region.

3. Copy the license key at the top of the **Add Application** page.



Step 3: Edit the YAML file of the application

1. Run the following command to view the configurations of the Deployment application:

```
### Run the following command to view the configurations of a specified Deployment application: kubectl get deployment {Name of the Deployment application} -o yaml
```

> **Note** If you do not know the `{Name of the Deployment application}`, run the following command to view all Deployment applications. Then you can find the target Deployment application in the results, and view the application configurations.
>
> ```
> ### Run the following command to view the configurations of all Deployment applications: kubectl get deployments --all-namespace
> ```

2. Run the following command to edit the YAML file of the Deployment application:

```
kubectl edit deployment {Name of the Deployment application} -o yaml
```

3. In the YAML file, go to the *spec -> template -> metadata -> labels* directory and append the following content:

```
ARMSApmAppName: xxx
ARMSApmLicenseKey: xxx
```

> **Notice**
>
> - Replace `xxx` with your application name and license key. The application name cannot contain Chinese characters.
> - Replace the at sign (`@`) in the license key with an underscore (`_`).

The following example shows a complete YAML file for creating a Deployment application in an open source environment and enabling ARMS to monitor the application.

```
apiVersion: apps/v1beta1 # for versions before 1.8.0 use apps/v1beta1
kind: Deployment
metadata:
 name: arms-springboot-demo
 labels:
  app: arms-springboot-demo
spec:
 replicas: 2
 selector:
  matchLabels:
   app: arms-springboot-demo
 template:
  metadata:
   labels:
    app: arms-springboot-demo
    ARMSApmLicenseKey: "xxx_xxx"
    ARMSApmAppName: "arms-k8s-demo"
  spec:
   containers:
    - resources:
       limits:
        cpu: 0.5
      image: registry.cn-hangzhou.aliyuncs.com/arms-docker-repo/arms-springboot-demo:v0.1
      imagePullPolicy: Always
      name: arms-springboot-demo
      env:
       - name: MYSQL_SERVICE_HOST
         value: "arms-demo-mysql"
       - name: MYSQL_SERVICE_PORT
         value: "3306"
---
apiVersion: apps/v1beta1 # for versions before 1.8.0 use apps/v1beta1
kind: Deployment
metadata:
 name: arms-demo-mysql
 labels:
  app: mysql
spec:
 replicas: 1
 selector:
  matchLabels:
   app: mysql
 template:
  metadata:
   labels:
    app: mysql
  spec:
   containers:
    - resources:
       limits:
        cpu: 0.5
      image: registry.cn-hangzhou.aliyuncs.com/arms-docker-repo/arms-demo-mysql:v0.1
      name: mysql
      ports:
```

```
        - containerPort: 3306
          name: mysql
---
apiVersion: v1
kind: Service
metadata:
 labels:
   name: mysql
 name: arms-demo-mysql
spec:
 ports:
   # the port that this service should serve on
   - name: arms-mysql-svc
     port: 3306
     targetPort: 3306
 # label keys and values that must match in order to receive traffic for this service
 selector:
   app: mysql
---
```

4. After the preceding configurations are saved, the application automatically restarts and then the configurations take effect.
   After 2 to 5 minutes, if your application is displayed on the **Application Monitoring > Applications** page in the ARMS console and some data records are sent, it indicates that your application is monitored by ARMS.

Uninstall the ARMS agent

1. If you no longer want to use ARMS to monitor your Java applications in an open-source Kubernetes environment, run the following command to uninstall arms-pilot:

   ```
   helm del --purge arms-pilot
   ```

2. Restart your business pod.

Related information

- FAQ

2.2.7. Install the ARMS agent for a Java application deployed in a Docker cluster

After you install the Application Real-Time Monitoring Service (ARMS) agent for a Java application that is deployed in a Docker cluster, ARMS starts to monitor the Java application. ARMS automatically adapts to the environment where the application runs. You do not need to set up the runtime environment for Tomcat, Jetty, or Spring Boot applications. This topic describes how to install the ARMS agent for a Java application that is deployed in a Docker cluster.

Prerequisites

- ARMS is activated. For more information, see Activate and upgrade ARMS.
- A Java application is deployed in a Docker cluster.

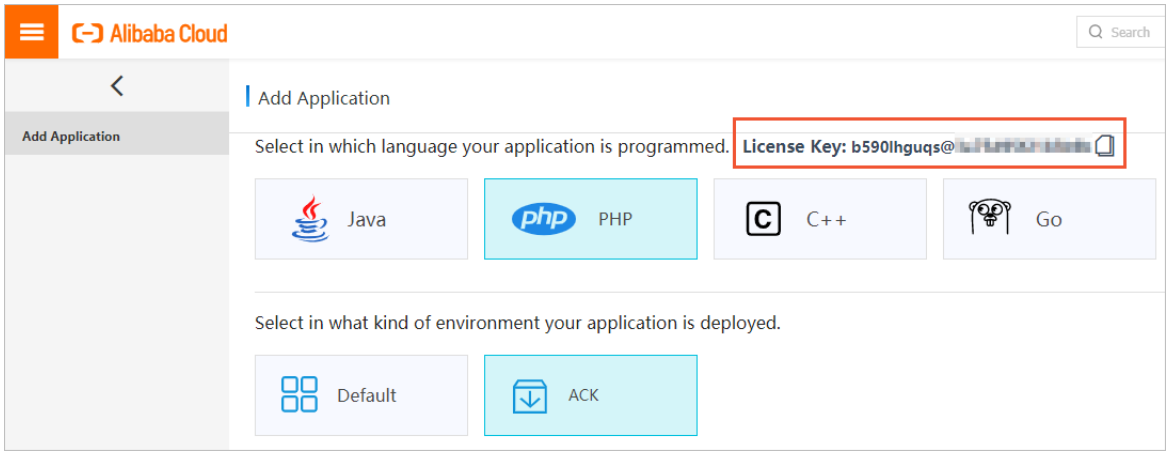
- If the JDK Version is 1.8.0_25 or 1.8.0_31, you may fail to install the arms Agent. In this case, upgrade the JDK version to the latest version, which is 1.8.X.

Context

ARMS Application Monitoring allows you to monitor Java applications by using the *{original-docker-image:tag}* image. To do so, edit the *Dockerfile* file to integrate an existing image. Then, build and start a new image.

Step 1: Obtain the license key

1. Log on to the ARMS console .
2. In the left-side navigation pane, choose **Application Monitoring > Applications**. In the top navigation bar, select a region.
3. On the **Applications** page, click **Add Application** in the upper-right corner.
4. Copy the license key at the top of the **Add Application** page.



Step 2: Integrate an existing image

Edit the *Dockerfile* file to integrate the *{original-docker-image:tag}* image, as shown in the following example.

```
###################################
##                ###
##    ARMS APM DEMO Docker   ###
##       For Java        ###
##    withAgent  V0.1     ###
##                ###
###################################
# Replace {original-docker-image:tag} with your own image address.
FROM {original-docker-image:tag}
WORKDIR /root/
# Replace the link for downloading the ARMS agent based on your region.
RUN wget "http://arms-apm-hangzhou.oss-cn-hangzhou.aliyuncs.com/ArmsAgent.zip" -O ArmsAgent.zip
RUN unzip ArmsAgent.zip -d /root/
# Obtain the license key, as shown in Step 1.
# {AppName} is the name of the application that is monitored by ARMS. The application name cannot contai
n Chinese characters.
# If all images are connected to the same application monitoring job, you only need to specify the arms_licen
seKey and arms_appName parameters.
# To connect the image to another application monitoring job, run the docker run command and use the -e p
arameter to specify the arms_licenseKey and arms_appName parameters. This overwrites the configuration
s in the Dockerfile file.
ENV arms_licenseKey={LicenseKey}
ENV arms_appName={AppName}
ENV JAVA_TOOL_OPTIONS ${JAVA_TOOL_OPTIONS} '-javaagent:/root/ArmsAgent/arms-bootstrap-1.7.0-SN
APSHOT.jar -Darms.licenseKey='${arms_licenseKey}' -Darms.appName='${arms_appName}
### for check the args
RUN env | grep JAVA_TOOL_OPTIONS
### Add custom Dockerfile logic.
### ......
```

Replace the example values in the preceding configuration file based on the following instructions.

- Replace `{original-docker-image:tag}` with your own image address. If you do not have a custom image, use a system image instead.
- Replace the link for downloading the ARMS agent based on your region.

> **Note** Use the public endpoint. If the download fails, use the VPC endpoint.

Region	Download link for the Internet	Download link for VPC
China (Hangzhou)	`wget "http://arms-apm-hangzhou.oss-cn-hangzhou.aliyuncs.com/ArmsAgent.zip" -O ArmsAgent.zip`	`wget "http://arms-apm-hangzhou.oss-cn-hangzhou-internal.aliyuncs.com/ArmsAgent.zip" -O ArmsAgent.zip`

Region	Download link for the Internet	Download link for VPC
China (Shanghai)	wget "http://arms-apm-shanghai.oss-cn-shanghai.aliyuncs.com/ArmsAgent.zip" -O ArmsAgent.zip	wget "http://arms-apm-shanghai.oss-cn-shanghai-internal.aliyuncs.com/ArmsAgent.zip" -O ArmsAgent.zip
China (Qingdao)	wget "http://arms-apm-qingdao.oss-cn-qingdao.aliyuncs.com/ArmsAgent.zip" -O ArmsAgent.zip	wget "http://arms-apm-qingdao.oss-cn-qingdao-internal.aliyuncs.com/ArmsAgent.zip" -O ArmsAgent.zip
China (Beijing)	wget "http://arms-apm-beijing.oss-cn-beijing.aliyuncs.com/ArmsAgent.zip" -O ArmsAgent.zip	wget "http://arms-apm-beijing.oss-cn-beijing-internal.aliyuncs.com/ArmsAgent.zip" -O ArmsAgent.zip
China (Zhangjiakou)	wget "http://arms-apm-zhangjiakou.oss-cn-zhangjiakou.aliyuncs.com/ArmsAgent.zip" -O ArmsAgent.zip	wget "http://arms-apm-zhangjiakou.oss-cn-zhangjiakou-internal.aliyuncs.com/ArmsAgent.zip" -O ArmsAgent.zip
China (Shenzhen)	wget "http://arms-apm-shenzhen.oss-cn-shenzhen.aliyuncs.com/ArmsAgent.zip" -O ArmsAgent.zip	wget "http://arms-apm-shenzhen.oss-cn-shenzhen-internal.aliyuncs.com/ArmsAgent.zip" -O ArmsAgent.zip
China (Hong Kong)	wget "http://arms-apm-hongkong.oss-cn-hongkong.aliyuncs.com/ArmsAgent.zip" -O ArmsAgent.zip	wget "http://arms-apm-hongkong.oss-cn-hongkong-internal.aliyuncs.com/ArmsAgent.zip" -O ArmsAgent.zip
Singapore (Singapore)	wget "http://arms-apm-ap-southeast.oss-ap-southeast-1.aliyuncs.com/cloud_ap-southeast-1/ArmsAgent.zip" -O ArmsAgent.zip	wget "http://arms-apm-ap-southeast.oss-ap-southeast-1-internal.aliyuncs.com/cloud_ap-southeast-1/ArmsAgent.zip" -O ArmsAgent.zip

Region	Download link for the Internet	Download link for VPC
Japan (Tokyo)	wget "http://arms-apm-japan.oss-ap-northeast-1.aliyuncs.com/ArmsAgent.zip" -O ArmsAgent.zip	wget "http://arms-apm-japan.oss-ap-northeast-1-internal.aliyuncs.com/ArmsAgent.zip" -O ArmsAgent.zip
US (Silicon Valley)	wget "http://arms-apm-usw.oss-us-west-1.aliyuncs.com/ArmsAgent.zip" -O ArmsAgent.zip	wget "http://arms-apm-usw.oss-us-west-1-internal.aliyuncs.com/ArmsAgent.zip" -O ArmsAgent.zip
China East 1 Finance	wget "http://arms-apm-hangzhou.oss-cn-hangzhou.aliyuncs.com/finance/ArmsAgent.zip" -O ArmsAgent.zip	wget "http://arms-apm-hangzhou.oss-cn-hangzhou-internal.aliyuncs.com/finance/ArmsAgent.zip" -O ArmsAgent.zip
China East 2 Finance	wget "http://arms-apm-sh-finance-1.oss-cn-shanghai-finance-1.aliyuncs.com/ArmsAgent.zip" -O ArmsAgent.zip	wget "http://arms-apm-sh-finance-1.oss-cn-shanghai-finance-1-internal.aliyuncs.com/ArmsAgent.zip" -O ArmsAgent.zip
China South 1 Finance	wget "http://arms-apm-sz-finance.oss-cn-shenzhen-finance-1.aliyuncs.com/ArmsAgent.zip" -O ArmsAgent.zip	wget "https://arms-apm-sz-finance.oss-cn-shenzhen-finance-1-internal.aliyuncs.com/ArmsAgent.zip" -O ArmsAgent.zip

- Replace `{LicenseKey}` with your license key. Replace `{AppName}` with the name of your application. The application name cannot contain Chinese characters.

Step 3: Build and start a new image

1. Run the `docker build` command to build an image.

```
docker build -t registry.cn-hangzhou.aliyuncs.com/arms-docker-repo/arms-springboot-demo:v0.1 -f /{workspace}/Dockerfile /{workspace}/
```

> Note Replace `registry.cn-hangzhou.aliyuncs.com/arms-docker-repo/arms-springboot-demo:v0.1` with the actual image name.

2. Run the `docker run` command to start the image. To connect the image to another application

monitoring job, run the docker run command and use the -e parameter to specify the arms_licenseKey and *arms_appName* parameters. This overwrites the configurations in the *Dockerfile* file.

```
docker run -d -e "arms_licenseKey={LicenseKey}" -e "arms_appName={AppName}" -p 8081:8080 registry.cn-hangzhou.aliyuncs.com/arms-docker-repo/arms-springboot-demo:v0.1
```

> **Note** Replace `{LicenseKey}` with your license key. Replace `{AppName}` with the name of your application. The application name cannot contain Chinese characters. Replace `registry.cn-hangzhou.aliyuncs.com/arms-docker-repo/arms-springboot-demo:v0.1` with the actual image name.

Verify the result

After about 1 minute, if your application is displayed in the application list and some data records are sent, it indicates that your application is monitored by ARMS.

Uninstall the ARMS agent

If you no longer need to monitor the Java application in the Docker cluster, perform the following steps to uninstall the ARMS agent.

1. Delete the configurations that you added to the *Dockerfile* file in Step 2: Integrate an existing image.
2. Run the `docker build` command to build an image.
3. Run the `docker run` command to start the image.

Related information

- FAQ

2.3. Monitor PHP applications

2.3.1. Install the ARMS agent for a PHP application

After you install the Application Real-Time Monitoring Service (ARMS) agent for a PHP application, ARMS starts to monitor the PHP application. You can view the monitoring data of application topology, API requests, abnormal transactions, slow transactions, and SQL analysis. The performance of the latest ARMS agent is optimized. The CPU and memory usage of the agent is reduced to around 5%.

> **Note** If you need to try out the new version of PHP Agent, activate it now ARMS Trial Edition. For more information about the end of the trial period of the new version of the PHP Agent, see the ARMS console announcement. If you have other questions, you can join the DingTalk Q&A group: 23328286.

Install the ARMS agent

1. Run the **wget** command to download the installation package. Download the installation package based on your region.

Region	Download link for the Internet	Download link for VPC
China (Hangzhou)	wget "http://arms-apm-hangzhou.oss-cn-hangzhou.aliyuncs.com/arms-php-agent.zip" -O arms-php-agent.zip	wget "http://arms-apm-hangzhou.oss-cn-hangzhou-internal.aliyuncs.com/arms-php-agent.zip" -O arms-php-agent.zip
China (Shanghai)	wget "http://arms-apm-shanghai.oss-cn-shanghai.aliyuncs.com/arms-php-agent.zip" -O arms-php-agent.zip	wget "http://arms-apm-shanghai.oss-cn-shanghai-internal.aliyuncs.com/arms-php-agent.zip" -O arms-php-agent.zip
China (Qingdao)	wget "http://arms-apm-qingdao.oss-cn-qingdao.aliyuncs.com/arms-php-agent.zip" -O arms-php-agent.zip	wget "http://arms-apm-qingdao.oss-cn-qingdao-internal.aliyuncs.com/arms-php-agent.zip" -O arms-php-agent.zip
China (Beijing)	wget "http://arms-apm-beijing.oss-cn-beijing.aliyuncs.com/arms-php-agent.zip" -O arms-php-agent.zip	wget "http://arms-apm-beijing.oss-cn-beijing-internal.aliyuncs.com/arms-php-agent.zip" -O arms-php-agent.zip
China (Zhangjiakou)	wget "http://arms-apm-zhangjiakou.oss-cn-zhangjiakou.aliyuncs.com/arms-php-agent.zip" -O arms-php-agent.zip	wget "http://arms-apm-zhangjiakou.oss-cn-zhangjiakou-internal.aliyuncs.com/arms-php-agent.zip" -O arms-php-agent.zip
China (Shenzhen)	wget "http://arms-apm-shenzhen.oss-cn-shenzhen.aliyuncs.com/arms-php-agent.zip" -O arms-php-agent.zip	wget "http://arms-apm-shenzhen.oss-cn-shenzhen-internal.aliyuncs.com/arms-php-agent.zip" -O arms-php-agent.zip

Region	Download link for the Internet	Download link for VPC
China (Hong Kong)	wget "http://arms-apm-hongkong.oss-cn-hongkong.aliyuncs.com/arms-php-agent.zip" -O arms-php-agent.zip	wget "http://arms-apm-hongkong.oss-cn-hongkong-internal.aliyuncs.com/arms-php-agent.zip" -O arms-php-agent.zip

2. Run the following command to decompress the installation package and move it to the */usr/local/arms/arms-php-agent* directory:

```
unzip arms-php-agent.zip
mkdir -p /usr/local/arms
mv arms-php-agent /usr/local/arms/arms-php-agent
```

3. Log on to the ARMS console .
4. In the left-side navigation pane, choose **Application Monitoring > Applications**. In the top navigation bar, select a region.
5. On the **Applications** page, click **Add Application** in the upper-right corner.
6. Copy the license key at the top of the **Add Application** page.
7. Add the following code to the *php.ini* configuration file.

```
extension=/usr/local/arms/arms-php-agent/arms-x.x.so
[ARMS]
arms.enable=1
arms.app_name=<yourAppName>
arms.license_key=<yourLicenseKey>
arms.sock_path=/arms.sock
```

> **Note**
> - `x.x` in `arms-x.x.so` is the version of your PHP application. Versions 5.4 to 7.3 are supported.
> - Set `<yourAppName>` to a custom name. The name is displayed as your PHP application name in the ARMS console.
> - Replace `<yourLicenseKey>` with the license key that you obtained in Step 6.
> - If `with-config-file-scan-dir` is configured for the version of your PHP application, you can create the *arms.ini* file in the */etc/php/7.2/php-fpm/conf.d* directory. The content of this file is the same as that you added to the *php.ini* file.

8. Run the following command to start the Hercules service to transfer the data of your PHP application:

```
cd /usr/local/arms/arms-php-agent/
sudo ./hercules service install
sudo ./hercules service start
```

9. Restart the service.

- If you are using an NGINX server, restart the PHP-FPM service.
- If you are using an Apache server, restart the Apache2 service.

  Wait for about 1 minute. If ARMS console Of **Application Monitoring > Applications**If your application is displayed with the name of custom <yourAppName>, the probe is installed.

Uninstall the ARMS agent

1. Delete the content of the *php.ini* or *arms.ini* file that you added in Step 7.
2. Restart the service.
   - If you are using an NGINX server, restart the PHP-FPM service.
   - If you are using an Apache server, restart the Apache2 service.
3. Run the following commands to stop and uninstall the Hercules service:

   ```
   sudo ./hercules service stop
   sudo ./hercules service uninstall
   ```

4. Run the following command to delete the directory of the ARMS agent:

   ```
   sudo rm -rf /usr/local/arms/arms-php-agent
   ```

   You have uninstalled the ARMS agent for the PHP application.

Related information

- Install the ARMS agent for PHP applications deployed on multiple servers in standalone mode

2.3.2. Install the ARMS agent for PHP applications deployed on multiple servers in standalone mode

After you install the Application Real-Time Monitoring Service (ARMS) agent for PHP applications, ARMS starts to monitor the PHP applications. You can view the monitoring data of application topology, API requests, abnormal transactions, slow transactions, and SQL analysis. This topic describes how to install the ARMS agent for PHP applications that are deployed on multiple servers in standalone mode.

> **Note** If you need to try out the new version of PHP Agent, activate it now ARMS Trial Edition . For more information about the end of the trial period of the new version of the PHP Agent, see the ARMS console announcement. If you have other questions, you can join the DingTalk Q&A group: 23328286.

Install the ARMS agent

1. Install the ARMS agent. For more information, see Install the ARMS agent for a PHP application.
2. Edit the configuration file of Apache or NGINX.
   - For PHP applications that are deployed on multiple Apache servers in standalone mode, add `php_value arms.app_name "<yourAppNewName>"` to each VirtualHost. Replace `<yourAppNewName>` with the name of your PHP application. Example:

```
<VirtualHost *:80>
  ServerName www.example.com
  DocumentRoot /home/www/html
  php_value arms.app_name "example"
  <Directory "/home/www/html">
     Options FollowSymLinks
     AllowOverride All
     Require all granted
  </Directory>
</VirtualHost>
<VirtualHost *:80>
  ServerName www.test.com
  DocumentRoot /home/www/test
  php_value arms.app_name "test"
  <Directory "/home/www/test">
     Options FollowSymLinks
     AllowOverride All
     Require all denied
     Require all granted
  </Directory>
</VirtualHost>
```

- For PHP applications that are deployed on multiple NGINX servers in standalone mode, add `fastcgi_param PHP_VALUE "arms.app_name=<yourAppNewName>"` to the PHP-FPM configuration file of each server. Replace `<yourAppNewName>` with the name of your PHP application. Example:

```
server {
    listen       80;
    server_name  localhost;
    location / {
            try_files $uri $uri/ /index.php? $query_string;
      }
    error_page   500 502 503 504  /50x.html;
    location = /50x.html {
      root   /usr/share/nginx/html;
    }
    location ~ \.php$ {
      fastcgi_pass   localhost:9000;
      fastcgi_index  index.php;
      fastcgi_param  PHP_VALUE "arms.app_name=example"
      fastcgi_param  SCRIPT_FILENAME  /var/www/html/$fastcgi_script_name;
      include        fastcgi_params;
    }
  }
 server {
    listen       80;
    server_name  www.example.com;
    location / {
            try_files $uri $uri/ /index.php? $query_string;
      }
    error_page   500 502 503 504  /50x.html;
    location = /50x.html {
      root   /usr/share/nginx/html;
    }
    location ~ \.php$ {
      fastcgi_pass   localhost:9000;
      fastcgi_index  index.php;
      fastcgi_param  PHP_VALUE "arms.app_name=test"
      fastcgi_param  SCRIPT_FILENAME  /var/www/test/$fastcgi_script_name;
      include        fastcgi_params;
    }
  }
```

Wait for about 1 minute. If ARMS console Of **Application Monitoring > Applications**If your application is displayed with the name of custom <yourAppName>, the probe is installed.

Uninstall the ARMS agent

1. Delete the content of the *php.ini* or *arms.ini* file. For more information, see Step 7 in Install the ARMS agent for a PHP application.
2. Delete the content that you added to the Apache or NGINX configuration file in Step 2 of this topic.
3. Restart the service.
   - If you are using NGINX servers, restart the PHP-FPM service.
   - If you are using Apache servers, restart the Apache2 service.
4. Run the following commands to stop and uninstall the Hercules service:

```
sudo ./hercules service stop
sudo ./hercules service uninstall
```

5. Run the following command to delete the directory of the ARMS agent:

```
sudo rm -rf /usr/local/arms/arms-php-agent
```

You have uninstalled the ARMS agent for PHP applications.

Related information

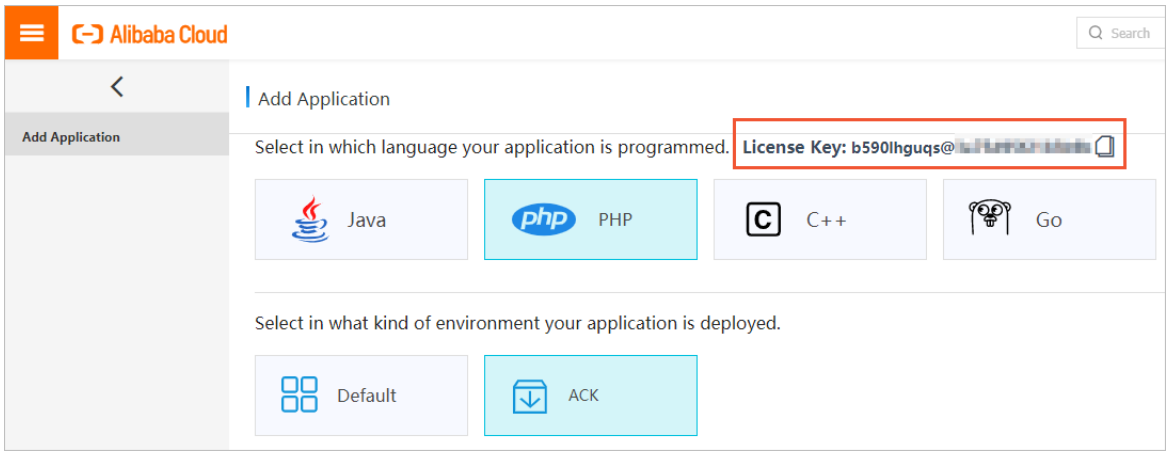
- Install the ARMS agent for a PHP application

2.3.3. Install the ARMS agent for a PHP application deployed in Container Service for Kubernetes

After you install the Application Real-Time Monitoring Service (ARMS) agent for a PHP application that is deployed in Container Service for Kubernetes, ARMS starts to monitor the PHP application. You can view the monitoring data of application topology, API requests, abnormal transactions, slow transactions, and SQL analysis. This topic describes how to install the ARMS agent for a PHP application that is deployed in Container Service for Kubernetes.

Obtain the license key

1. Log on to the ARMS console .
2. In the left-side navigation pane, choose **Application Monitoring > Applications**. In the top navigation bar, select a region.
3. On the **Applications** page, click **Add Application** in the upper-right corner.
4. Copy the license key at the top of the **Add Application** page.



Install the ARMS application monitoring agent

Install the ARMS application monitoring components ack-arms-pilot.

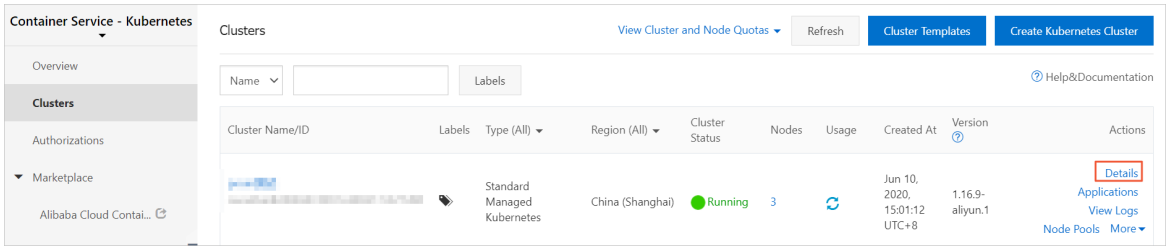
1. Log on to the Alibaba Cloud Container Service for Kubernetes console .
2. In the left-side navigation pane, click **Latest version > App Catalog** . On the right page, click **ack-arms-pilot** .

3. Log on to the **App Catalog-ack-arms-pilot** On the page, on the right **Creation** Panel, select the cluster created in prerequisites, and click **Creation** .

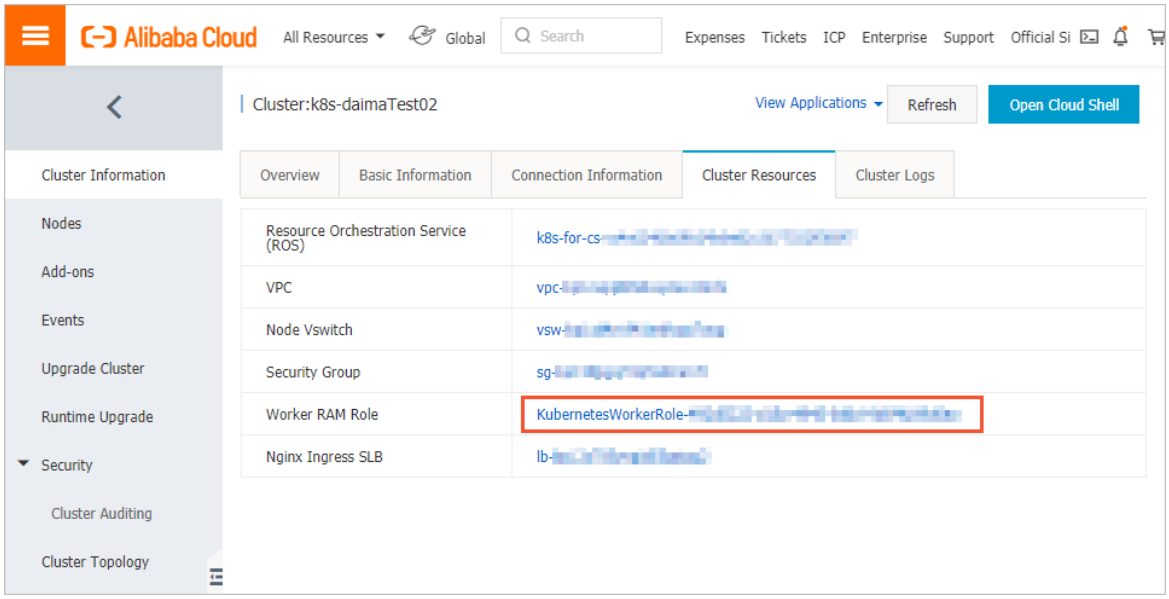
Authorize ACK to access ARMS

Grant Alibaba Cloud Container Service for Kubernetes access permissions on ARMS resources.

1. Log on to the Log on to the Container Service console. .
2. In the left-side navigation pane, click **cluster** , in **Clusters** Page on the right of the target cluster **Action** Column click **View Details** .

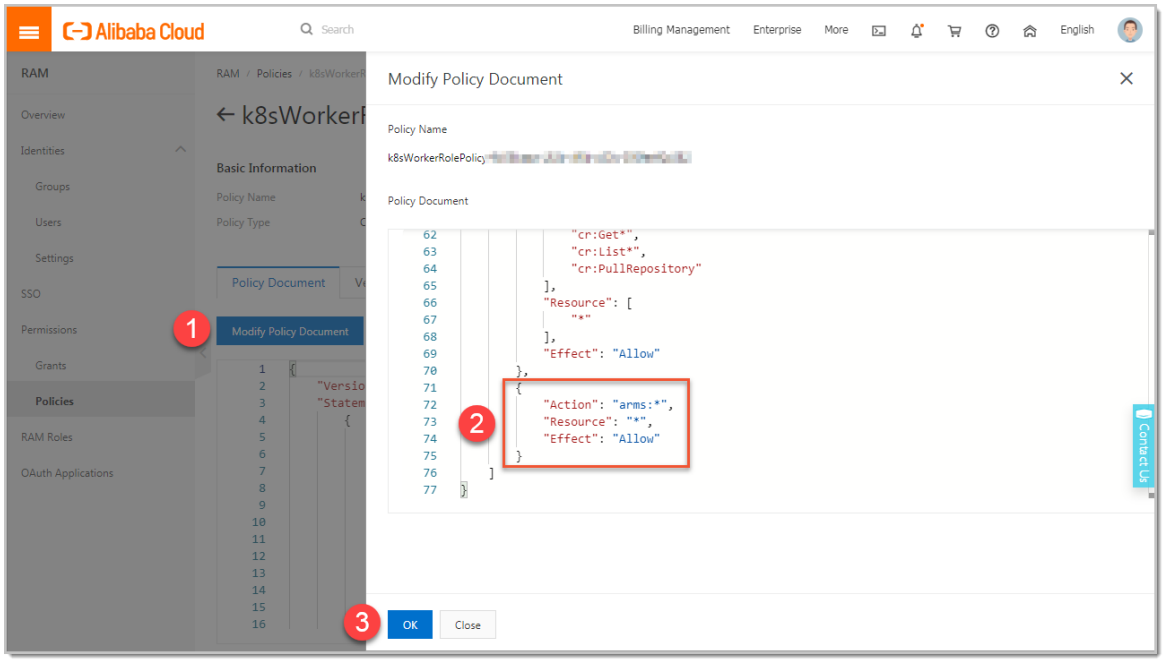


3. In the destination cluster, **Create CDH Cluster Configuration** On the page, click **Cluster Resources** Tab, and then click **Worker RAM Role** Link on the right.



4. In the Resource Access Management RAM console, **RAM Roles** Page, click **Permission management** The name of the permission policy on the tab.
5. Log on to the **Policy Document** On the tab, click **Modify Policy Document** , and add the following to the **Policy Document** In, finally click **OK** .

```
{"Action": "arms:*", "Resource": "*", "Effect": "Allow"}
```



Install the Hercules Deploy component to transfer the data of the PHP application

1. Create a local YAML file, name it *hercules.yaml*, and then copy the following content to the YAML file:
   You must replace the value of the image parameter with one of the following addresses based on the region.

Region	Download link
China (Hangzhou)	registry.cn-hangzhou.aliyuncs.com/arms-docker-repo/arms-hercules:v1.1
China (Shanghai)	registry.cn-shanghai.aliyuncs.com/arms-docker-repo/arms-hercules:v1.1
China (Qingdao)	registry.cn-qingdao.aliyuncs.com/arms-docker-repo/arms-hercules:v1.1
China (Beijing)	registry.cn-beijing.aliyuncs.com/arms-docker-repo/arms-hercules:v1.1
China (Zhangjiakou)	registry.cn-zhangjiakou.aliyuncs.com/arms-docker-repo/arms-hercules:v1.1
China (Shenzhen)	registry.cn-shenzhen.aliyuncs.com/arms-docker-repo/arms-hercules:v1.1

Region	Download link
China (Hong Kong)	registry.cn-hongkong.aliyuncs.com/arms-docker-repo/arms-hercules:v1.1
Singapore	registry.ap-southeast-1.aliyuncs.com/arms-docker-repo/arms-hercules:v1.1

2. Run the following command to install the Hercules Deploy component:

```
kubectl create -f hercules.yaml
```

> **Note** If a message indicating that the namespace already exists is displayed in the command output, ignore the message.

Enable ARMS to monitor a PHP application

1. Log on to the Container Service for Kubernetes console. In the left-side navigation pane, click **Clusters**. On the **Clusters** page, click **Applications** in the **Actions** column of the cluster where your application is deployed.
2. On the **Deployments** tab, click **Create from Template** in the upper-right corner.
3. On the page that appears, select a template from the **Sample Template** drop-down list, and add the following `annotations` to the *spec > template > metadata* section.

```
annotations:
 armsPilotAutoEnable: "on"
 armsPilotCreateAppName: "<your-deployment-name>"
 armsAppType: PHP
```

> **Note** Replace *<your-deployment-name>* with the name of your Deployment application.

4. (This step is required only when you install the ARMS agent for the first time.) In the namespace of your application, create a ConfigMap file named *arms-<yourAppName>.ini* and copy the following content to the file:

```
apiVersion: v1
kind: ConfigMap
metadata:
 name: arms-<yourAppName>.ini
 namespace: <yourAppNamespace>
data:
 arms.ini: |
  extension=/usr/local/arms/arms-php-agent/arms-[x.y].so
  [ARMS]
  arms.enable=1
  arms.app_name=<yourAppName>
  arms.license_key=<yourLicenseKey>
  arms.agent_env=PHPK8S
  arms.network_type=tcp
  arms.tcp_host=arms-hercules-service.arms-pilot
  arms.tcp_port=11234
```

Note

- Replace *<yourAppName>* with the name of your application.
- Replace *<yourAppNamespace>* with the namespace of your application.
- Replace *<yourLicenseKey>* with the license key that you obtained on the **Add Application** page in the ARMS console.
- In the `extension=/usr/local/arms/arms-php-agent/arms-[x.y].so` configuration, `[x.y]` in `arms-[x.y].so` is the version of the PHP application. The supported versions are 5.4, 5.5, 5.6, 7.0, 7.1, 7.2, and 7.3.
- If your container image is an Alpine Linux system, change `arms-[x.y].so` to `arms-[x.y]-alpine.so`. `[x.y]` can be 5.5, 5.6, 7.0, 7.1, 7.2, or 7.3.
- A ConfigMap file has a one-to-one mapping with an application. To connect to another application, create another ConfigMap file and delete the file when the application is no longer needed.

5. Add ConfigMap of *arms-<yourAppName>.ini* to the spec > template > spec > containers section of the Deployment application. Set mountPath to the path of the PHP configuration file.

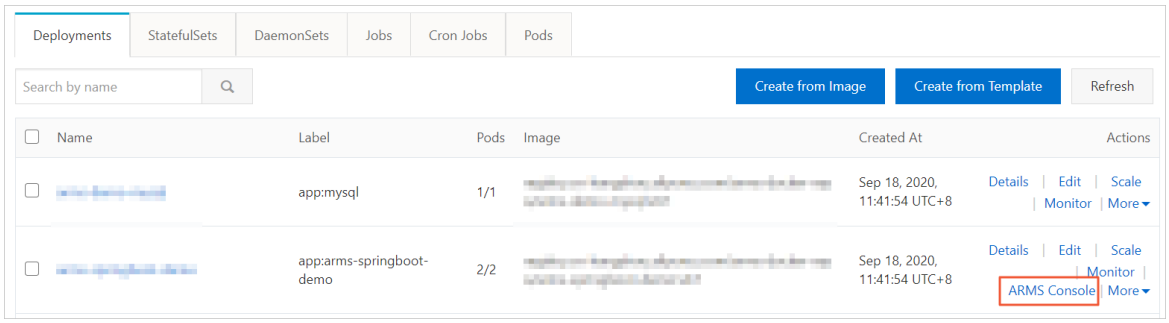
```
volumeMounts:
- mountPath: /etc/php/7.2/fpm/conf.d/arms.ini
 name: arms-ini
 subPath: arms.ini
```

In the preceding information, `/etc/php/7.2/fpm/conf.d/` is the configuration loading directory of the PHP application. Add the following content to the *spec > template > spec > volumes* section:

```
volumes:
- name: arms-ini
 configMap:
  name: arms-<yourAppName>.ini
```

> **Note** If the PHP application does not have a configuration file path, add the content of arms-<yourAppName>.ini to the php.ini configuration file of the application.

Log on to the Container Service for Kubernetes console. On the **Deployments** or **StatefulSets** tab, if **ARMS Console** appears in the **Actions** column of the application, the ARMS agent is installed.



> **Note** If you cannot find **ARMS Console** in the **Actions** column, check whether you have authorized Container Service to access ARMS.

Uninstall the ARMS agent

1. (Optional)If you need to pause the ARMS agent, delete the ConfigMap file that you added in Step 5 and deploy the application again.
2. If you need to uninstall the ARMS agent, delete the Hercules Deploy component that you added in Step 2 and the ConfigMap file that you added in Step 5.

More information

If you have other questions, you can join the DingTalk Q&A group: 23328286.

2.3.4. Install the ARMS agent for a PHP application deployed in a Docker cluster

After you install the Application Real-Time Monitoring Service (ARMS) agent for a PHP application that is deployed in a Docker cluster, ARMS starts to monitor the PHP application. You can view the monitoring data of application topology, API requests, abnormal transactions, slow transactions, and SQL analysis. This topic describes how to install the ARMS agent for a PHP application that is deployed in a Docker cluster.

> **Note** If you need to try out the new version of PHP Agent, activate it now ARMS Trial Edition. For more information about the end of the trial period of the new version of the PHP Agent, see the ARMS console announcement. If you have other questions, you can join the DingTalk Q&A group: 23328286.

Procedure

1. Run a Docker container on the host where the ARMS agent needs to be installed.

```
sudo docker run -d -p 11234:11234 <IMAGE>
```

Replace `<IMAGE>` with one of the following download links based on your region.

Region	Download link
China (Hangzhou)	registry.cn-hangzhou.aliyuncs.com/arms-docker-repo/arms-hercules:v1.1
China (Shanghai)	registry.cn-shanghai.aliyuncs.com/arms-docker-repo/arms-hercules:v1.1
China (Qingdao)	registry.cn-qingdao.aliyuncs.com/arms-docker-repo/arms-hercules:v1.1
China (Beijing)	registry.cn-beijing.aliyuncs.com/arms-docker-repo/arms-hercules:v1.1
China (Zhangjiakou)	registry.cn-zhangjiakou.aliyuncs.com/arms-docker-repo/arms-hercules:v1.1
China (Shenzhen)	registry.cn-shenzhen.aliyuncs.com/arms-docker-repo/arms-hercules:v1.1
China (Hong Kong)	registry.cn-hongkong.aliyuncs.com/arms-docker-repo/arms-hercules:v1.1
Singapore	registry.ap-southeast-1.aliyuncs.com/arms-docker-repo/arms-hercules:v1.1

Note In this case, port 11234 of the host exposes the Hercules service. If a port conflict occurs, map port 11234 to another port of the host.

2. Run the `wget` command to download the installation package. Select the download link based on your network environment.

Region	Download link for the Internet	Download link for VPC
China (Hangzhou)	wget "http://arms-apm-hangzhou.oss-cn-hangzhou.aliyuncs.com/arms-php-agent.zip" -O arms-php-agent.zip	wget "http://arms-apm-hangzhou.oss-cn-hangzhou-internal.aliyuncs.com/arms-php-agent.zip" -O arms-php-agent.zip
China (Shanghai)	wget "http://arms-apm-shanghai.oss-cn-shanghai.aliyuncs.com/arms-php-agent.zip" -O arms-php-agent.zip	wget "http://arms-apm-shanghai.oss-cn-shanghai-internal.aliyuncs.com/arms-php-agent.zip" -O arms-php-agent.zip

Region	Download link for the Internet	Download link for VPC
China (Qingdao)	wget "http://arms-apm-qingdao.oss-cn-qingdao.aliyuncs.com/arms-php-agent.zip" -O arms-php-agent.zip	wget "http://arms-apm-qingdao.oss-cn-qingdao-internal.aliyuncs.com/arms-php-agent.zip" -O arms-php-agent.zip
China (Beijing)	wget "http://arms-apm-beijing.oss-cn-beijing.aliyuncs.com/arms-php-agent.zip" -O arms-php-agent.zip	wget "http://arms-apm-beijing.oss-cn-beijing-internal.aliyuncs.com/arms-php-agent.zip" -O arms-php-agent.zip
China (Zhangjiakou)	wget "http://arms-apm-zhangjiakou.oss-cn-zhangjiakou.aliyuncs.com/arms-php-agent.zip" -O arms-php-agent.zip	wget "http://arms-apm-zhangjiakou.oss-cn-zhangjiakou-internal.aliyuncs.com/arms-php-agent.zip" -O arms-php-agent.zip
China (Shenzhen)	wget "http://arms-apm-shenzhen.oss-cn-shenzhen.aliyuncs.com/arms-php-agent.zip" -O arms-php-agent.zip	wget "http://arms-apm-shenzhen.oss-cn-shenzhen-internal.aliyuncs.com/arms-php-agent.zip" -O arms-php-agent.zip
China (Hong Kong)	wget "http://arms-apm-hongkong.oss-cn-hongkong.aliyuncs.com/arms-php-agent.zip" -O arms-php-agent.zip	wget "http://arms-apm-hongkong.oss-cn-hongkong-internal.aliyuncs.com/arms-php-agent.zip" -O arms-php-agent.zip
Singapore	wget "http://arms-apm-ap-southeast.oss-ap-southeast-1.aliyuncs.com/cloud_ap-southeast-1/arms-php-agent.zip" -O arms-php-agent.zip	wget "http://arms-apm-ap-southeast.oss-ap-southeast-1-internal.aliyuncs.com/cloud_ap-southeast-1/arms-php-agent.zip" -O arms-php-agent.zip
Alibaba Gov Cloud	wget "http://arms-apm-gov.oss-cn-north-2-gov-1.aliyuncs.com/arms-php-agent.zip" -O arms-php-agent.zip	wget "http://arms-apm-gov.oss-cn-north-2-gov-1-internal.aliyuncs.com/arms-php-agent.zip" -O arms-php-agent.zip

3. Decompress the installation package and move it to the */usr/local/arms/arms-php-agent* directory.

```
unzip arms-php-agent.zip
mkdir -p /usr/local/arms
mv arms-php-agent /usr/local/arms/arms-php-agent
```

4. Log on to the ARMS console .
5. In the left-side navigation pane, choose **Application Monitoring > Applications**. In the top navigation bar, select a region.
6. On the **Applications** page, click **Add Application** in the upper-right corner.
7. Copy the license key in the upper part of the **Add Application** page.
8. Add the following code to the *php.ini* configuration file.

```
extension=/usr/local/arms/arms-php-agent/arms-x.x.so
[ARMS]
arms.enable=1
arms.app_name=<yourAppName>
arms.license_key=<yourLicenseKey>
arms.network_type=tcp
arms.tcp_host=<host>
arms.tcp_port=<port>
```

> Note
>
> - `x.x` in `arms-x.x.so` is the version of your PHP application. Versions 5.4 to 7.3 are supported.
> - If your container image system is Alpine Linux, change `arms-x.x.so` to `arms-[x.y]-alpine.so`, where `[x.y]` can be 5.5, 5.6, 7.0, 7.1, 7.2, or 7.3.
> - Set *<yourAppName>* to a custom name. The name is displayed as your PHP application name in the ARMS console.
> - Replace *<yourLicenseKey>* with the license key that you obtained in Step 7.
> - *<host>* and *<port>* indicate the IP address and port number that are used by the container to access the host. The default port number is 11234. If you update the port number in Step 1, you must also update it in this step.
> - If `with-config-file-scan-dir` is configured for the version of your PHP application, you can create the *arms.ini* file in the */etc/php/7.2/php-fpm/conf.d* directory. The content of this file is the same as that you added to the *php.ini* file.

9. (Optional)Add the commands in Step 2 to Step 8 to the *dockerfile* file for auto running.
10. Restart your PHP application.

Related information

-
- Install the ARMS agent for a PHP application

2.4. Monitor other applications

Application Real-Time Monitoring Service (ARMS) can monitor Java and PHP applications. Tracing Analysis can be used to monitor applications written in other programming languages such as C++, Go, Node.js, and NET. After you prepare for application monitoring, Tracing Analysis can provide features such as query and diagnostics of distributed traces, real-time integration of performance data, and dynamic discovery of distributed topologies. These features can help you monitor applications.

Background information

Tracing Analysis provides a set of tools for distributed application development. These tools include trace mapping, call request statistics, trace topology, and application dependency analysis. You can use these tools to analyze and diagnose performance bottlenecks in a distributed application architecture and make microservice development and diagnostics more efficient. Tracing Analysis provides the following features:

- Query and diagnostics of distributed traces: This feature tracks microservice user requests in the distributed architecture and summarizes these requests into distributed traces.
- Real-time collection of application performance data: This feature tracks all user requests for an application and collects and analyzes in real time the performance data of the services and resources that constitute the application.
- Dynamic discovery of distributed topologies: This feature collects information about distributed calls to your distributed microservice applications and Platform as a Service (PaaS) products.
- Multi-language development program: Tracing Analysis is fully compatible with open source communities such as Jaeger and Zipkin based on the OpenTracing standard.
- Integration with various downstream analysis platforms: This feature uses collected traces for log analysis and allows Tracing Analysis to connect to downstream analysis platforms such as MaxCompute.

Monitor multi-language applications

Check documents based on the language of your application.

- Integrate your application with Tracing Analysis - Go
- Use Zipkin to report Go application data
- Integrate your application with Tracing Analysis - Python
- Instrument Node.js applications
- Use Jaeger to report .NET application data
- Use Zipkin to report .NET application data
- Integrate your application with Tracing Analysis - C++

What to do next

After you complete the preparations, you can use the following features of Tracing Analysis:

- View the key metrics of an application, such as the health score, number of requests today, and number of errors today. For more information, see View application list.
- View the key performance metrics and topology of an application. For more information, see View application performance metrics and topology.
- View the key performance metrics, call topology, and traces of an application on each host where the application is deployed. For more information, see View application details.
- View the API calls of an application. For more information, see View interface invocation information.
- Query traces by using the multi-dimensional query feature. For more information, see Query invocation traces.

- View the application traces, trace topologies, real-time aggregate trace tables, and waterfall plots of traces. For more information, see Analyze traces.
- Specify whether to display the host name and whether to collect application data, manage custom tags of an application, and delete an application. For more information, see Manage applications and tags.

3.Console functions

3.1. 3D topology

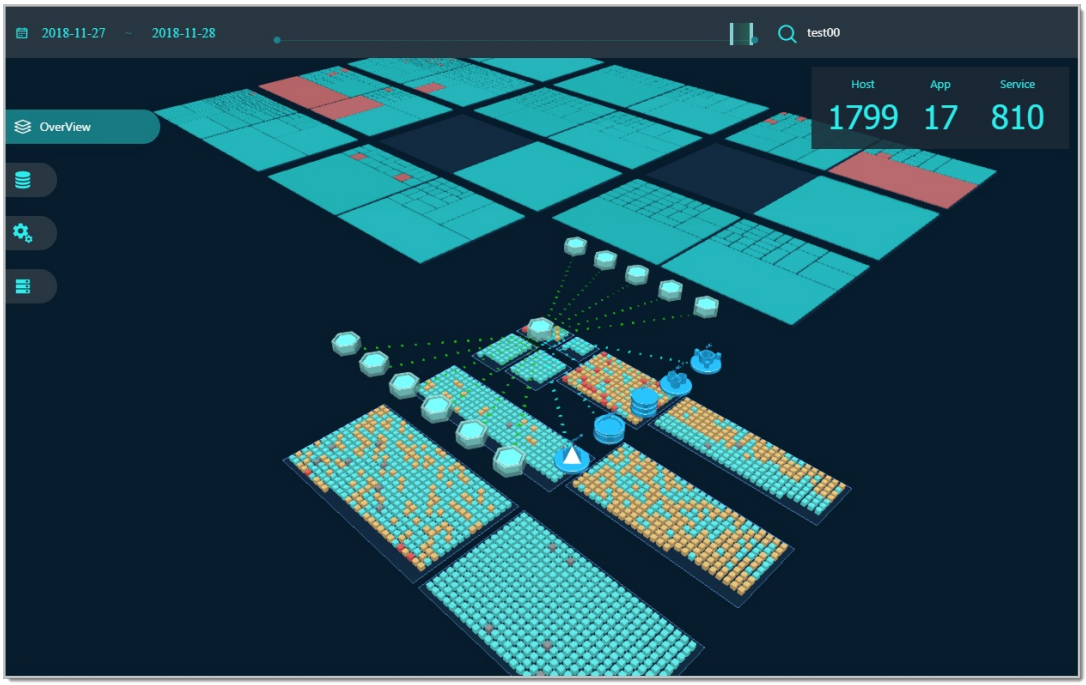
3D topology of Application Real-Time Monitoring Service (ARMS) can show the health condition of applications, services, and hosts, in addition to the upstream and downstream dependencies of the applications. 3D topology helps you identify the services that caused failures, applications affected by the failures, and associated hosts. This way, you can thoroughly diagnose the root cause of failures and troubleshoot these failures.

Procedure

1. Log on to the ARMS console .
2. In the left-side navigation pane, choose **Application Monitoring > Applications**. In the upper part of the Applications page, select the required region.
3. On the **Applications** page, click **3D Topology** of the required application in the **Actions** column.

Overview

On the **Overview** page that is displayed by default, you can view all content of the service layer, application layer, and host layer. In the upper-right corner of the page, you can find the number of hosts, applications, and services.



On the overview layer, you can perform the following operations:

- In the upper-left corner, click the time range section, and select specific start and end time in the pop-up time picker.
- In the timeline on the top of the page, drag the slider to change the time range of the current view.
- In the search box in the upper-right corner of the page, enter your keywords and press the Enter key to search.
- Drag with your pointer to view the data on all three layers from different angles.

- Click any object in the view to check metrics related to that object on the right-side panel.

Service

The service layer shows the services that your applications depend on.

Services under each application are grouped into a block. The more the services are called, the bigger their block is. Different statuses of the services are displayed in different colors.

- : Service calls are normal.
- : The error rate of the service is relatively high.
- : No data is returned from the service.

> **Note** The response time threshold of the service is configurable. In the left-side navigation pane, click the triangle icon next to Service to open the threshold setting box. Drag the slider in the setting box to set the threshold.

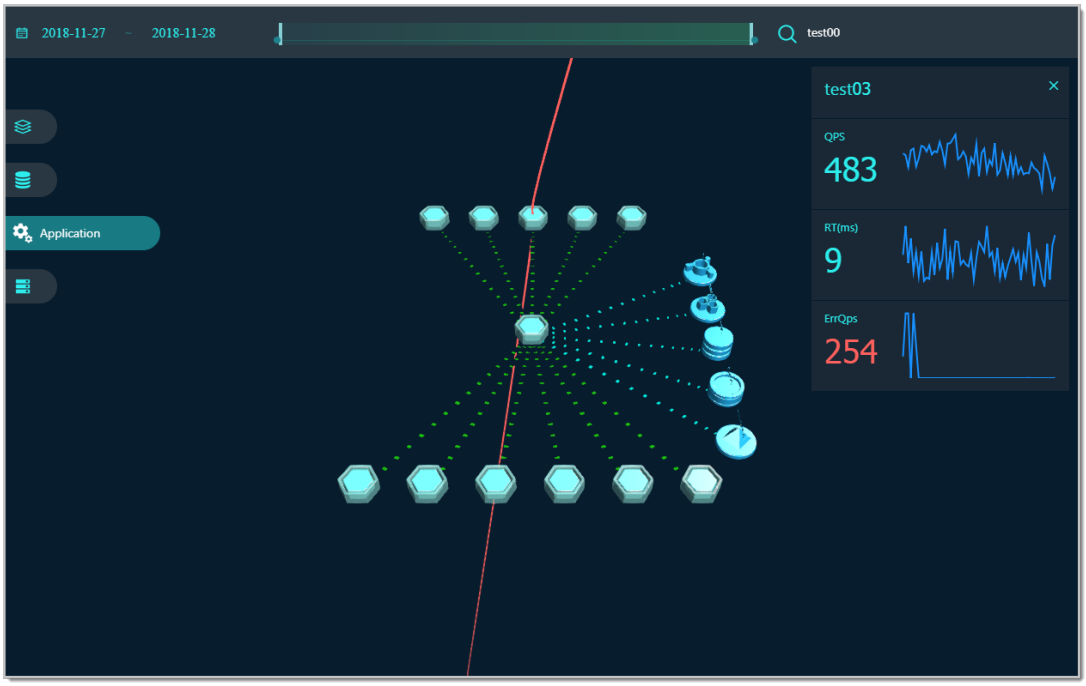
After you click a service, the right-side panel shows the following information:

- The name of the service
- QPS: the queries per second
- RT(ms): the response time in milliseconds
- ErrQps: the error queries per second

> **Note** In the **QPS**, **RT(ms)**, and **Error** sections, the left-side numbers are the average value within the selected time range, and the corresponding line chart is on the right side.

Application

The application layer shows the applications and their upstream and downstream dependencies, including the middleware that the applications and their upstream and downstream depend on. Follow the direction of the connecting lines. You can view the direction in which a call is made.



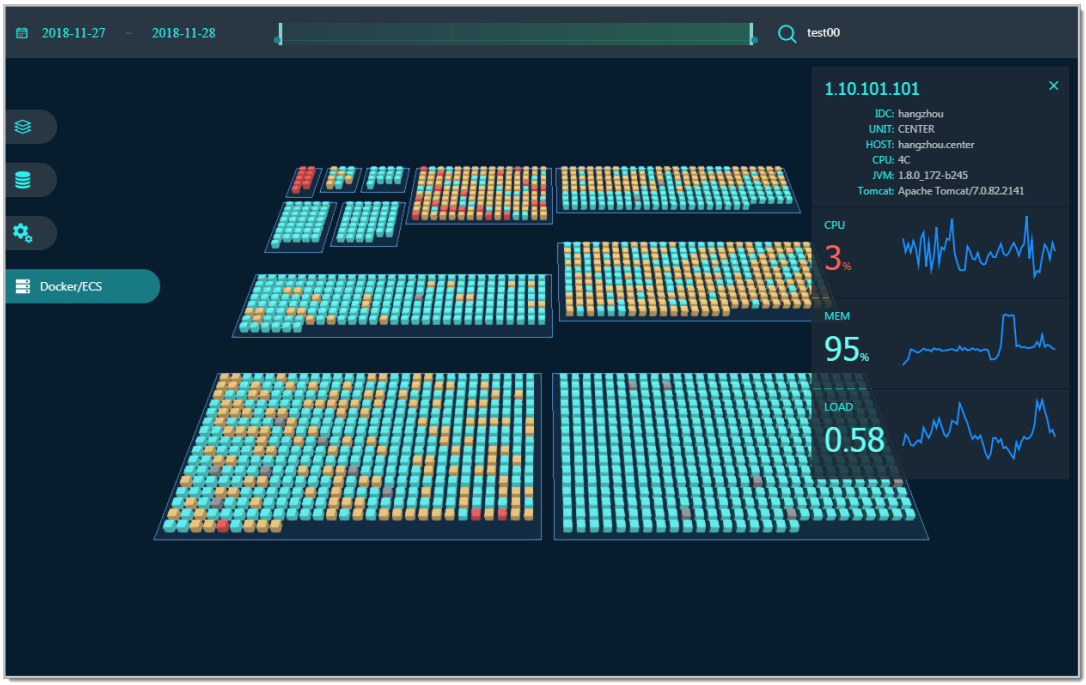
After you click an application, the right-side panel shows the following information of the application:

- Application name
- QPS: the queries per second
- RT(ms): the response time in milliseconds
- ErrQps: the error queries per second

> **Note** In the **QPS**, **RT(ms)**, and **Error** sections, the left-side numbers are the average value within the selected time range, and the corresponding line chart is on the right side.

Docker/ECS

The Docker/ECS layer shows the hosts of your applications.



Each cube indicates a host. All hosts are grouped by application. Different statuses of the hosts are displayed in different colors.

- : Normal
- : Slow
- : Alerting
- : Abnormal
- : Offline

After you click a host, the right-side panel shows the following information of the host:

- IP address and basic information of the host:
  - Response time
  - Number of requests
  - Number of errors
- CPU: CPU utilization
- MEM: memory usage
- DISK: disk usage
- GC TIME: the total GC time
- GC COUNT: the count of GC

> **Note** In the **CPU**, **MEM**, and **Disk** sections, the left-side numbers are the average value within selected time range, while the corresponding line chart is on the right side.

3.2. Trace query

On the **Call link query** page, you can query the details about a specific trace based on the trace ID or query traces by using multiple filter conditions. You can also perform aggregation analysis on multiple traces.

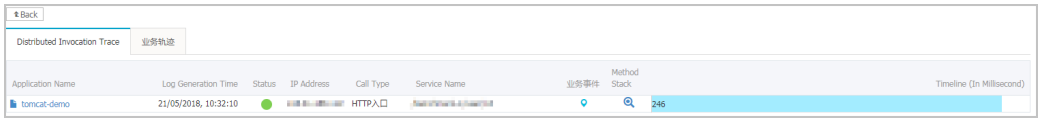
Query traces

1. Log on to the ARMS console .
2. In the left-side navigation pane, choose **Application Monitoring > Invocation Trace Query**. In the top navigation bar, select a region.
3. On the **Call link query** page, select a parameter from the **Parameter type** drop-down list, enter a custom tag in the **Parameter value** field, and then click **Add to query criteria**.
   You can set Parameter type to TraceId for exact match.

Parameter description

Parameter	Description
TraceId	Enter a trace ID.
Interface name	Enter an operation name. Fuzzy match is not supported.
Client application name	Enter the name of an application on the client.
Server application name	Enter the name of an application on the server.
Time-consuming greater	Specify a number of milliseconds to query calls that take longer than the specified number of milliseconds.
Call type	Select a call type.
Abnormal call	Set this parameter to true to search for all traces that contain abnormal calls.
Only thread profiling snapshots are included.	Set this parameter to true to search for all traces that contain thread profiling snapshots.
Client IP	Enter the IP address of an application that initiates the call.
Server IP	Enter the IP address of an application that is called.
Business primary key	Enter a business primary key to search for business events.
Response code	Enter a response code.

4. Click the ID of the trace that you want to view to go to the **Trace Details** page.



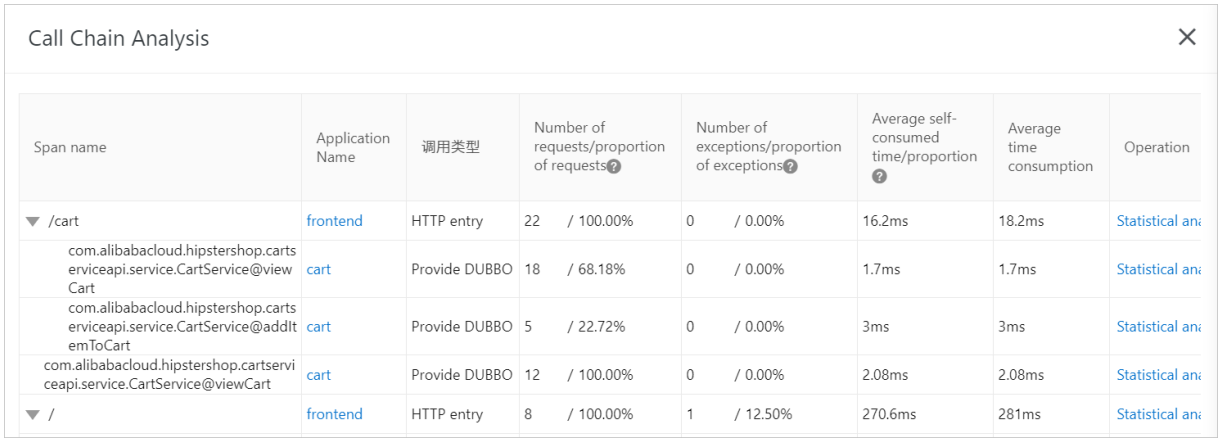
The following section describes the parameters on the **Trace Details** page:

- Application Name: the name of the application to which the trace belongs.

- Log generation time: the time when the log was generated.
- Status: Red indicates that an exception exists in the local trace called by the service. Green indicates that the trace is normal.
- IP Address: the IP address of the application.
- Call Type: the type of the call, which corresponds to the call type of the ad hoc query.
- Service Name: the name of the service operation that is called.
- Timeline: the time consumed by each service to call the trace and the proportion of time consumed by each service in the time consumed to call the entire trace.

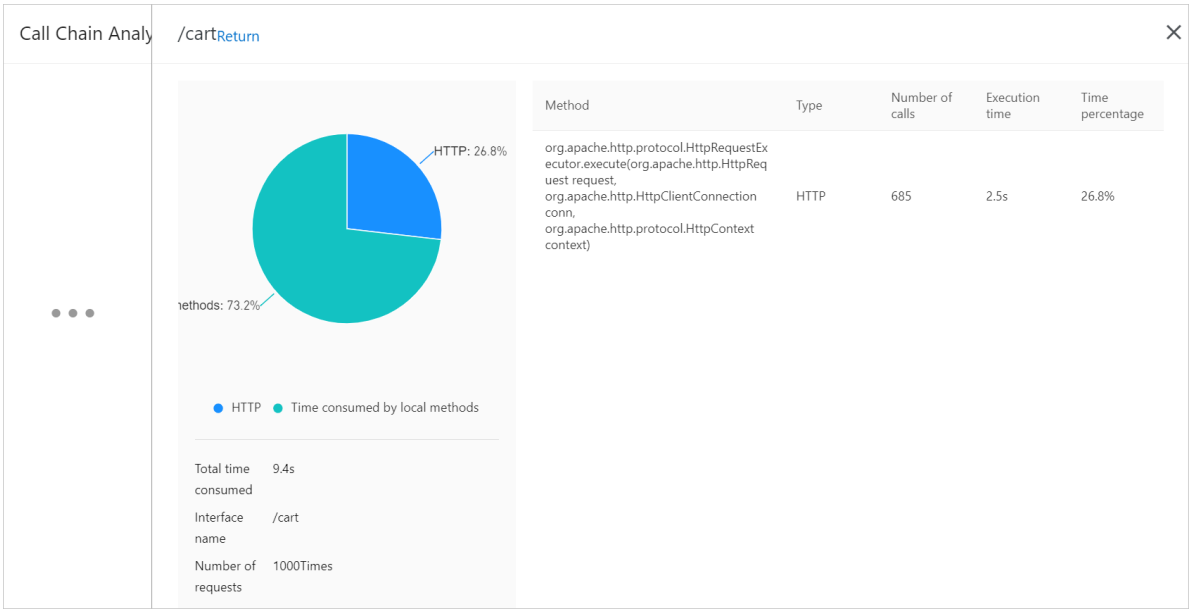
Analyze traces

On the **Call link query** page, select all the traces that you want to analyze and click **Analyze the selected call link**.
In the **Call Chain Analysis** panel, you can view the span name, application name, call type, number of requests, proportion of requests, number of exceptions, proportion of exceptions, average self-consumed time, and average consumed time of all the selected traces.


Call Chain Analysis

Span name	Application Name	调用类型	Number of requests/proportion of requests	Number of exceptions/proportion of exceptions	Average self-consumed time/proportion	Average time consumption	Operation
/cart	frontend	HTTP entry	22 / 100.00%	0 / 0.00%	16.2ms	18.2ms	Statistical ana
com.alibabacloud.hipstershop.cartserviceapi.service.CartService@viewCart	cart	Provide DUBBO	18 / 68.18%	0 / 0.00%	1.7ms	1.7ms	Statistical ana
com.alibabacloud.hipstershop.cartserviceapi.service.CartService@addItemToCart	cart	Provide DUBBO	5 / 22.72%	0 / 0.00%	3ms	3ms	Statistical ana
com.alibabacloud.hipstershop.cartserviceapi.service.CartService@viewCart	cart	Provide DUBBO	12 / 100.00%	0 / 0.00%	2.08ms	2.08ms	Statistical ana
/	frontend	HTTP entry	8 / 100.00%	1 / 12.50%	270.6ms	281ms	Statistical ana

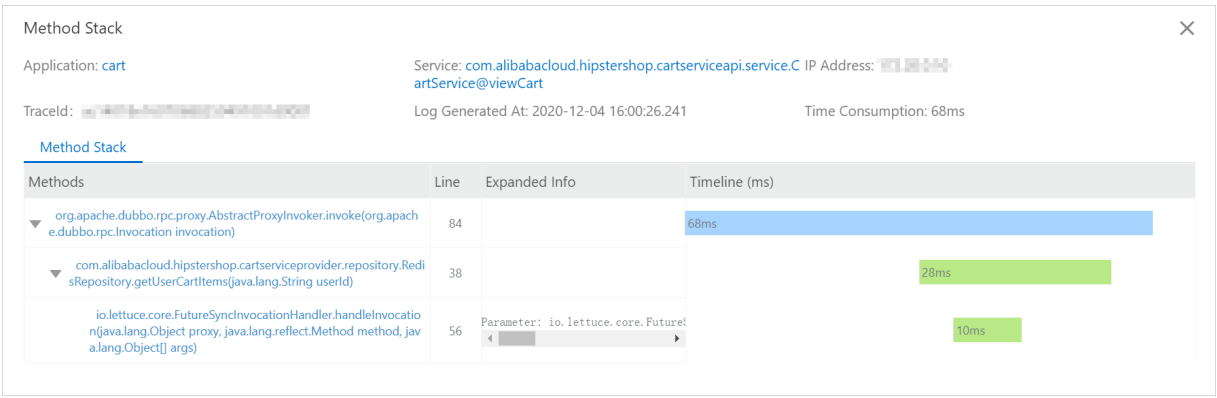
- Move the pointer over a span name to view the trace ID that contains the span.
- Click an application name to go to the **Application Overview** page of the application.
- Click **Statistical analysis** in the **Actions** column corresponding to a span to view details about the span. The details include the proportions of different call types for each operation, total consumed time, operation name, number of requests, recommended samples, the number of times that each calling method is used, as well as the name, type, execution time, and time percentage of each calling method.



Related operations

On the **Trace Details** page, click the line chart in the **Metric Monitored** column to view the number of requests, response time, and number of errors during different periods of time.

On the **Trace Details** page, click the icon in the **Method Stack** column. The **Method Stack** dialog box appears.



The following section describes the parameters in the Method Stack dialog box:

- Calling Method: the method used to call the local method stack. When the Calling Method section is shown, the next calls of the method are displayed.
- Line Number: the number of the line where the code of the local method is located.
- Extended Information:
  - Parameter: input parameters of the call
  - SQL: SQL statements to call the database
  - Exception: exception details
- Timeline: the distribution of time consumed by each method call of the local trace.

3.3. Application overview

On the **Application Overview** page, you can view the health metrics of an application. You can view the upstream and downstream dependent components of an application in an application topology graph. You can also view the health status of the application, services, and hosts in a 3D topology.The health metrics of an application include general metrics such as Total Requests and Average Response Time, metrics related to the application services and dependent services, and system information such as the CPU utilization and memory usage.

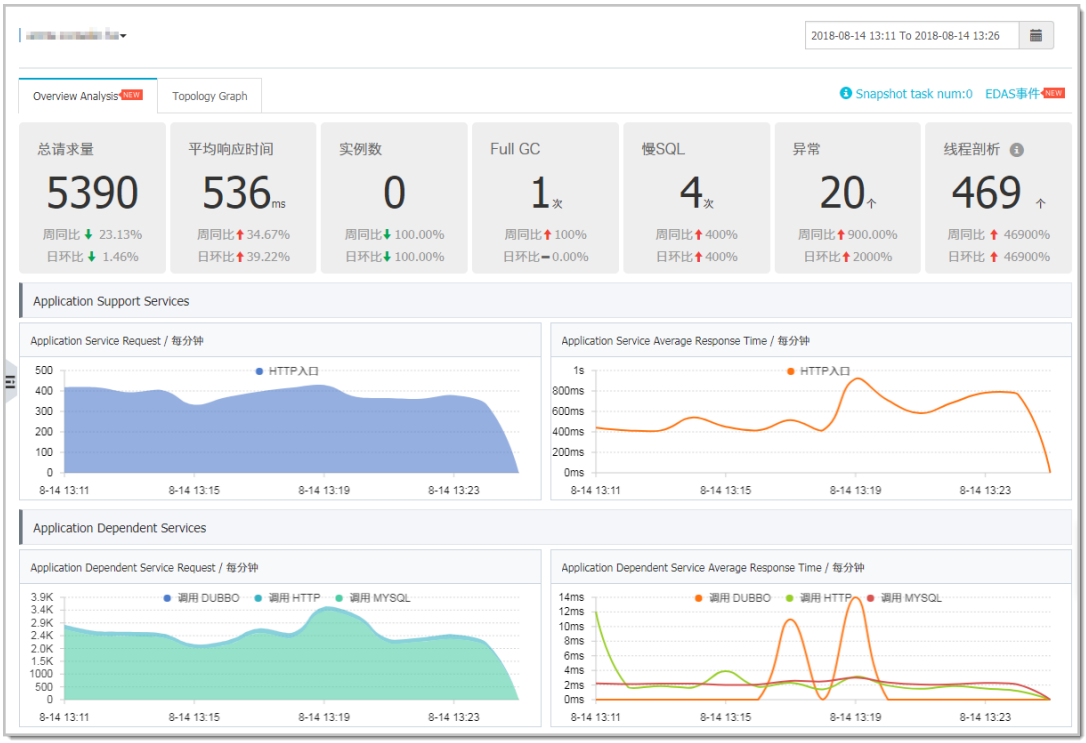
Procedure

1. Log on to the ARMS console .
2. In the left-side navigation pane, choose **Application Monitoring > Application**, and select a region in the top navigation bar.
3. On the **Applications** page, click the application that you want to view to go to the **Application Overview** page.
   On the **Application Overview** page, view information on the **Overview**, **Topology**, and **3D Topology (Beta)** tabs.

Overview

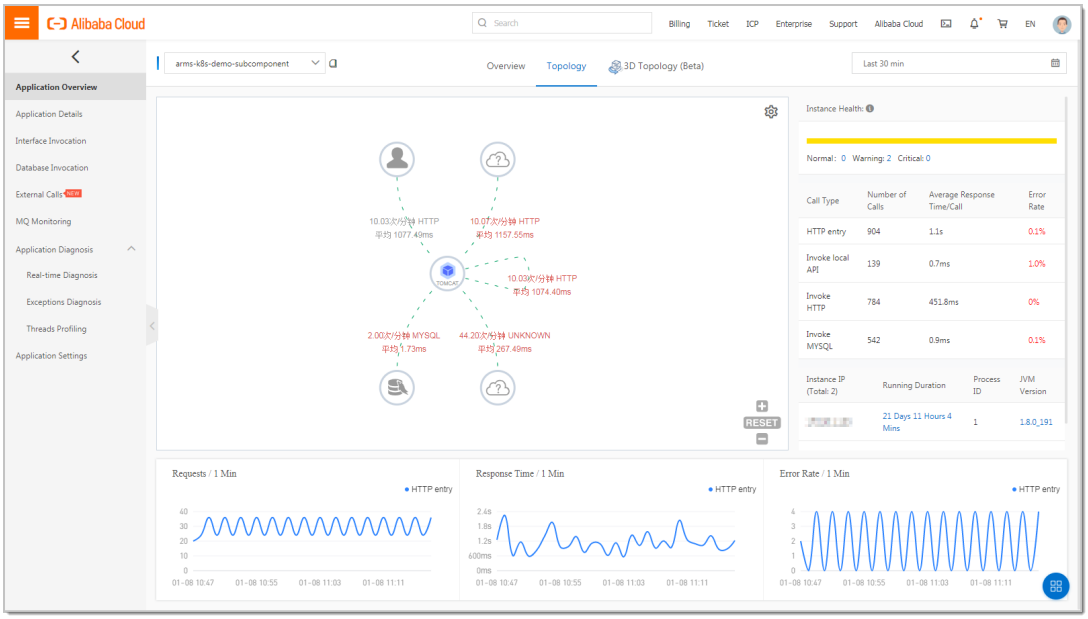
The following key metrics are displayed on the **Overview** tab:

- Total Requests, Average Response Time, Errors, Real Time Instance Count, Full GC, Slow SQL, Exceptions, Thread Profiling. You can also check how the values of these metrics have changed since the last week or last day on this tab.
- Application Events: application events, such as 0-1 alerts, application monitoring alerts, and Kubernetes cluster events.
- Application Support Services: time sequence curves for Application Service Request and Application Service Average Response Time.
- Application Dependent Services: time sequence curves for Application Dependent Service Request, Application Dependent Service Average Response Time, App Instance Count, and HTTP - Status Code.
- System Info: time sequence curves for CPU, MEM, and Load.
- Thread Profiling: time sequence curves and details about Slow Calls.
- Statistical Analysis: analysis on Interface with Slow Calls and Exception.



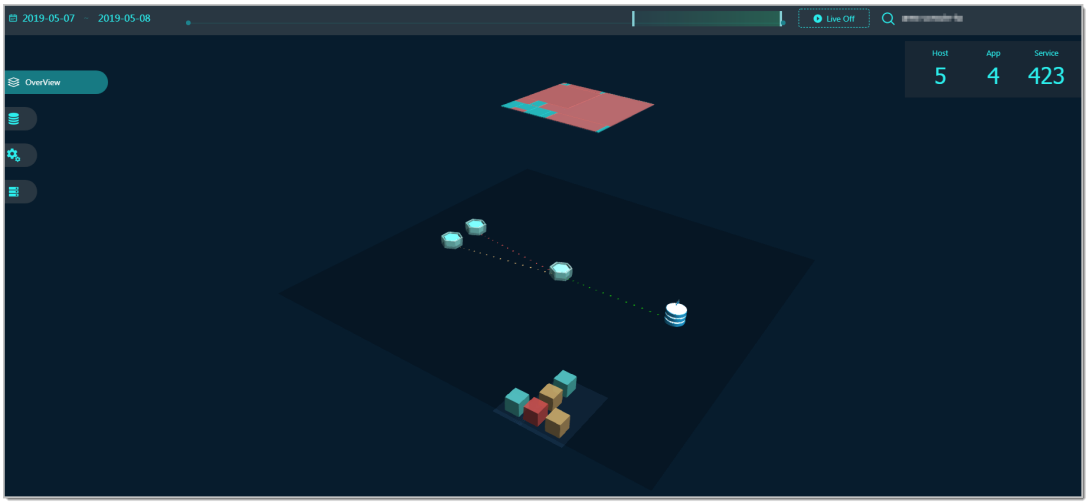
Application topology

On the **Topology** tab, you can obtain a better view of the upstream and downstream components of your application and the call relationship between the components and the application. Then, you can identify the bottlenecks of your application.



3D Topology

On the **3D Topology (Beta)** tab, the health status of an application, services, and hosts, and the upstream and downstream dependencies of the application are displayed. You can use the 3D topology to identify the services that caused failures, applications affected by the failures, and associated hosts. The 3D topology helps you diagnose the root causes of failures and troubleshoot these failures. For more information about 3D topology, see 3D topology.



3.4. Application details

3.4.1. Overview

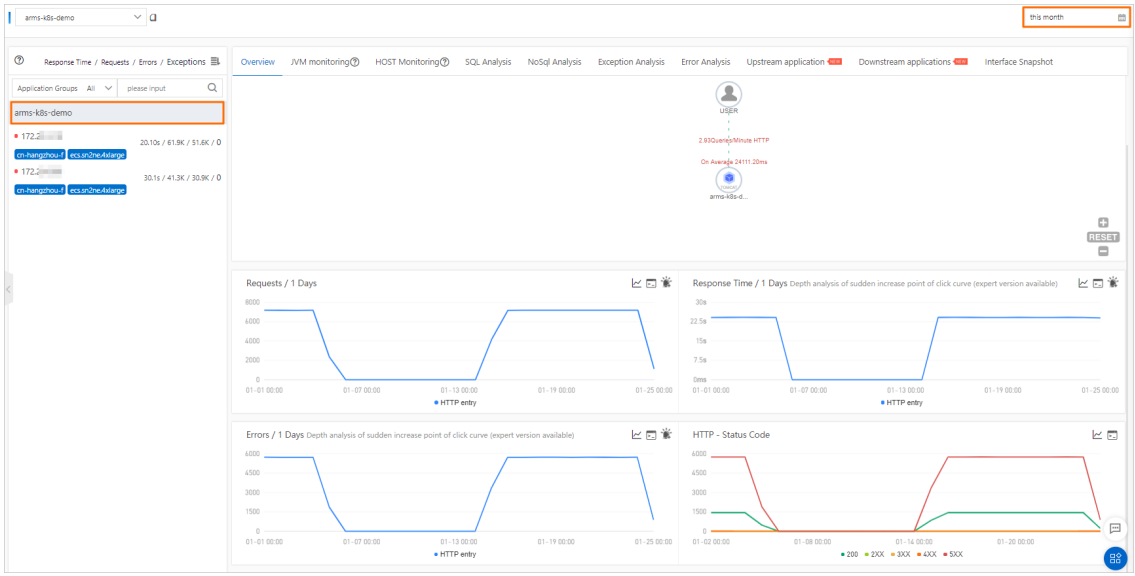
This topic shows you how to obtain an overview of an application. On the Overview tab of the application, you can view the application topology, number of requests, response time, number of errors, and HTTP status codes.

Prerequisites

The Application Real-Time Monitoring Service (ARMS) agent is installed for an application. For more information, see Overview.

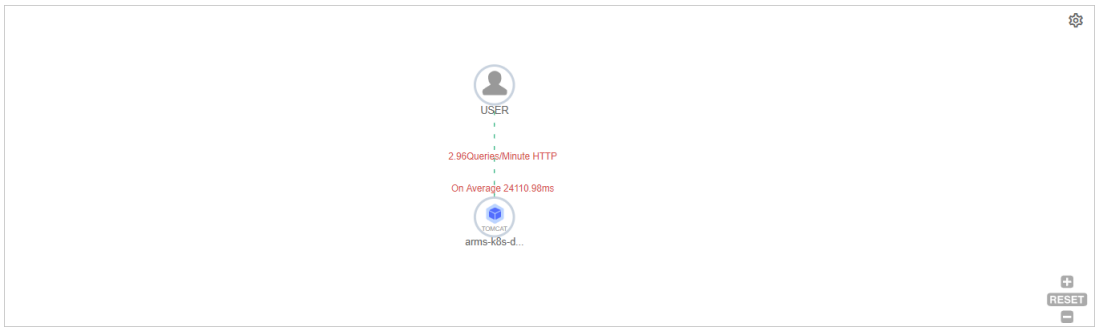
Procedure

1. Log on to the ARMS console .
2. In the left-side navigation pane, Select **Application monitoring > Applications** .
3. In the top navigation bar of the MNS console, select the region where your cluster is deployed.
4. Log on to the **Applications** Page, click the application name.
5. Log on to the **Left-side navigation pane** Place place your cursor over the vertical dots next to **Application details** .
6. On the **Application Details** page, select an instance where the application is deployed and set the time period. The **Overview** tab appears by default.



Application topology

The application topology section displays the topology of call relationships between internal services of the application in the specified time period.



1. (Optional)In the application topology section, perform the following operations as required:
   - Click the ⚙ icon to configure the display settings of the application topology.

     > **Note** The settings are stored in the browser and remain effective the next time you access the Overview tab.

   - Click the plus sign or scroll the mouse wheel up to zoom in the application topology.
   - Click the minus sign or scroll the mouse wheel down to zoom out the application topology.
   - Click the RESET icon to restore the application topology to the default size.

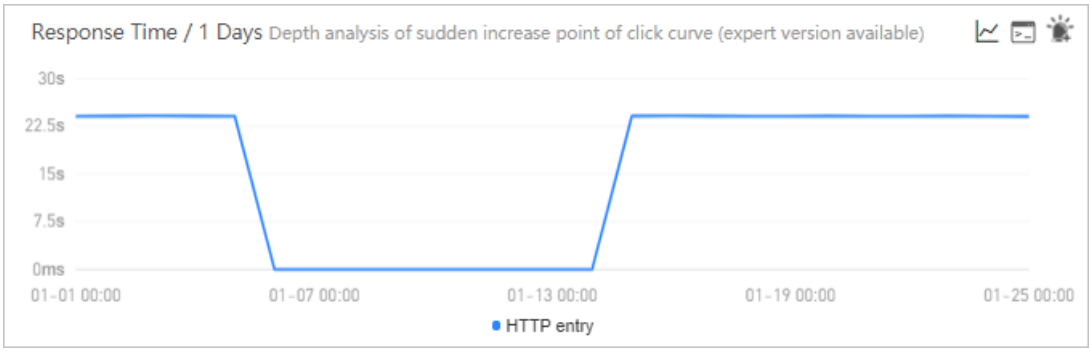
Number of requests

The **Requests** section displays the time series curve that indicates the number of requests of the application in the specified time period.

1. (Optional)In the **Requests** section, perform the following operations as required:
   - Move the cursor over the statistics chart to view the statistics.
   - Select a period of time to view the statistics for the specified period.
   - Click Icon to view the statistics of the metric in a certain time period or compare the statistics of the same time period on different dates.
   - Click Icon to view the API details for this metric.
   - Click Icon to create an alarm for the metric. For more information, see Create an alert.

Response time

The **Response Time** section displays the time series curve that indicates the response time of the application in the specified time period.



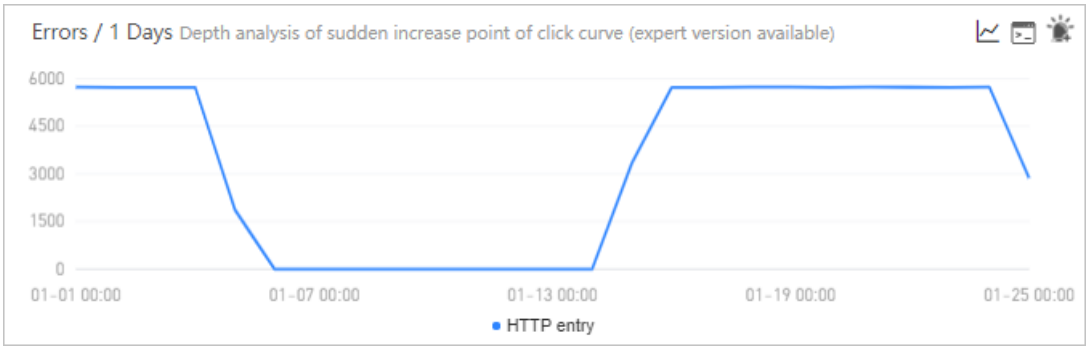
1. (Optional)In the **Response Time** section, perform the following operations as required:
   - Move the cursor over the statistics chart to view the statistics.
   - Select a period of time to view the statistics for the specified period.
   - Click the spike point of the curve for deep analysis.

     > **Note** Only the Pro Edition supports this feature.

   - Click Icon to view the statistics of the metric in a certain time period or compare the statistics of the same time period on different dates.
   - Click Icon to view the API details for this metric.
   - Click Icon to create an alarm for the metric. For more information, see Create an alert.

Number of errors

The **Errors** section displays the time series curve that indicates the number of errors of the application in the specified time period.



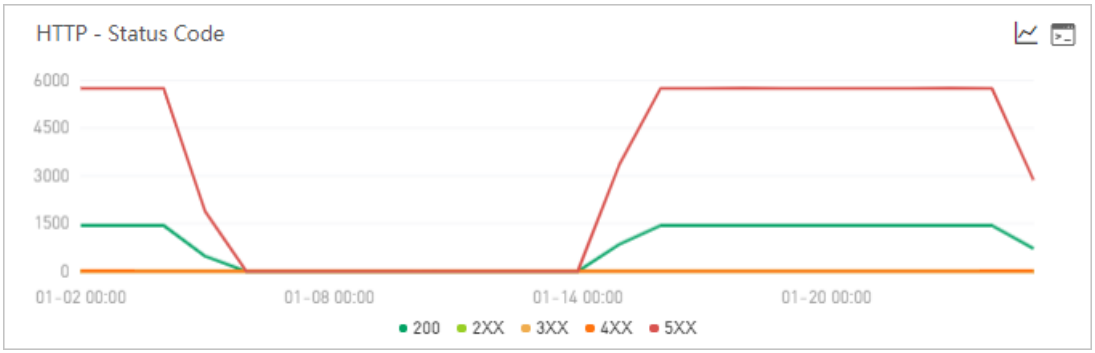
1. (Optional)In the **Errors** section, perform the following operations as required:
   - Move the cursor over the statistics chart to view the statistics.
   - Select a period of time to view the statistics for the specified period.
   - Click the spike point of the curve for deep analysis.

     > **Note** Only the Pro Edition supports this feature.

   - Click Icon to view the statistics of the metric in a certain time period or compare the statistics of the same time period on different dates.
   - Click Icon to view the API details for this metric.
   - Click Icon to create an alarm for the metric. For more information, see Create an alert.

HTTP status code

The **HTTP - Status Code** section displays the time series curve that indicates the HTTP status code statistics of the application in the specified time period.

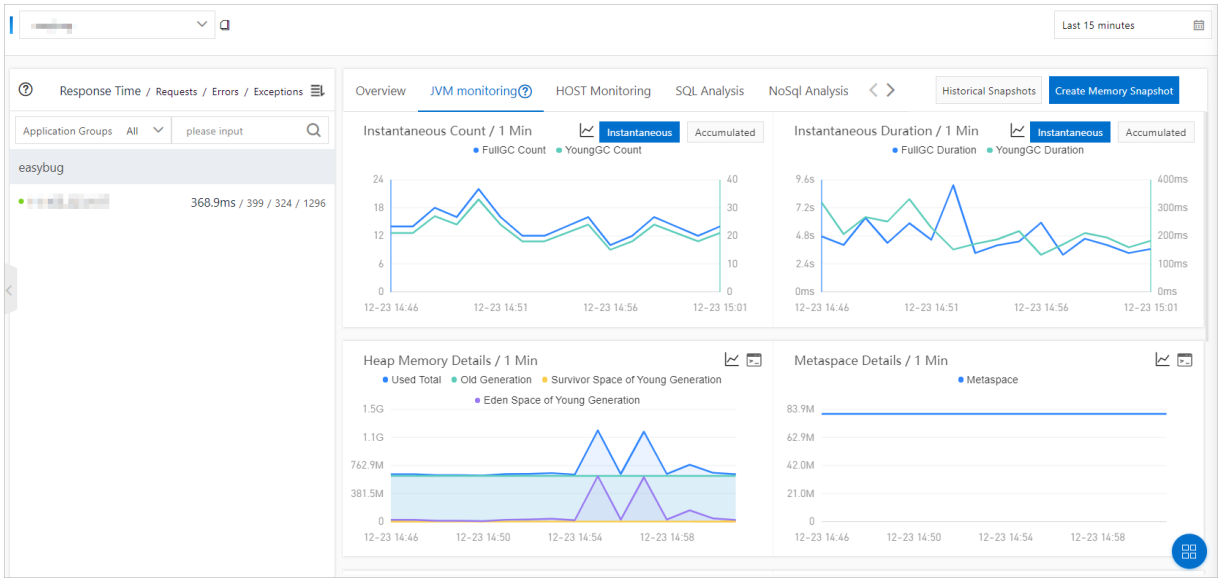


1. (Optional)In the **HTTP - Status Code** section, perform the following operations as required:
   - Move the cursor over the statistics chart to view the statistics.
   - Select a period of time to view the statistics for the specified period.
   - Click legend to hide or show the data.
   - Click Icon to view the statistics of the metric in a certain time period or compare the statistics of the same time period on different dates.

- Click Icon to view the API details for this metric.

3.4.2. JVM monitoring

The Java Virtual Machine (JVM) monitoring feature allows you to monitor critical JVM metrics. The critical metrics include heap metrics, non-heap metrics, direct buffer metrics, memory-mapped buffer metrics, garbage collection (GC) details, and JVM thread count. This topic describes the JVM monitoring feature and how to monitor JVM metrics.



Features

The JVM monitoring feature allows you to monitor the following metrics:

- Instantaneous and accumulated GC details
  - Total times of GC
  - Times of young GC
  - Total time consumption of GC
  - Time consumption of young GC
- Heap memory details
  - Total heap memory
  - Bytes of old heap memory
  - Bytes of young heap memory (Survivor)
  - Bytes of young heap memory (Eden)
- Non-heap memory
  - Submitted bytes of the non-heap memory
  - Initial bytes of the non-heap memory
  - Maximum bytes of the non-heap memory
- Metaspace
  Bytes of metaspace
- Direct buffer

- Total bytes of direct buffer
- Used bytes of direct buffer

- Number of JVM threads
  - Total number of threads
  - Number of deadlocked threads
  - Number of new threads
  - Number of blocked threads
  - Number of runnable threads
  - Number of terminated threads
  - Number of threads in timed waiting
  - Number of waiting threads

View JVM metrics

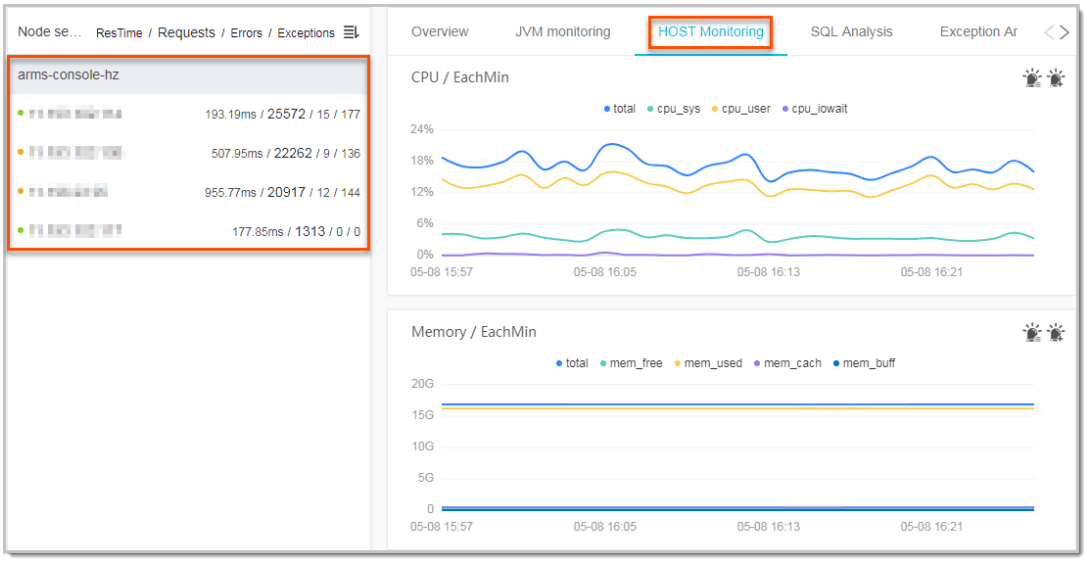
1. Log on to the ARMS console .
2. In the left-side navigation pane, choose **Application Monitoring > Applications**. At the top of the Applications page, select a region.
3. On the **Applications** page, click the application that you want to view.
4. In the left-side navigation pane, click **Application Details**.
5. On the **Application Details** page, select the node and click **JVM monitoring** tab on the right side of the page.On the **JVM monitoring** tab, the time sequence curves of the instantaneous GC count, instantaneous GC duration, heap memory details, metadata details, non-heap memory details, direct buffer, and JVM threads are displayed.
   - Click **Instantaneous** and **Accumulated** in the upper-right corner of **Instantaneous Count / 1 Min** and **Instantaneous Duration / 1 Min** panels. You can view the time sequence curves of the instantaneous GC count or accumulated GC count. You can also view the time sequence curves of instantaneous GC duration.
   - Click a metric name (for example, the total times of GC) on a monitoring panel to enable or disable the visibility of the metric in the chart.

     > **Note** Each chart must contain at least one visible metric. If only one metric is displayed in the chart, you cannot disable the visibility of the metric.

   - Click View API in the upper-right corner of **Heap Memory Details / 1 Min**, **Metadata Details / 1 Min**, **Non-Heap Memory / 1 Min**, **Direct Buffer / 1 Min**, and **JVM Threads / 1 Min** panels to view the API details of the monitoring metric.

3.4.3. Host monitoring

The host monitoring feature is used to monitor the metrics of CPU, memory, disk, load, network traffic, and network packets. This topic describes the host monitoring feature and how to view host monitoring metrics.



Features

The host monitoring feature can monitor the following metrics:

- CPU
  - Total CPU usage
  - System CPU usage
  - User CPU usage
  - Usage of CPU waiting for I/O
- Memory
  - Total memory
  - Free memory
  - Used memory
  - Memory in PageCache
  - Memory in BufferCache
- Disk
  - Total disk space (bytes)
  - Free disk space (bytes)
  - Used disk space (bytes)
- Load
  System load
- Network traffic
  - Received network traffic (bytes)
  - Sent network traffic (bytes)
- Network packets
  - Received packets per minute
  - Sent packets per minute
  - Received errors per minute

- Discarded packets per minute

View host monitoring metrics

1. Log on to the ARMS console .
2. In the left-side navigation pane, choose **Application Monitoring > Applications**. In the top navigation bar, select a region.
3. On the **Applications** page, click the application that you want to monitor.
4. In the left-side navigation pane, click **Application Details**.
5. On the **Application Details** page, click the node that you want to view, and then click the **HOST Monitoring** tab on the right side.
   On the **HOST Monitoring** tab, you can view the time sequence curves of metrics including CPU, memory, disk, load, network traffic, and network packets.
   - You can click the name of a metric such as cpu_sys on each monitoring panel to toggle the visibility of this metric.

     > **Note** Each chart must contain at least one visible metric. Therefore, if only one metric is displayed on a monitoring panel, it cannot be set to invisible.

   - You can click the line chart icon in the upper-right corner to view the metrics by range or compare the metrics.
   - You can click the View API icon in the upper-right corner to view the detailed information about the API operations related to the metric.
   - You can click the or icon in the upper-right corner of the monitoring panel to create an alert or view the existing alert points. For more information about how to create an alert, see Create an alert.

3.4.4. Pod monitoring

This topic shows you how to view the monitored metrics of a pod, including the metrics of CPU, physical memory, network traffic, and network packets.

Prerequisites

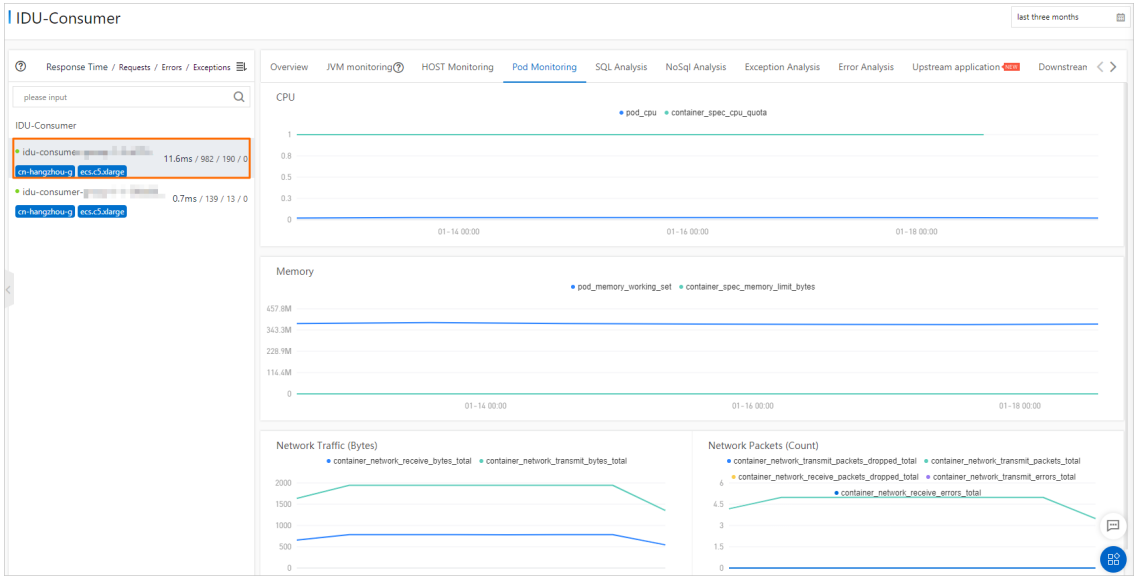
The Application Real-Time Monitoring Service (ARMS) agent is installed for an application. For more information, see Overview.

> **Note** Pod metrics are available only for applications that are deployed in pods.

Procedure

1. Log on to the ARMS console .
2. In the left-side navigation pane, Select **Application monitoring > Applications** .
3. In the top navigation bar of the MNS console, select the region where your cluster is deployed.
4. Log on to the **Applications** Page, click the application name.
5. Log on to the **Left-side navigation pane** Place place your cursor over the vertical dots next to **Application details** .
6. On the **Application Details** page, select a pod where the application is deployed, set the time

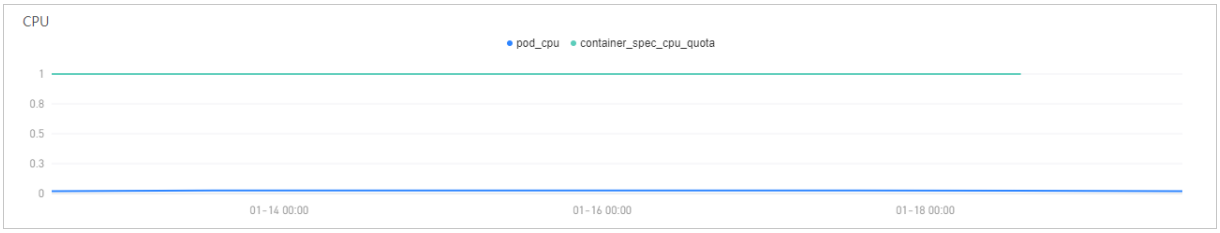
period, and then click the **Pod Monitoring** tab.



CPU

The **CPU** section displays the CPU metrics of the pod where the application is deployed in the specified time period, including the following metrics:

- Cumulative CPU usage
- CPU quota

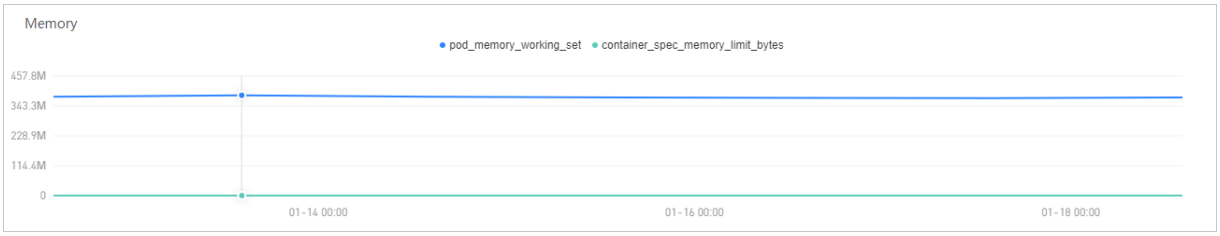


1. (Optional)In the **CPU** section, perform the following operations as required:
   - Move the cursor over the statistics chart to view the statistics.
   - Select a period of time to view the statistics for the specified period.
   - Click legend to hide or show the data.

Physical memory

The **Memory** section displays the physical memory metrics of the pod where the application is deployed in the specified time period, including the following metrics:

- Memory usage
- Memory quota

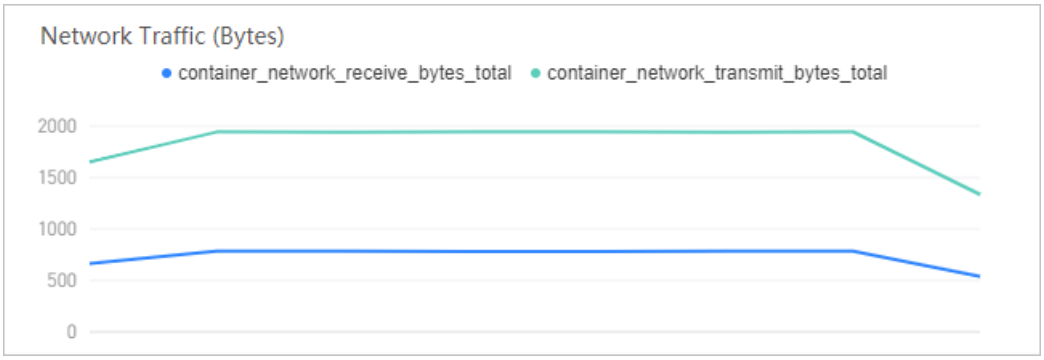


1. (Optional)In the **Memory** section, perform the following operations as required:
   - Move the cursor over the statistics chart to view the statistics.
   - Select a period of time to view the statistics for the specified period.
   - Click legend to hide or show the data.

Network traffic

The **Network Traffic** section displays the network traffic metrics of the pod where the application is deployed in the specified time period, including the following metrics:

- Received network traffic (bytes)
- Sent network traffic (bytes)

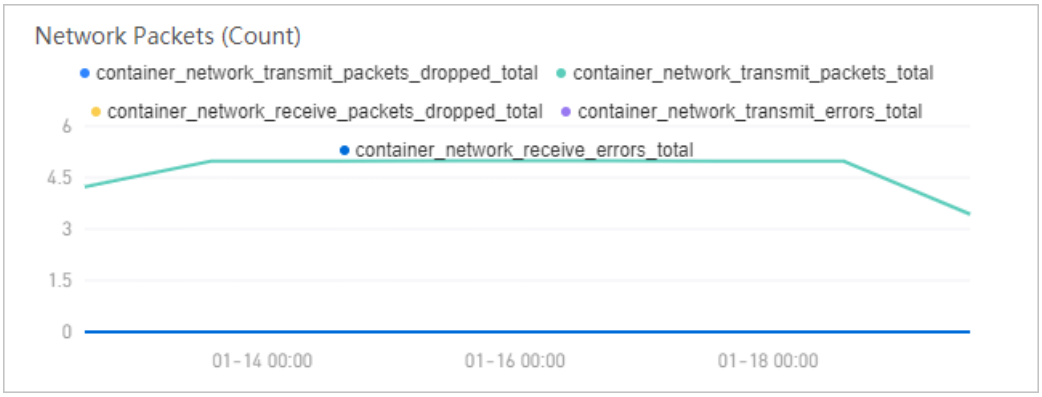


1. (Optional)In the **Network Traffic** section, perform the following operations as required:
   - Move the cursor over the statistics chart to view the statistics.
   - Select a period of time to view the statistics for the specified period.
   - Click legend to hide or show the data.

Network packets

The **Network Packets** section displays the network packet metrics of the pod where the application is deployed in the specified time period, including the following metrics:

- Number of discarded network packets among the sent network packets
- Number of sent network packets
- Number of discarded network packets among the received network packets
- Number of errors that occur when network packets are sent
- Number of errors that occur when network packets are received



1. (Optional)In the **Network Packets** section, perform the following operations as required:

- Move the cursor over the statistics chart to view the statistics.
- Select a period of time to view the statistics for the specified period.
- Click legend to hide or show the data.

3.4.5. SQL analysis

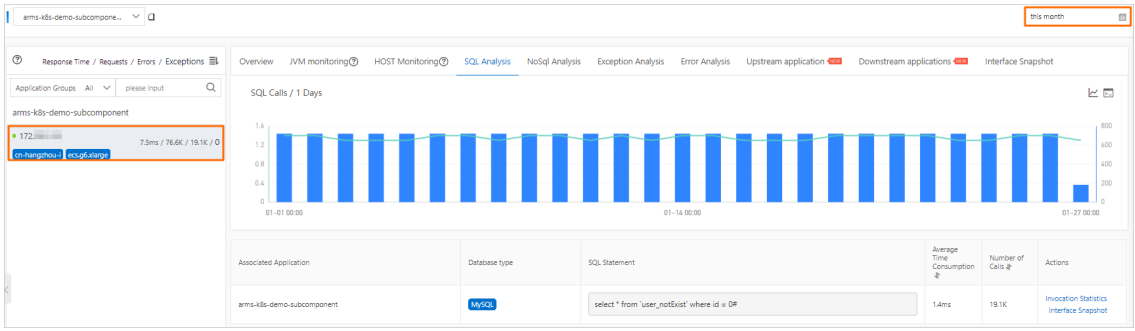
This topic shows you how to view the SQL analysis of an application.

Prerequisites

The Application Real-Time Monitoring Service (ARMS) agent is installed for an application. For more information, see Overview.

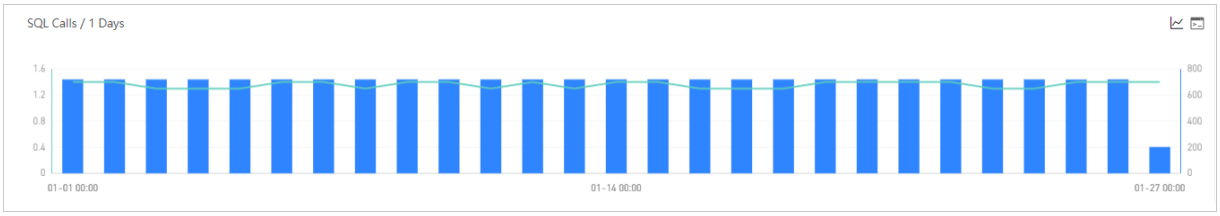
Procedure

1. Log on to the ARMS console .
2. In the left-side navigation pane, Select **Application monitoring > Applications** .
3. In the top navigation bar of the MNS console, select the region where your cluster is deployed.
4. Log on to the **Applications** Page, click the application name.
5. Log on to the **Left-side navigation pane** Place place your cursor over the vertical dots next to **Application details** .
6. On the **Application Details** page, select an instance where the application is deployed, set the time period, and then click the **SQL Analysis** tab.



SQL call statistics

The **SQL Calls** section displays the time series curve that indicates the SQL call statistics of the application in the specified time period.

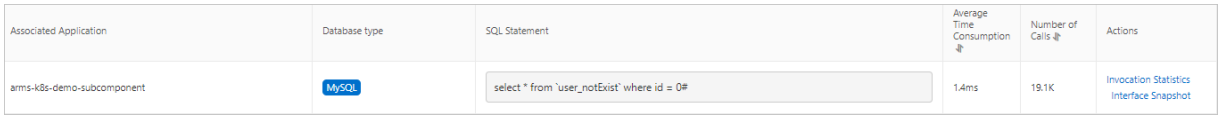


1. (Optional)In the **SQL Calls** section, perform the following operations as required:
   - Move the cursor over the statistics chart to view the statistics.
   - Select a period of time to view the statistics for the specified period.
   - Click Icon to view the statistics of the metric in a certain time period or compare the statistics of the same time period on different dates.

- Click Icon to view the API details for this metric.

SQL statement list

The SQL statement list displays all SQL statements that are executed in the application in the specified time period.



1. (Optional)In the SQL statement list, perform the following operations as required:
   - To view the time series curve of the SQL call statistics of an SQL statement, click **Invocation Statistics** in the **Actions** column of the SQL statement.
   - To view the snapshots of the operation that is called by an SQL statement, click **Interface Snapshot** in the **Actions** column of the SQL statement.
     For more information, see Operation snapshot.

3.4.6. NoSQL analysis

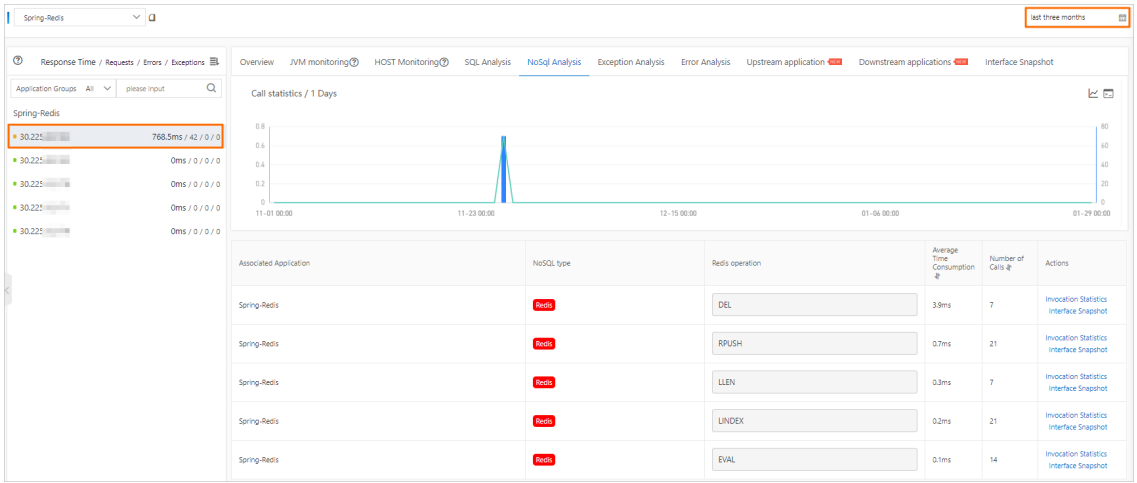
This topic shows you how to view the NoSQL analysis of an application.

Prerequisites

The Application Real-Time Monitoring Service (ARMS) agent is installed for an application. For more information, see Overview.

Procedure

1. Log on to the ARMS console .
2. In the left-side navigation pane, Select **Application monitoring > Applications** .
3. In the top navigation bar of the MNS console, select the region where your cluster is deployed.
4. Log on to the **Applications** Page, click the application name.
5. Log on to the **Left-side navigation pane** Place place your cursor over the vertical dots next to **Application details** .
6. On the **Application Details** page, select an instance where the application is deployed, set the time period, and then click the **NoSQL Analysis** tab.



NoSQL call statistics

The **SQL Calls** section displays the time series curve that indicates the NoSQL call statistics of the application in the specified time period.

1. (Optional)In the **SQL Calls** section, perform the following operations as required:
   - Move the cursor over the statistics chart to view the statistics.
   - Select a period of time to view the statistics for the specified period.
   - Click Icon to view the statistics of the metric in a certain time period or compare the statistics of the same time period on different dates.
   - Click Icon to view the API details for this metric.

Operation command list

The operation command list displays all operation commands that are run in NoSQL calls of the application in the specified time period.

Associated Application	NoSQL type	Redis operation	Average Time Consumption	Number of Calls	Actions
Spring-Redis	Redis	DEL	3.9ms	7	Invocation Statistics Interface Snapshot
Spring-Redis	Redis	RPUSH	0.7ms	21	Invocation Statistics Interface Snapshot
Spring-Redis	Redis	LLEN	0.3ms	7	Invocation Statistics Interface Snapshot
Spring-Redis	Redis	LINDEX	0.2ms	21	Invocation Statistics Interface Snapshot
Spring-Redis	Redis	EVAL	0.1ms	14	Invocation Statistics Interface Snapshot

1. (Optional)In the operation command list, perform the following operations as required:
   - To view the NoSQL call statistics of an operation command, click **Invocation Statistics** in the **Actions** column of the operation command.
   - To view the snapshots of the operation that is called by an operation command, click **Interface Snapshot** in the **Actions** column of the operation command.
     For more information, see Operation snapshot.

3.4.7. Exception analysis

This topic shows you how to view the exception analysis of an application.

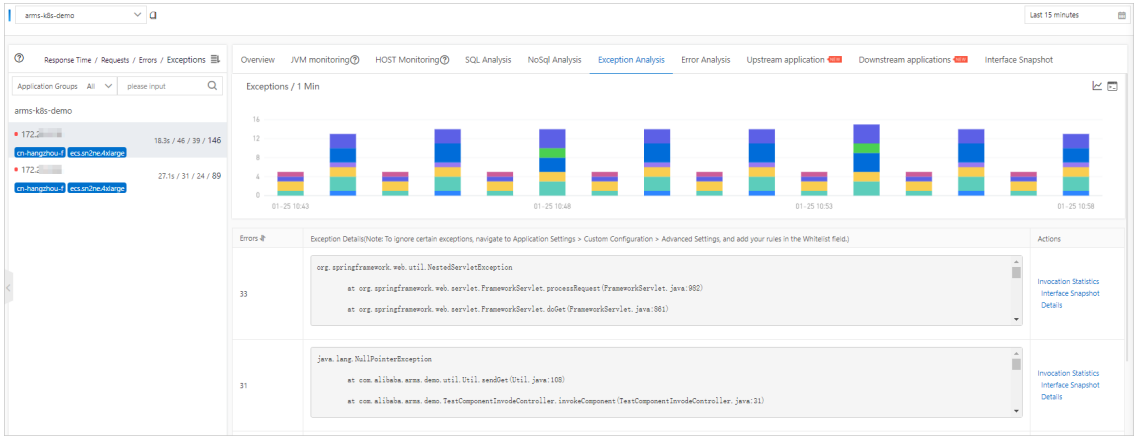
Prerequisites

The Application Real-Time Monitoring Service (ARMS) agent is installed for an application. For more information, see Overview.

Procedure

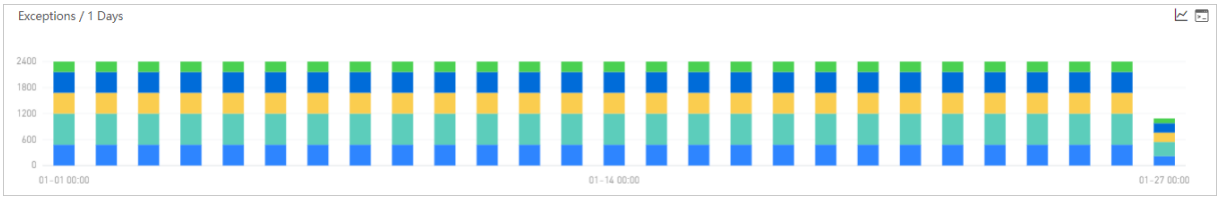
1. Log on to the ARMS console .

2. In the left-side navigation pane, Select **Application monitoring > Applications** .
3. In the top navigation bar of the MNS console, select the region where your cluster is deployed.
4. Log on to the **Applications** Page, click the application name.
5. Log on to the **Left-side navigation pane** Place place your cursor over the vertical dots next to **Application details** .
6. On the **Application Details** page, select an instance where the application is deployed, set the time period, and then click the **Exception Analysis** tab.



Exception statistics

The **Exceptions** section displays the stacked column chart of the exception statistics of the application in the specified time period and the exception list.

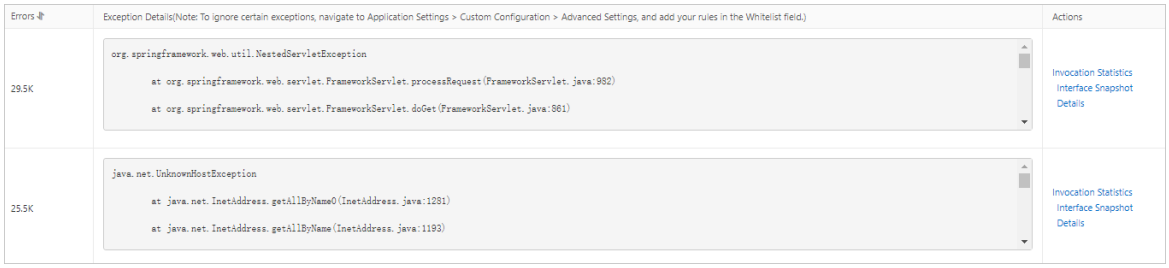


1. (Optional)In the **Exceptions** section, perform the following operations as required:
   - Move the cursor over the statistics chart to view the statistics.
   - Select a period of time to view the statistics for the specified period.
   - Click Icon to view the statistics of the metric in a certain time period or compare the statistics of the same time period on different dates.
   - Click Icon to view the API details for this metric.

Exception list

The exception list displays all exceptions of the application in the specified time period.

1. (Optional)In the exception list, perform the following operations as required:



> **Note** To filter exceptions, perform the following steps: In the left-side navigation pane, click **Application Settings**. On the page that appears, click the **Custom Configuration** tab. In the **Advanced Settings** section, set the **Whitelist** field.

- To view the stacked column chart of an exception, click **Invocation Statistics** in the **Actions** column of the exception.
- To view the snapshots of the operations that are called when an exception occurs, click **Interface Snapshot** in the **Actions** column of the exception.
  For more information, see Operation snapshot.
- To view the details of an exception, click **Details** in the **Actions** column of the exception.

3.4.8. Error analysis

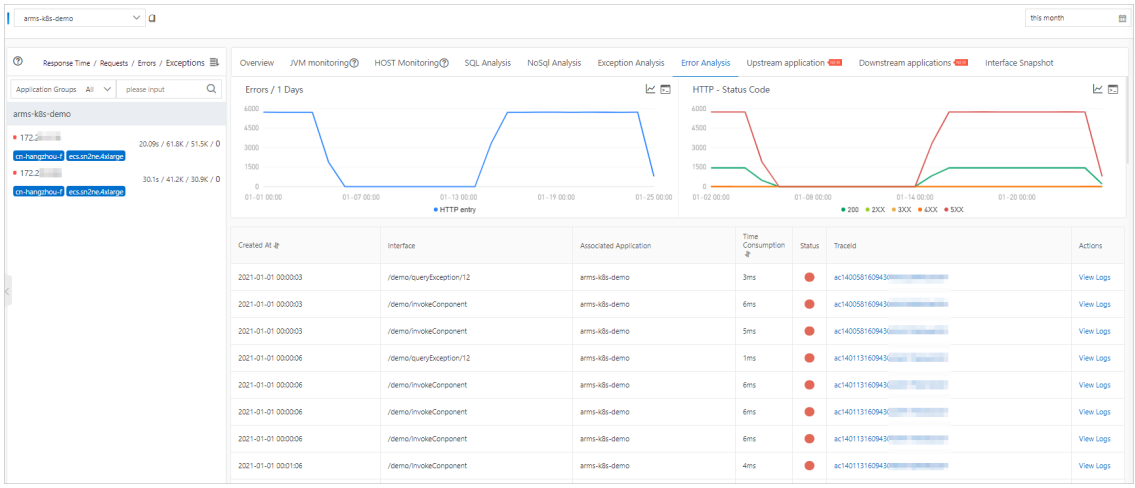
This topic shows you how to view the error analysis of an application.

Prerequisites

The Application Real-Time Monitoring Service (ARMS) agent is installed for an application. For more information, see Overview.

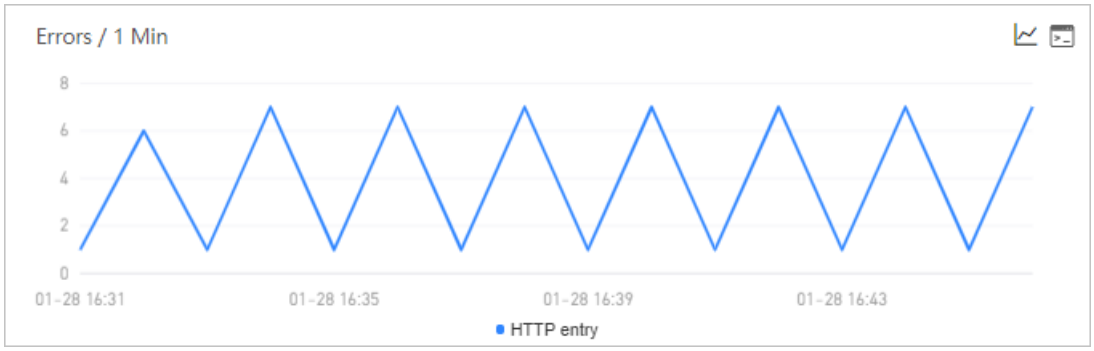
Procedure

1. Log on to the ARMS console .
2. In the left-side navigation pane, Select **Application monitoring > Applications** .
3. In the top navigation bar of the MNS console, select the region where your cluster is deployed.
4. Log on to the **Applications** Page, click the application name.
5. Log on to the **Left-side navigation pane** Place place your cursor over the vertical dots next to **Application details** .
6. On the **Application Details** page, select an instance where the application is deployed, set the time period, and then click the **Error Analysis** tab.



Number of errors

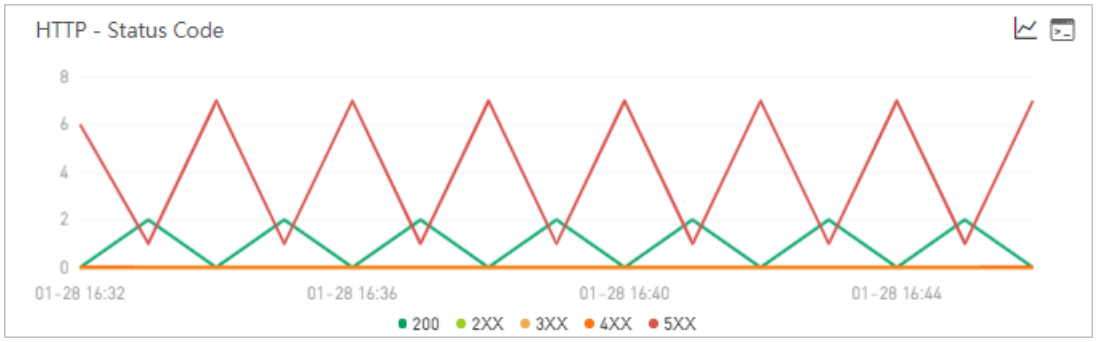
The **Errors** section displays the time series curve that indicates the number of errors of the application in the specified time period.



1. (Optional)In the **Errors** section, perform the following operations as required:
   - Move the cursor over the statistics chart to view the statistics.
   - Select a period of time to view the statistics for the specified period.
   - Click Icon to view the statistics of the metric in a certain time period or compare the statistics of the same time period on different dates.
   - Click Icon to view the API details for this metric.

HTTP status code

The **HTTP - Status Code** section displays the time series curve that indicates the HTTP status code statistics of the application in the specified time period.



1. (Optional)In the **HTTP - Status Code** section, perform the following operations as required:
   - Move the cursor over the statistics chart to view the statistics.
   - Select a period of time to view the statistics for the specified period.
   - Click legend to hide or show the data.
   - Click Icon to view the statistics of the metric in a certain time period or compare the statistics of the same time period on different dates.
   - Click Icon to view the API details for this metric.

Error list

The error list displays all errors of the application in the specified time period.

Created At	Interface	Associated Application	Time Consumption	Status	TraceId	Actions
2021-01-28 16:32:03	/demo/queryException/12	arms-k8s-demo	2ms		ac14005816118	View Logs
2021-01-28 16:32:03	/demo/invokeConponent	arms-k8s-demo	6ms		ac14005816118	View Logs
2021-01-28 16:32:03	/demo/invokeConponent	arms-k8s-demo	6ms		ac14005816118	View Logs
2021-01-28 16:32:06	/demo/queryException/12	arms-k8s-demo	14ms		ac14011316118	View Logs
2021-01-28 16:32:06	/demo/invokeConponent	arms-k8s-demo	24ms		ac14011316118	View Logs
2021-01-28 16:32:06	/demo/invokeConponent	arms-k8s-demo	25ms		ac14011316118	View Logs

1. (Optional)In the error list, perform the following operations as required:
   - To view a trace of an error, click the trace ID in the **TraceId** column of the error.
   - To view the logs of an error, click **View Logs** in the **Actions** column of the error.

3.4.9. Upstream applications

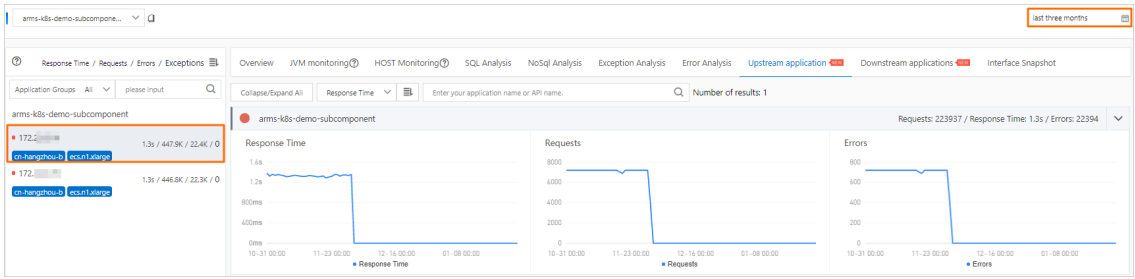
An upstream application of a specific application is an application that sends data to the specific application. This topic shows you how to view the information about upstream applications, including the response time, number of requests, and number of errors.

Prerequisites

The Application Real-Time Monitoring Service (ARMS) agent is installed for an application. For more information, see Overview.

Procedure

1. Log on to the ARMS console .
2. In the left-side navigation pane, Select **Application monitoring > Applications** .
3. In the top navigation bar of the MNS console, select the region where your cluster is deployed.
4. Log on to the **Applications** Page, click the application name.
5. Log on to the **Left-side navigation pane** Place place your cursor over the vertical dots next to **Application details** .
6. On the **Application Details** page, select an instance where the application is deployed, set the time period, and then click the **Upstream application** tab.



Response time

The **Response Time** section displays the time series curve that indicates the response time of the upstream applications of the application in the specified time period.

1. In the **Response Time** section, perform the following operations as required:
   - Move the cursor over the statistics chart to view the statistics.
   - Select a period of time to view the statistics for the specified period.

Number of requests

The **Requests** section displays the time series curve that indicates the number of requests of the upstream applications of the application in the specified time period.

1. In the **Requests** section, perform the following operations as required:
   - Move the cursor over the statistics chart to view the statistics.
   - Select a period of time to view the statistics for the specified period.

Number of errors

The **Errors** section displays the time series curve that indicates the number of errors of the upstream applications of the application in the specified time period.

1. In the **Errors** section, perform the following operations as required:
   - Move the cursor over the statistics chart to view the statistics.
   - Select a period of time to view the statistics for the specified period.

3.4.10. Downstream applications

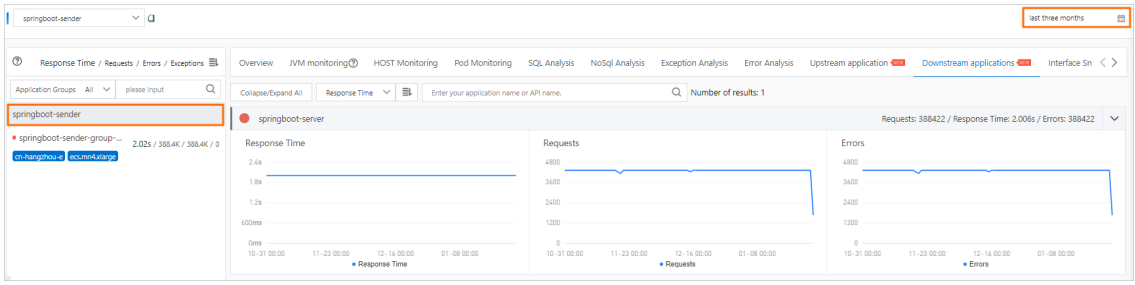
A downstream application of a specific application is an application that receives data from the specific application. This topic shows you how to view the information about downstream applications, including the response time, number of requests, and number of errors.

Prerequisites

The Application Real-Time Monitoring Service (ARMS) agent is installed for an application. For more information, see Overview.

Procedure

1. Log on to the ARMS console .
2. In the left-side navigation pane, Select **Application monitoring > Applications** .
3. In the top navigation bar of the MNS console, select the region where your cluster is deployed.
4. Log on to the **Applications** Page, click the application name.
5. Log on to the **Left-side navigation pane** Place place your cursor over the vertical dots next to **Application details** .
6. On the **Application Details** page, select an instance where the application is deployed, set the time period, and then click the **Downstream applications** tab.



Response time

The **Response Time** section displays the time series curve that indicates the response time of the downstream applications of the application in the specified time period.

1. In the **Response Time** section, perform the following operations as required:
   - Move the cursor over the statistics chart to view the statistics.
   - Select a period of time to view the statistics for the specified period.

Number of requests

The **Requests** section displays the time series curve that indicates the number of requests of the downstream applications of the application in the specified time period.

1. In the **Requests** section, perform the following operations as required:
   - Move the cursor over the statistics chart to view the statistics.
   - Select a period of time to view the statistics for the specified period.

Number of errors

The **Errors** section displays the time series curve that indicates the number of errors of the downstream applications of the application in the specified time period.

1. In the **Errors** section, perform the following operations as required:
   - Move the cursor over the statistics chart to view the statistics.
   - Select a period of time to view the statistics for the specified period.

3.4.11. Operation snapshots

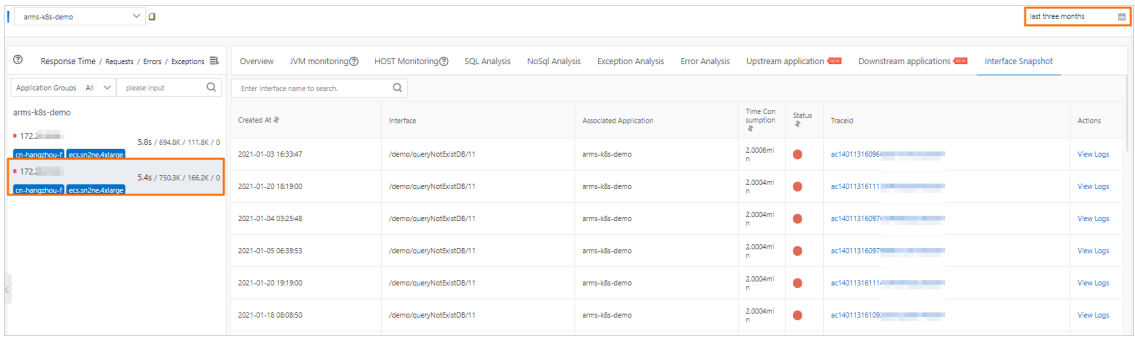
This topic shows you how to view the snapshots of all operations that are called in an application. You can view the time when snapshots are created, the time that is consumed for calling each operation, and the status of each operation.

Prerequisites

The Application Real-Time Monitoring Service (ARMS) agent is installed for an application. For more information, see Overview.

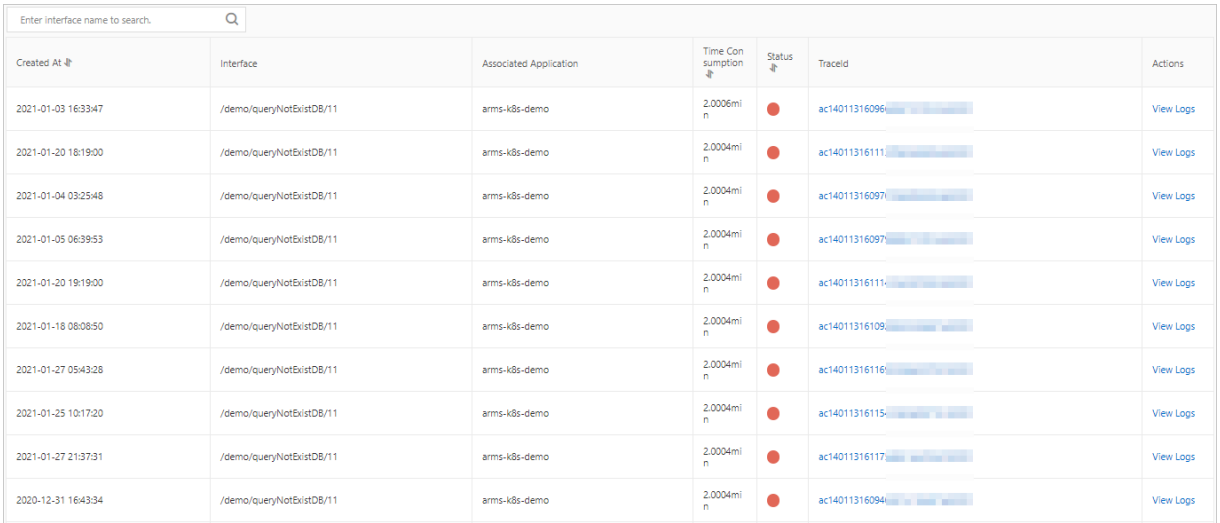
Procedure

1. Log on to the ARMS console .
2. In the left-side navigation pane, Select **Application monitoring > Applications** .
3. In the top navigation bar of the MNS console, select the region where your cluster is deployed.
4. Log on to the **Applications** Page, click the application name.
5. Log on to the **Left-side navigation pane** Place place your cursor over the vertical dots next to **Application details** .
6. On the **Application Details** page, select an instance where the application is deployed, set the time period, and then click the **Interface Snapshot** tab.



Operation snapshot

The **Interface Snapshot** tab lists all operations that are called in the application in the specified time period.



1. (Optional)On the **Interface Snapshot** tab, perform the following operations as required:
   - To view the snapshots of an operation, enter the operation name in the search box and click the icon.
   - To view a trace of an operation, click the trace ID in the **TraceId** column of the operation.
   - To view the logs of an operation, click **View Logs** in the **Actions** column of the operation.

3.4.12. Logs

This topic shows you how to view the pod logs of an application.

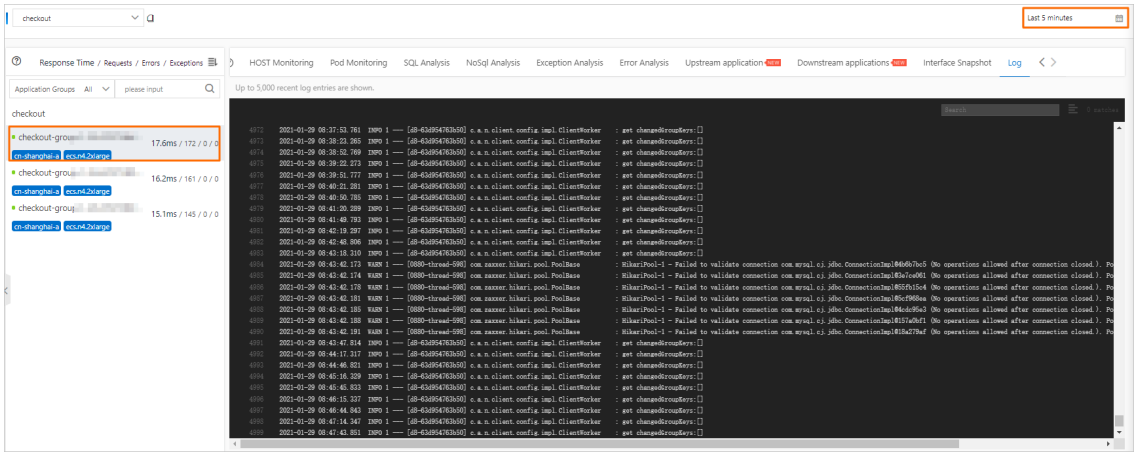
Prerequisites

The Application Real-Time Monitoring Service (ARMS) agent is installed for an application. For more information, see Overview.

> **Note** Pod logs are available only for applications that are deployed in pods.

Procedure

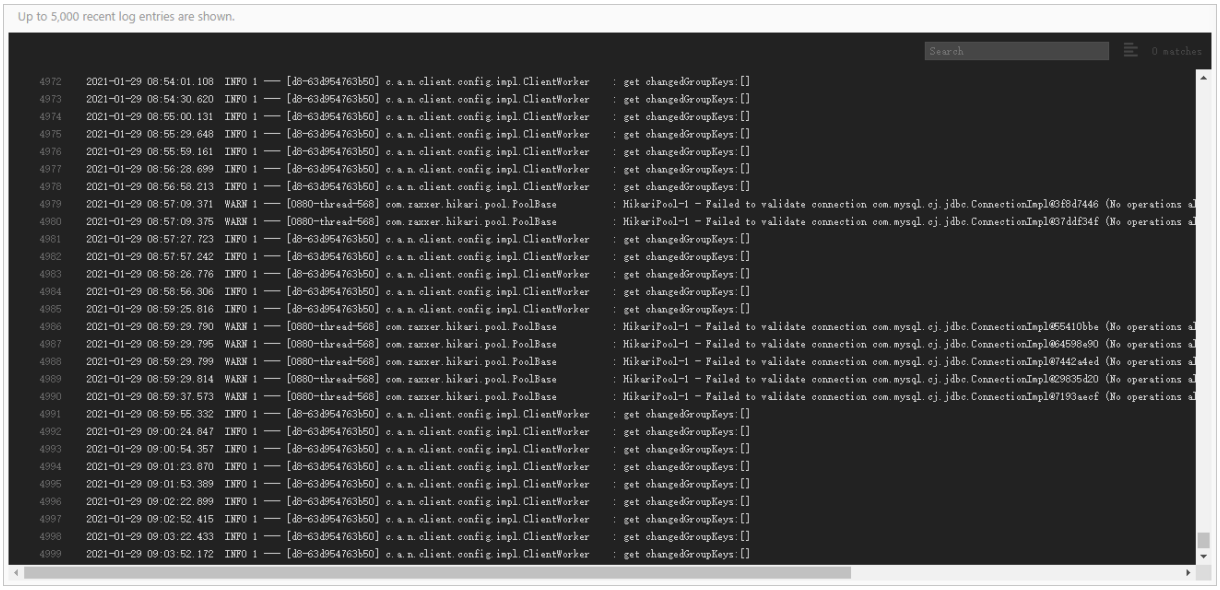
1. Log on to the ARMS console .
2. In the left-side navigation pane, Select **Application monitoring > Applications** .
3. In the top navigation bar of the MNS console, select the region where your cluster is deployed.
4. Log on to the **Applications** Page, click the application name.
5. Log on to the **Left-side navigation pane** Place place your cursor over the vertical dots next to **Application details** .
6. On the **Application Details** page, select a pod where the application is deployed, set the time period, and then click the **Log** tab.



Logs

The **Log** tab displays the latest logs of the pod where the application is deployed.

> **Note** You can view up to 5,000 latest logs.



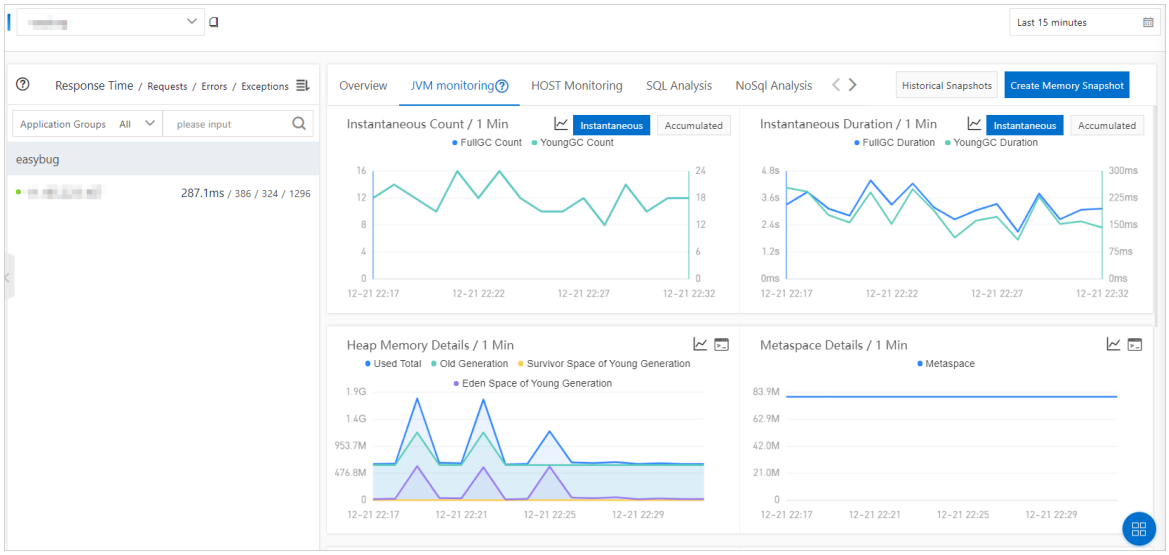
1. On the **Log** tab, enter a keyword in the search box and click the icon to filter logs.

3.4.13. Memory snapshot

JVM monitoring can display multiple memory metrics within a specified period of time. However, although the charts can reflect excessive memory usage, specific information cannot be displayed. Therefore, it cannot help you to troubleshoot problems. You can create a memory snapshot and view detailed memory usage in logs. This can help you troubleshoot memory problems such as memory leakage and memory waste.

Create memory snapshot

1. Log on to the ARMS console .
2. In the left-side navigation pane, choose **Application Monitoring > Applications**. In the top navigation bar, select a region.
3. On the **Applications** page, click the name of the application that you want to manage.
4. In the left-side navigation pane, click **Application Details**, and click **JVM Monitoring** tab on the right side.



5. In the upper-right corner of the **JVM Monitoring** tab, click **Create Memory Snapshot**.

   > **Note** When you click **Create Memory Snapshot**, if the previous snapshot task is still running, an error message is prompted. Wait until the previous snapshot task is finished. You can only create memory snapshots for the Linux system.

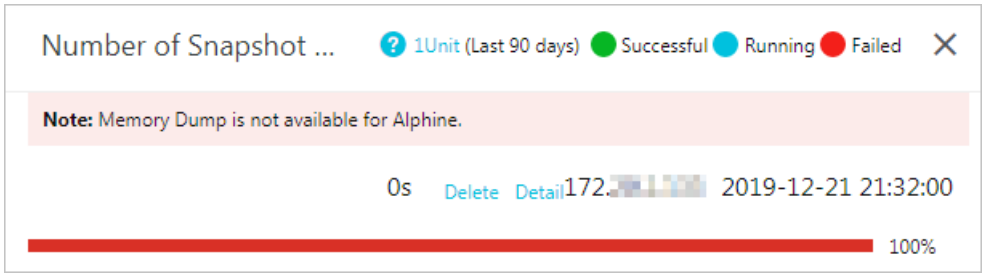
6. In the **Create Memory Snapshot** dialog box, select an IP address and click **Save**.

   > **Warning** The running time of a snapshot task varies from a few minutes to half an hour. The application stops responding during a dump. Proceed with caution.

   > **Note** If you have selected an instance on the left side of the **Application Details** page, the IP address of the instance is selected by default in the **IP** field.

View memory snapshot details

1. In the upper-right corner of the **JVM Monitoring** tab, click **Historical Snapshots**.
   The **Number of Snapshot Jobs** section displays the task execution status. Green indicates that the snapshot task is successful, blue indicates that the snapshot task is executing, and red indicates that the snapshot task fails.
   The name of a snapshot task contains the following information:
   - Memory analysis status
   - The ID of the snapshot task. It consists of an IP address and a timestamp.
   - Snapshot creation time

2. Click **Start Analysis**. In the **Note** message, click **OK**.
3. Click **Analysis Results**. On the memory analysis page that appears, you can view memory analysis details. This can help you troubleshoot memory leaks and reduce memory waste.
   - Click the **Overview** tab to view the heap usage, the number of classes, the number of objects, the number of class loaders, and the number of root objects. You can also view the memory usage displayed in a circular bar.
   - Click the **Leakage Report** tab to view suspicious memory-consuming objects. Click **Problem Suspect** at the lower part of the page to view the corresponding instances, memory usage, and class loading information of a suspect object.
   - Click the **GC Root Object** tab to view all root objects classified by root type and Java class type. Root objects are objects referenced by the GC root, such as static variables or threaded stacks
   - Click the **Dominator Tree** tab to view the dominator relationships among objects in a heap. You can identify objects that consume large amounts of memory and their object dependencies.
   - Click the **Class View** tab to view the heap usage and the number of instances for each heap

type.

- Click the **Unreachable Class View** tab to view the size and type of objects that are not referenced in the heap.
- Click the **Duplicate Class View** tab to view the type of objects loaded by multiple class loaders.
- Click the **Class Loader View** tab to view all class loaders used by the application and the loaded classes, such as the types of loaded classes and the number of instances in a class.
- Click the **Off-heap Memory View** tab to view all java.nio.DirectByteBuffer and off-heap memory information used by applications. You can use this information to troubleshoot excessive physical memory consumption caused by the off-heap memory.
- Click the **System Properties** tab to view system parameters and environment variables.
- Click the **Thread Information** tab to view thread information such as thread name, heap usage, call stack information, and local variables. You can use this view to analyze problems such as too many threads, deadlocks, and deep call stacks.
- Click the **OQL** tab to view heap information such as all strings greater than 2,000 characters in length.

3.5. API monitoring

The API monitoring feature is used to monitor the details of API calls of an application. This feature allows you to monitor the SQL analysis, NoSQL analysis, exception analysis, error analysis, upstream and downstream services, and API call snapshots.

Procedure

To go to the Interface Invocation page, perform the following steps:

1. Log on to the ARMS console .
2. In the left-side navigation pane, choose **Application Monitoring > Applications**. In the top navigation bar, select a region.
3. On the **Applications** page, click the application that you want to view.
4. In the left-side navigation pane, click **Interface Invocation**.

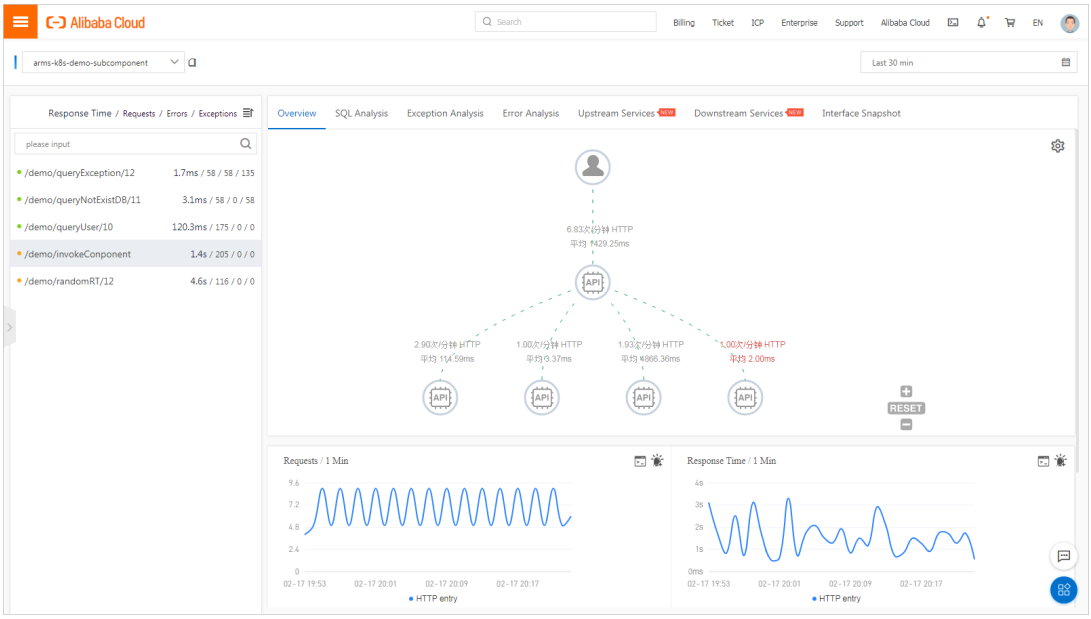
Framework

This feature module can automatically detect and monitor the APIs provided in the following web frameworks and remote procedure call (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+

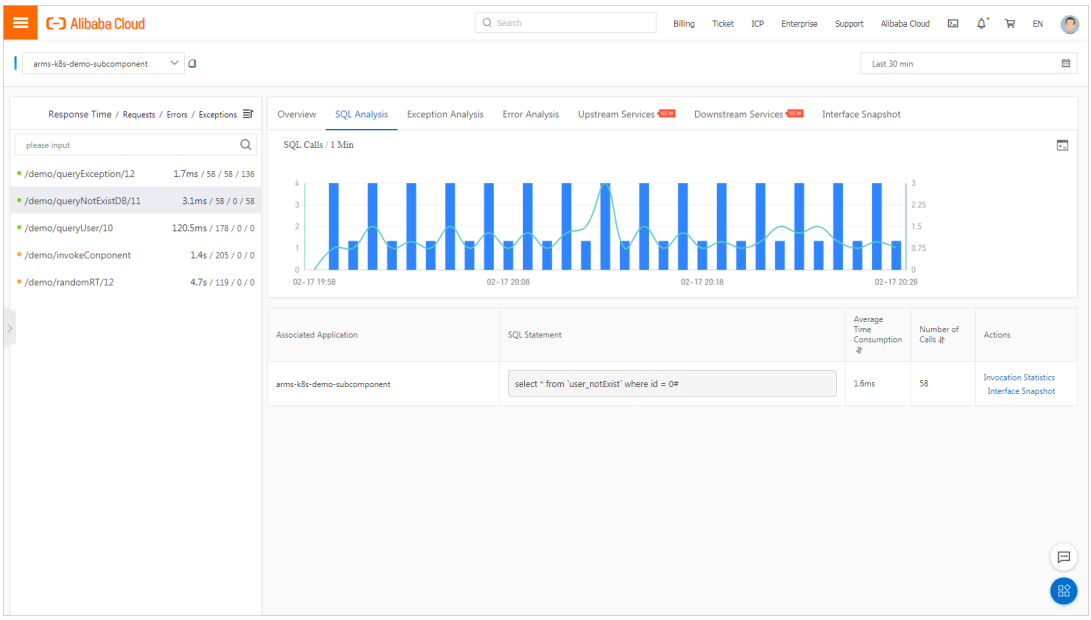
View the details of an API

On the **Overview** tab, you can view the detailed call topology of an API and the time sequence curves of the request count, response time, error count, and HTTP status codes.



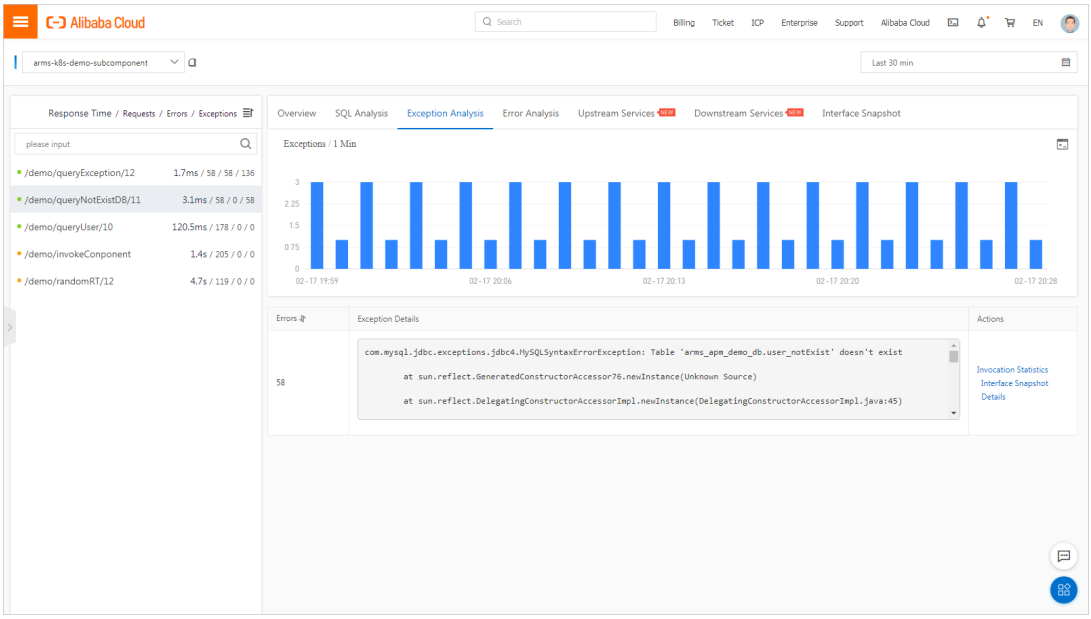
View SQL and NoSQL analysis

On the **SQL Analysis** and **NoSQL Analysis** tabs, you can view the SQL and NoSQL requests that are initiated within the code of the selected APIs in the left-side navigation pane. On this tab, you can find the SQL statements or NoSQL statements that cause slow responses of a service. You can also click **Interface Snapshot** in the Actions column of an SQL or NoSQL statement to view the complete code trace where the SQL or NoSQL execution logic resides.



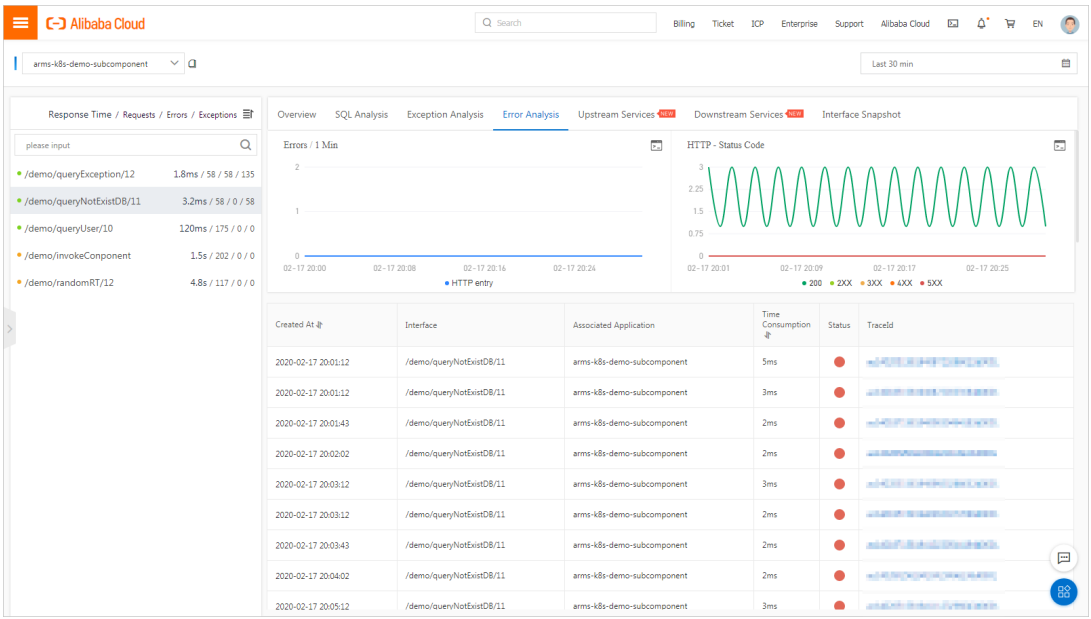
View exception analysis

On the **Exception Analysis** tab, you can view the Java exceptions that are thrown from the code of the selected APIs in the left-side navigation pane. You can also click **Interface Snapshot** in the Actions column of an exception to view the complete trace where the exception stack resides.



View error analysis

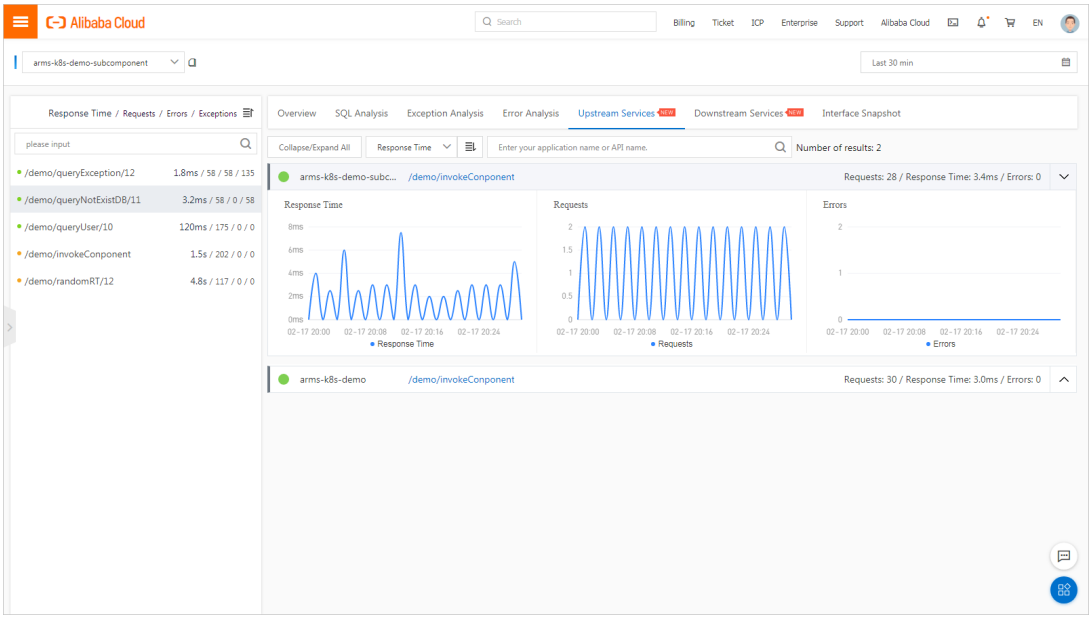
On the **Error Analysis** tab, you can view the errors and HTTP status codes of the application. You can also click a value in the TraceId column to view the trace information on a new page.



View upstream and downstream services

On the **Upstream Services** and **Downstream Services** tabs, you can view the APIs and performance metrics of the upstream services that call the application and downstream services that are called by the application. The performance metrics include the response time, request count, and error count.

Upstream Services tab

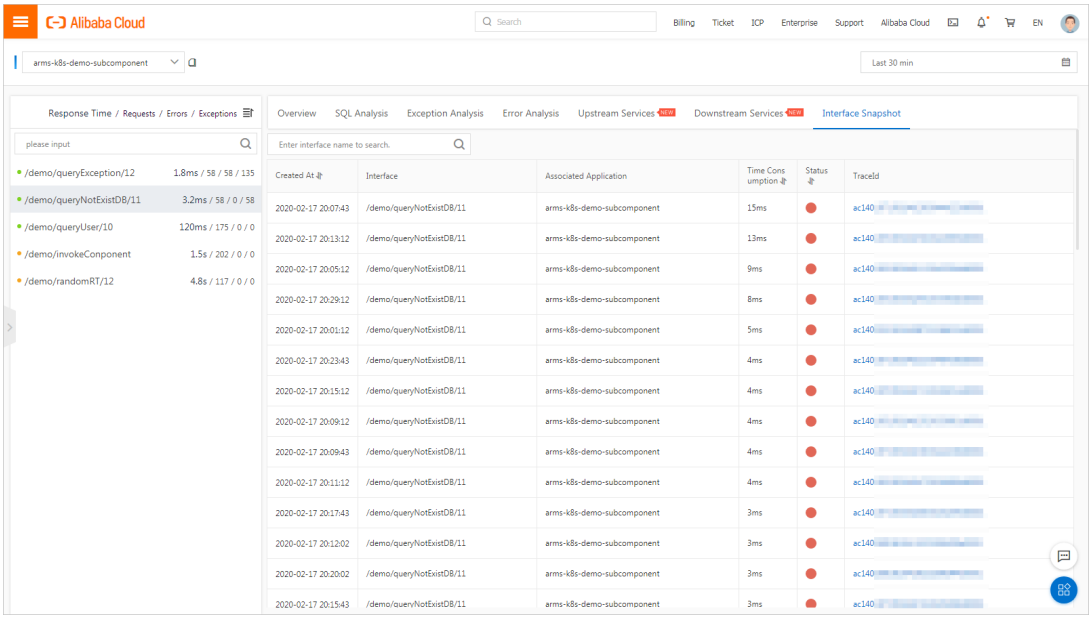


On the **Upstream Services** and **Downstream Services** tabs, you can perform the following operations based on your business requirements:

- On the tabs, click **Collapse/Expand All** to collapse or expand all APIs.
- On the tabs, enter an application name or an API (span) name in the search box, and click the Search icon to search the APIs that meet corresponding conditions.
- Click the collapse panel where the API information resides, or click the up or down arrow at the end of the row. You can then expand or collapse the performance metric information of the API.

View interface snapshots

On the **Interface Snapshot** tab, you can view the parameters of the selected APIs. You can click the trace ID to view the trace.



3.6. View event details in the event center

The event center is the feature module that centralizes, stores, analyzes, and shows the event data generated by some cloud services. The event center can manage the change events of Enterprise Distributed Application Service (EDAS) and alert events of Application Real-Time Monitoring Service (ARMS). The center can also manage 0 to 1 events such as deadlock, out-of-memory (OOM), and application startup, microservice management events of Microservice Engine (MSE), and Kubernetes cluster events. If your application uses one of the related services, the events under this service can be managed by the event center for easy view and analysis.

Event model

An event in the event center is defined by the following five parameters: `source, type, level, time, and data`. Typically, the data parameter is a JSON string. When you search for, demonstrate, and subscribe to events, the five parameters are used.

Some optional parameters such as PID, IP, ClusterId, and PodName can also be associated with an event.

Go to the event center

1. Log on to the ARMS console .
2. In the left-side navigation pane, choose **Application Monitoring > Applications**. In the top navigation bar, select the region where your application is located, and then click the application name.
3. In the left-side navigation pane, click **Event Center**.

Overview

The Event Center page appears in non-application mode or application mode, based on whether an application page is opened. The non-application mode displays all events of the account, whereas the application mode displays only the events that are related to a specified application of the account.

The Event Center page consists of typical events and the **Normal View**, **Topology View**, and **Subscription Rules** tabs.

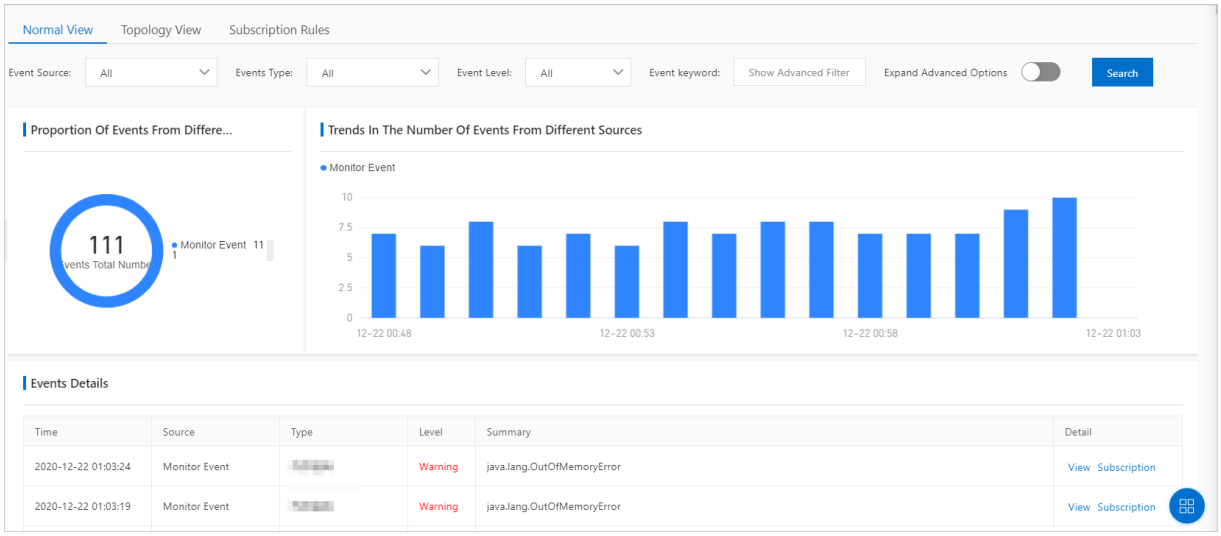
- Typical events: displays the number of typical events that are preset by the system. For more information, see Typical events.
- On the **Normal View** tab, all events associated with the current application are simply analyzed and displayed in multiple dimensions. For more information, see Normal view.
- On the **Topology View** tab, the events associated with the application and the resource topology of the application are displayed together. For more information, see Topology view.
- On the **Subscription Rules** tab, subscription rules that you create are displayed in a list. For more information, see Subscription rules.

Typical events

Typical events: displays the number of times that each type of typical event preset by the system occurred within the last 30 minutes. You can adjust the time range in the upper-right corner. Click **Subscribe** below an event to edit subscription rules of the event. For more information about subscription rules, see Subscription rules.

Normal view

On the **Normal View** tab, you can specify filter conditions to search for events. The search results are displayed in four views: **Heat Map of Events in Last Two Weeks**, **Proportion Of Events From Different Sources**, **Trends In The Number Of Events From Different Sources**, and **Event Details**.



- The **Heat Map of Events in Last Two Weeks** section shows the heat map for hourly occurrences of events that meet the filter conditions in the last two weeks. The darker color indicates more events in the hour.
- The **Proportion Of Events From Different Sources** section displays the proportions of events from different sources.
- The **Trends In The Number Of Events From Different Sources** section displays the trends of events from different sources within a specified period.
- The **Event Details** section displays details about all current events.

Perform the following steps to view information on the **Normal View** tab:

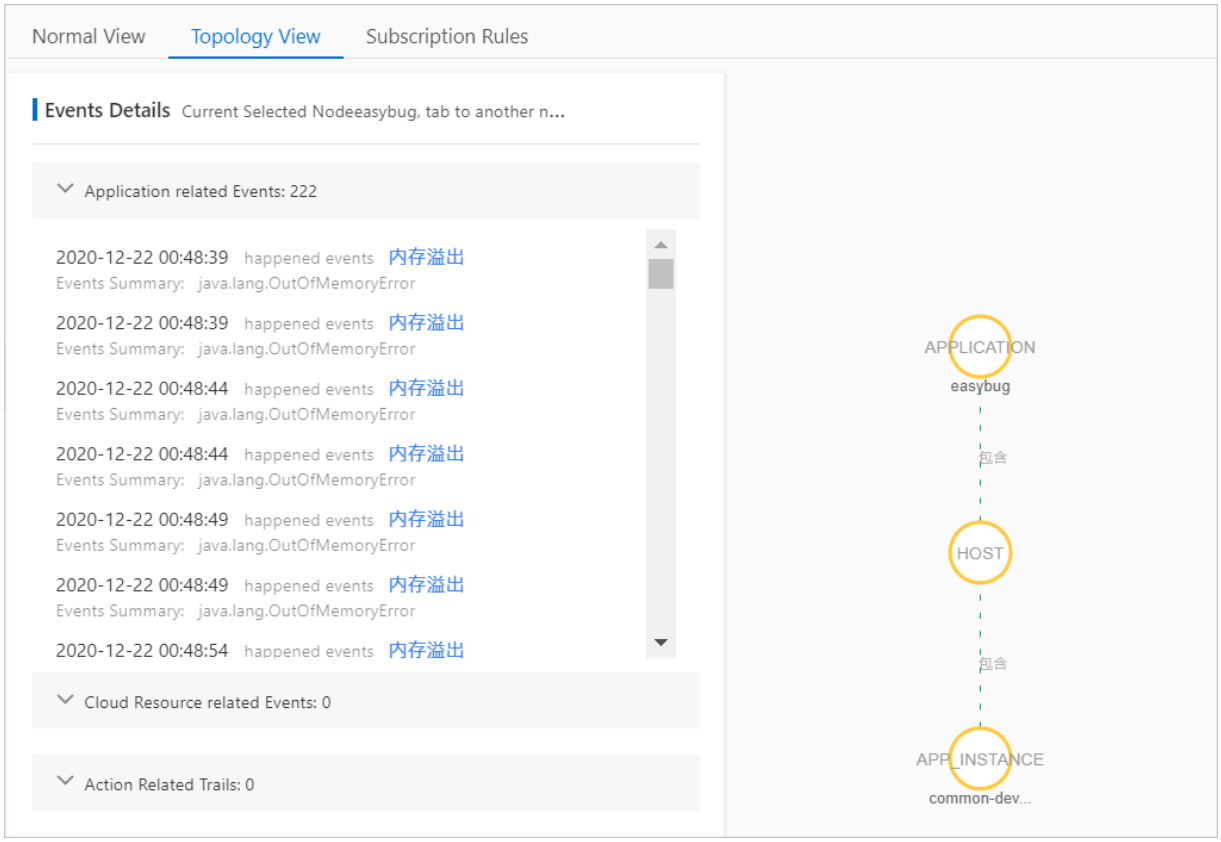
- In the **Heat Map of Events in Last Two Weeks** section, click each heat block to view event details in the hour.
- In the **Trends In The Number Of Events From Different Sources** section, click each column to view all the events that occurred in the corresponding period in the **Event Details** section.
- In the **Event Details** section, click **View** on the right side of an event to view details of the event.
- In the **Event Details** section, click **Subscription** on the right side of an event to subscribe to the event. For more information, see Subscription rules.

Topology view

On the Topology View tab, the resource topology of the application is displayed. The resources include the Elastic Compute Service (ECS) instances used by the application, pods deployed with the application, the ApsaraDB RDS and ApsaraDB for Redis middleware used by the application, and the Server Load Balancer (SLB) and NAT Gateway resources mounted to the application. After the resource topology is displayed, the obtained correlated events, ActionTrail events, and Cloud Monitor events are associated with topology nodes. If you click a node, the events associated with the node are displayed in the **Event Details** section in the upper-left corner.

- **Application-side events**: the events in the event center.
- **Events related to cloud resources**: the events that are related to Cloud Monitor.
- **Action-related audit**: the audit records from ActionTrail.

The topology view can help you troubleshoot the associated resources when an application error occurs. For example, in a large enterprise, an employee mistakenly restarts an ApsaraDB RDS instance in the production environment, which leads to an online service failure. The topology view helps you find out the restart operation on the ApsaraDB RDS instance that is accessed by the application.



Subscription rules

On the **Subscription Rules** tab, all your existing subscription rules are displayed. You can enable, disable, or modify subscription rules on this tab.

> **Notice** You can modify only the rules that you create in the console. You cannot modify the rules that are automatically created by the system.

A subscription rule is used as a basis for you to subscribe to events that meet specified criteria and send the events to a specified webhook URL. You can use one of the following methods to create a subscription rule:

- Method 1: On the **Subscription Rules** tab, click **Create a subscription rule** in the upper-right corner.
- Method 2: On the **Normal View** tab, click **Subscription** in the Operation column of an event in the **Event Details** section.

1. Click the **Subscription Rules** tab. On this tab, click **Create a subscription rule** in the upper-right corner.
2. In the **Create a subscription rule** pane, set **Rule name** and **Rule description**, and then click **Next** in the **Enter basic information** step.
3. In the **Select event mode** step, set the event rule parameters and click **Next**.

Parameter	Description
Event Source	Select an event source from the drop-down list.

Parameter	Description
Event Type	Select an event type from the drop-down list.
Event Level	Select an event level from the drop-down list.
Event Keywords	Enter an event keyword in the field.
Show advanced filtering options	By default, this feature is disabled. After you enable this feature, you can configure the filter conditions such as **Cluster ID**, **Interface/service name**, **Host IP**, and **POD name**.
Custom filter conditions	A custom filter condition is used to specify the condition that must be met by a field in the JSON data of the event body. The root node of a custom filter condition is data. The root node drills down to a field in the JSON data in the format of `.` . Enter custom filter conditions. You can enter up to six filter conditions.
Select a valid field	Enter a valid field in the format of data.x.y. Then, enter an alias for the field. The alias can be used as a placeholder when you enter the POST request body of webhook information. You can set up to six valid fields.
Notification Template	In the Notification Template field, enter the content that will be notified to you when the specified message occurs. If the notification object is a DingTalk chatbot webhook, include the keyword used to create the DingTalk chatbot in the notification template.

4. From the **Select Contact** drop-down list, select a contact. Then, click **Submitted**.If no contact is available in the **Select Contact** drop-down list, click **Create Contact** on the right side to create a contact. After you create it, select it from the Select Contact drop-down list.

1. On the **Normal View** tab, click **Subscription** in the Operation column of an event in the **Event Details** section.If you create a subscription rule by using this method, the source, type, and level filter conditions are automatically selected based on the selected event.
2. Enter a value in the *Value* field in the JSON file of the event. This way, you can specify custom filter conditions and valid fields. A filter condition and a valid field are automatically generated each time you click a field. You can manually modify or delete the fields to adjust the subscription rule. After you select an event mode, click **Next**.
3. From the **Select Contact** drop-down list, select a contact. Then, click **Submitted**.If no contact is available in the **Select Contact** drop-down list, click **Create Contact** on the right side to create a contact. After you create it, select it from the Select Contact drop-down list.

Create a subscription rule

Subscribe to events on the Normal View tab

3.7. Database calls

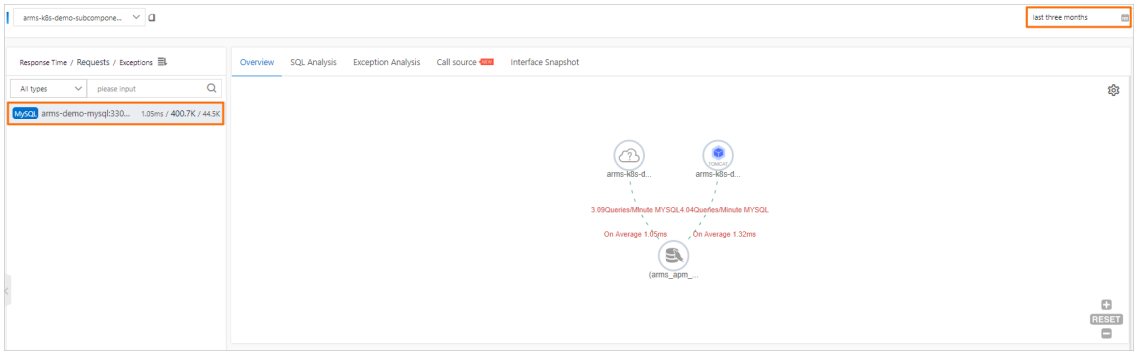
This topic shows you how to view the information about calls to the database of an application. You can obtain an overview of the database calls of the application and view the information about SQL calls, exceptions, call sources, and operation snapshots.

Prerequisites

The Application Real-Time Monitoring Service (ARMS) agent is installed for an application. For more information, see Overview.

Procedure

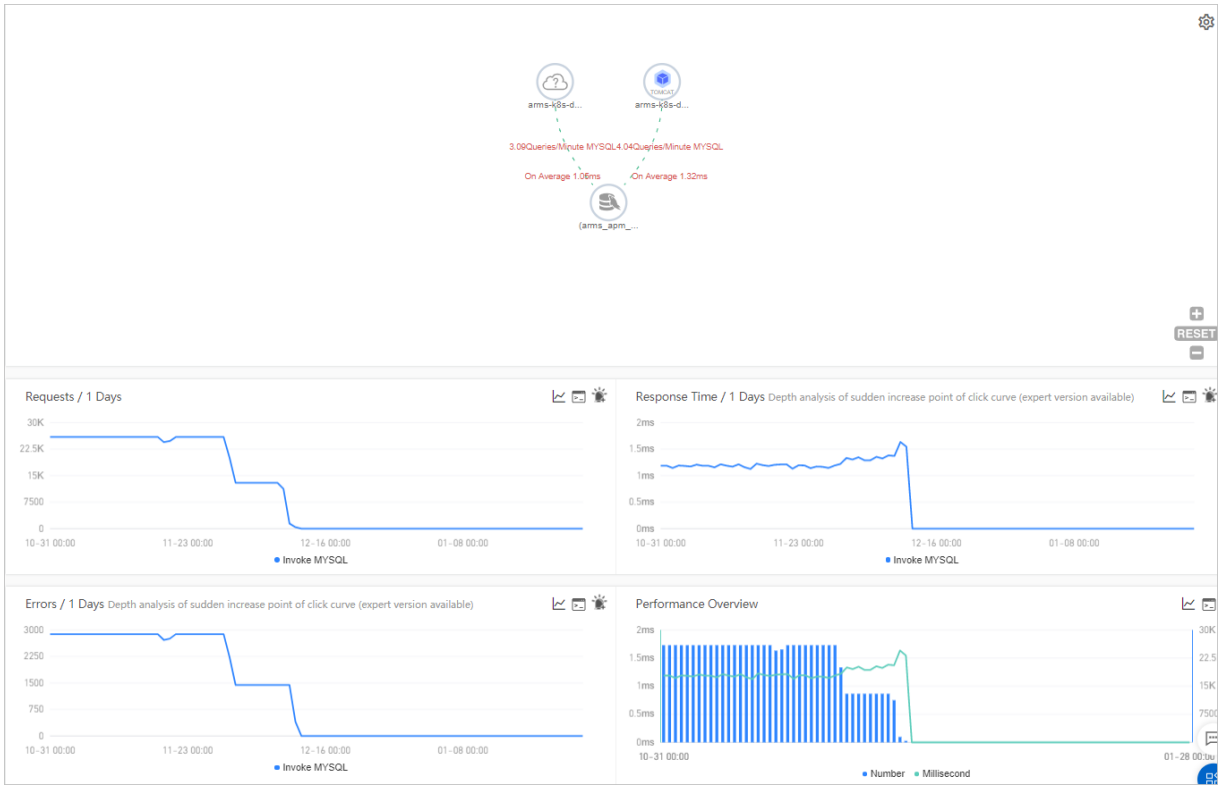
1. Log on to the ARMS console .
2. In the left-side navigation pane, Select **Application monitoring > Applications** .
3. In the top navigation bar of the MNS console, select the region where your cluster is deployed.
4. Log on to the **Applications** Page, click the application name.
5. In the left-side navigation pane, click **Database Invocation**.
6. On the page that appears, select a database and set the time period.



7. After you complete the settings, perform the following operations as required:
   - On the **Overview** tab, obtain an overview of the database calls of the application.
   - Click the **SQL Analysis** tab to view the SQL analysis of the application.
   - Click the **Exception Analysis** tab to view the database call exceptions of the application.
   - Click the **Call source** tab to view the information about the applications that call the database of the application.
   - Click the **Interface Snapshot** tab to view the snapshots of the database operations that are called in the application.

Overview

The **Overview** tab displays the information about the database. You can view the call relationship topology, time series curve of the number of requests, time series curve of the response time, and time series curve of the number of errors.



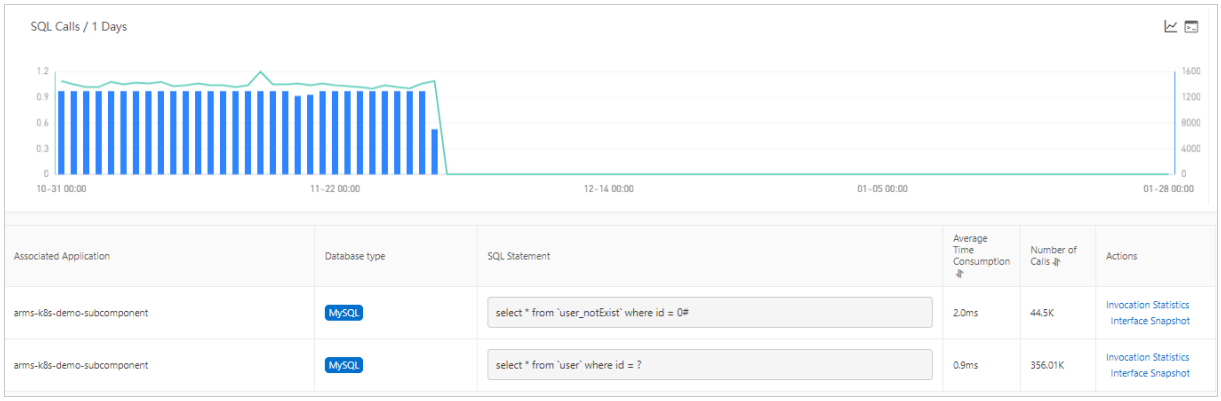
1. (Optional)On the **Overview** tab, perform the following operations as required:
   - Click the icon to configure the display settings of the application topology.

     > **Note** The settings are stored in the browser and remain effective the next time you access the Overview tab.

   - Click the plus sign or scroll the mouse wheel up to zoom in the application topology.
   - Click the minus sign or scroll the mouse wheel down to zoom out the application topology.
   - Click the RESET icon to restore the application topology to the default size.
   - Move the cursor over the statistics chart to view the statistics.
   - Select a period of time to view the statistics for the specified period.
   - Click Icon to view the statistics of the metric in a certain time period or compare the statistics of the same time period on different dates.
   - Click Icon to view the API details for this metric.
   - Click Icon to create an alarm for the metric. For more information, see Create an alert.

SQL analysis

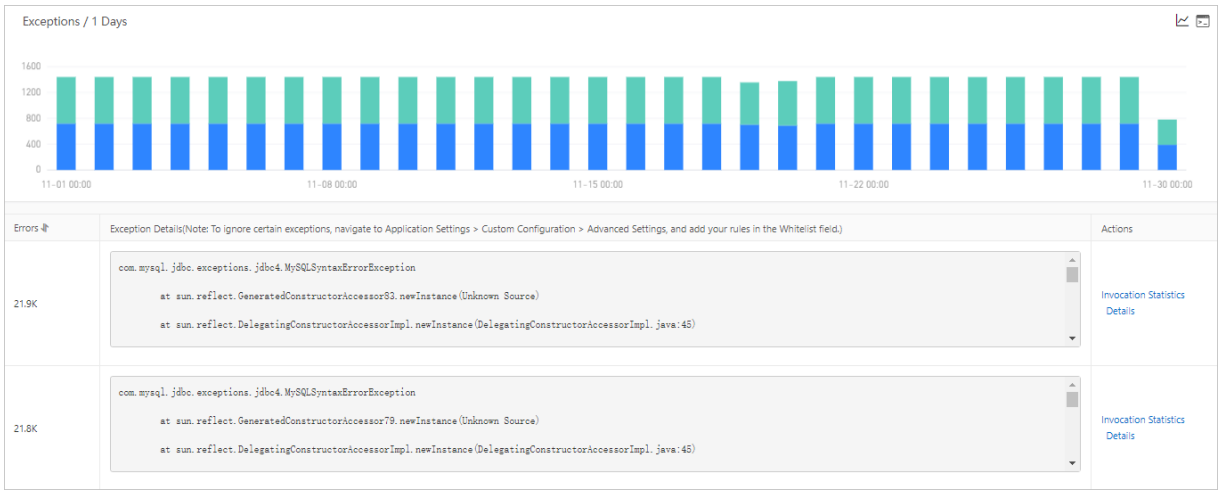
The **SQL Analysis** tab displays the column chart of the number of SQL calls, the time series curves of the response time, and a list of SQL statements that are executed in the database.



1. (Optional)On the **SQL Analysis** tab, perform the following operations as required:
   - Move the cursor over the statistics chart to view the statistics.
   - Select a period of time to view the statistics for the specified period.
   - Click Icon to view the statistics of the metric in a certain time period or compare the statistics of the same time period on different dates.
   - Click Icon to view the API details for this metric.
   - To view the SQL call statistics of an SQL statement, click **Invocation Statistics** in the **Actions** column of the SQL statement.
   - To view the snapshots of the operation that is called by an SQL statement, click **Interface Snapshot** in the **Actions** column of the SQL statement.

Exception analysis

The **Exception Analysis** tab displays the information about exceptions of the database.



1. (Optional)On the **Exception Analysis** tab, perform the following operations as required:

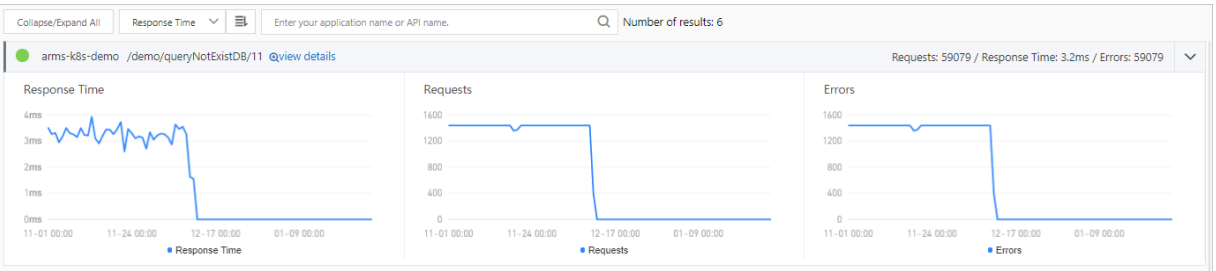
> **Note** To filter exceptions, perform the following steps: In the left-side navigation pane, click **Application Settings**. On the page that appears, click the **Custom Configuration** tab. In the **Advanced Settings** section, set the **Whitelist** field.

   - Move the cursor over the statistics chart to view the statistics.
   - Select a period of time to view the statistics for the specified period.

- Click Icon to view the statistics of the metric in a certain time period or compare the statistics of the same time period on different dates.
- Click Icon to view the API details for this metric.
- To view the statistics of an exception, click **Invocation Statistics** in the **Actions** column of the exception.
- To view the details of an exception, click **Details** in the **Actions** column of the exception.

Call source

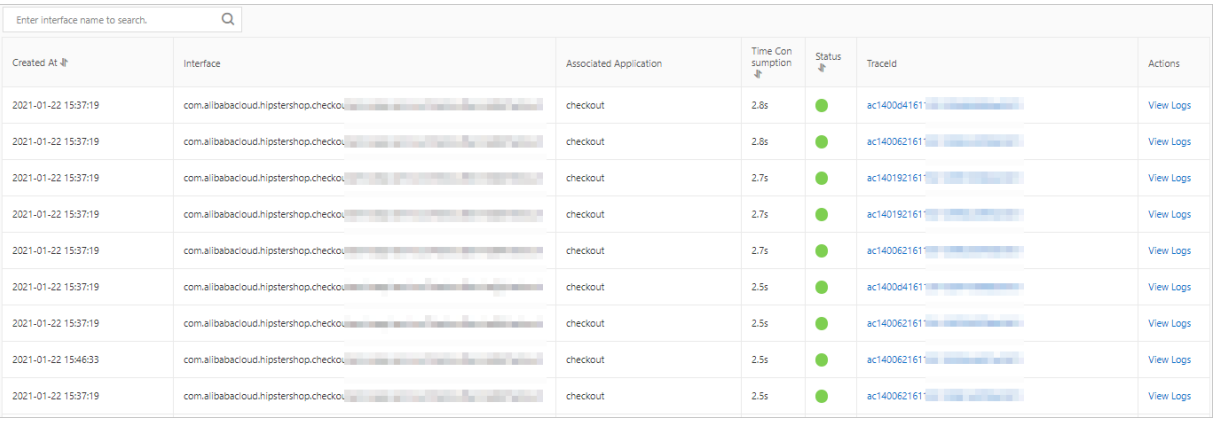
The **Call source** tab displays the information about the call sources of the database.



1. (Optional)On the **Call source** tab, perform the following operations as required:
   - To view the information about an application that calls the database or information about a database operation that is called, enter the application or operation name in the search box and click the icon.
   - To view the snapshots of a database operation that is called by a call source, click **view details** next to the operation.
   - Move the cursor over the statistics chart to view the statistics.
   - Click Icon to view the statistics of the metric in a certain time period or compare the statistics of the same time period on different dates.

Operation snapshot

The **Interface Snapshot** tab displays the snapshots of all operations that are called in the database.



1. (Optional)On the **Interface Snapshot** tab, perform the following operations as required:
   - To view the snapshots of an operation, enter the operation name in the search box and click the icon.

- To view a trace of an operation, click the trace ID in the **TraceId** column of the operation.
- To view the logs of an operation, click **View Logs** in the **Actions column** of the operation.

3.8. External call

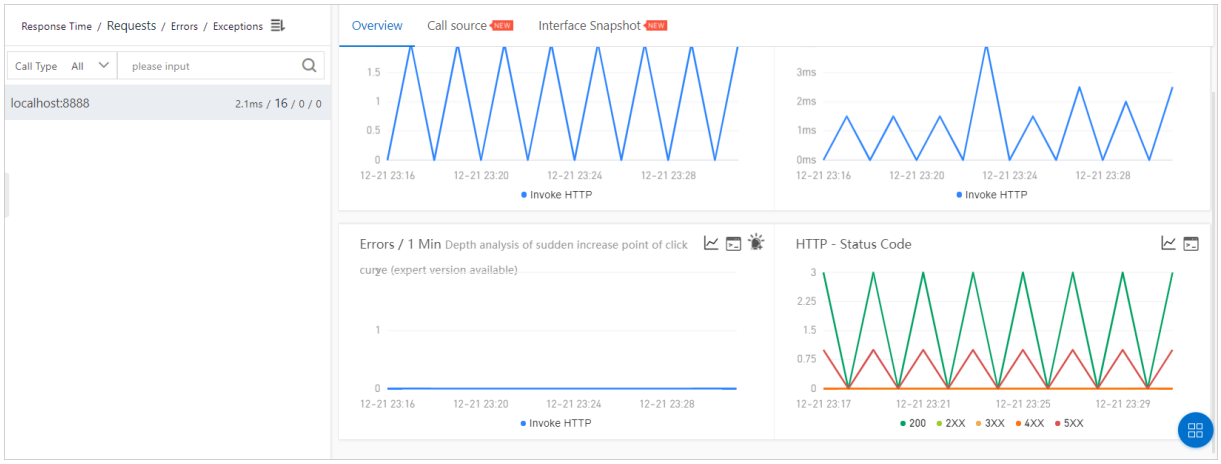
You can use the external call feature of Application Real-Time Monitoring Service (ARMS) application monitoring to locate slow calls or errors during the external calls of an application.

Procedure

1. Log on to the ARMS console .
2. In the left-side navigation pane, choose **Application Monitoring > Applications**. In the top navigation bar, select a region.
3. On the **Applications** page, click the name of the application you want to monitor.
4. In the left-side navigation pane, click **External Calls**.All external calls of the application are listed on the left of the **External Calls** page. You can sort the calls by response time, request count, error count, or exception count.

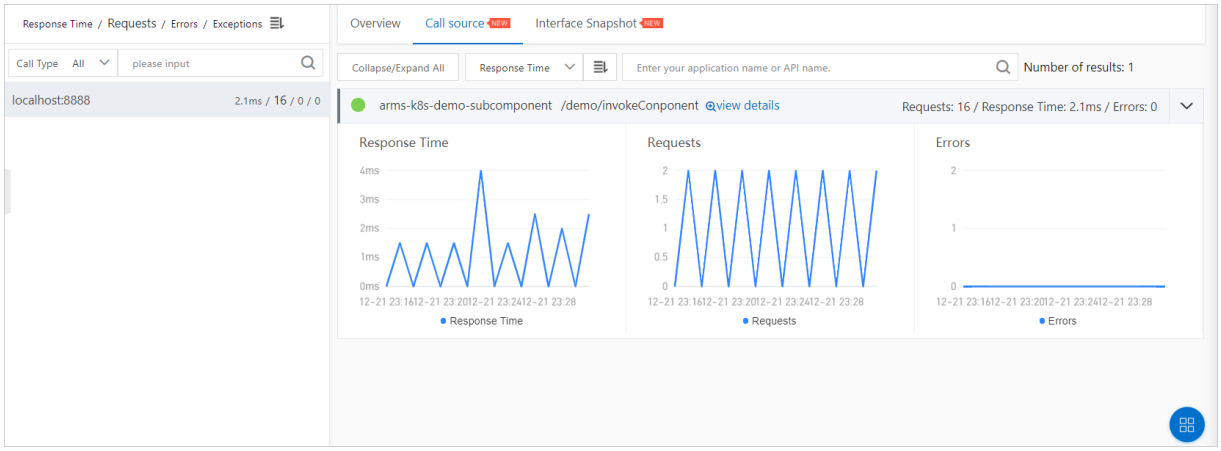
Overview

You can click an external call in the left-side call list to view the request count, response time, error count, and HTTP status code of the external call on the **Overview** tab.



Call source

You can click an external call in the left-side call list to view the request count, response time, and error count of all interfaces related to the external call on the **Call Source** tab.



You can perform the following operations on the **Call Source** tab:

- Click **Collapse/Expand All** to collapse or expand all interfaces.
- Enter the keyword of an application name or an interface (span) name in the search box on the top of the tab, and then click the search icon to filter out the interfaces that meet the conditions specified by the keyword.
- Click the collapse panel where the interface information is located, or click the up or down arrow at the end of the row to expand or collapse the performance metric information of this interface.

Interface snapshot

You can click an external call in the left-side call list to view the parameters of the call on the **Interface Snapshot** tab.

3.9. MQ monitoring

The MQ monitoring page in application monitoring of Application Real-Time Monitoring Service (ARMS) shows the message publishing and topic subscription in Message Queue for Apache RocketMQ.

Procedure

1. Log on to the ARMS console .
2. In the left-side navigation pane, choose **Application Monitoring > Applications**. In the top navigation bar, select a region.
3. On the **Applications** page, click the name of the application.
4. In the left-side navigation pane, click **MQ Monitoring**.
5. Click a search result link on the right of the page.

Features

- On the **Overview** tab, the message publishing and subscription relationship between your application and MQ data source is displayed in the topological graph.
- On the **Publishing Statistics** tab, the statistics of message publishing are displayed. The statistics include the request count, response time, and error count.
- On the **Subscription Statistics** tab, the statistics of subscription are displayed. The statistics include the request count, response time, and error count.
- On the **Interface Snapshot** tab, the interface snapshots of message publishing and subscription are provided. You can view the complete trace by using the trace ID and diagnose the issues.

3.10. Application diagnosis

3.10.1. Real-time diagnostics

The real-time diagnostics feature is suitable for scenarios where you want to monitor application performance and identify the cause of problems within a short period of time. This topic describes how to use the real-time diagnostics feature.

Context

If you want to monitor the performance of an application for a short period of time, such as releasing an application or performing stress tests on an application, you can use the real-time diagnostics feature. After the real-time diagnostics feature is enabled for an application, ARMS continuously monitors the application for 5 minutes and reports all the data of the traces during this period. Then, you can use the method stack waterfall chart and thread profiling to identify the causes of exceptions based on the trace that shows performance problems.

Procedure

1. Log on to the ARMS console .
2. In the left-side navigation pane, choose **Application Monitoring > Applications**. In the top navigation bar, select a region.
3. On the **Applications** page, click the name of the application.
4. In the left-side navigation pane, choose **Application Diagnosis > Real-time Diagnosis**.

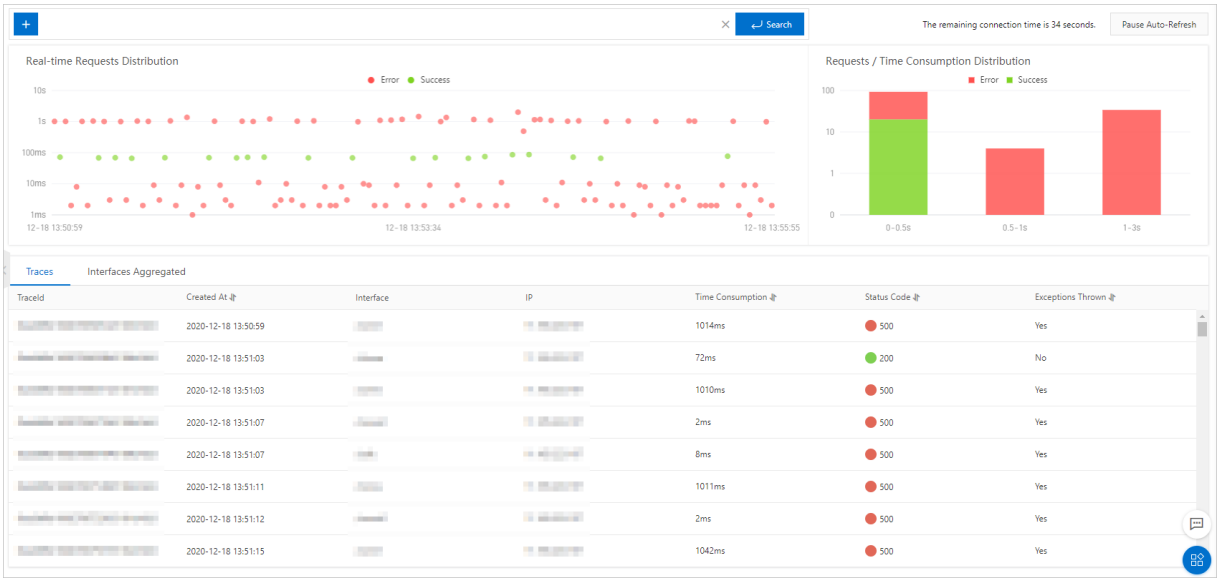
Enable and disable real-time diagnostics

The first time you access the **Real-time Diagnosis** page, real-time diagnostics is automatically enabled. To enable real-time diagnostics in other cases, click **Enable real-time diagnosis** in the upper-right corner.

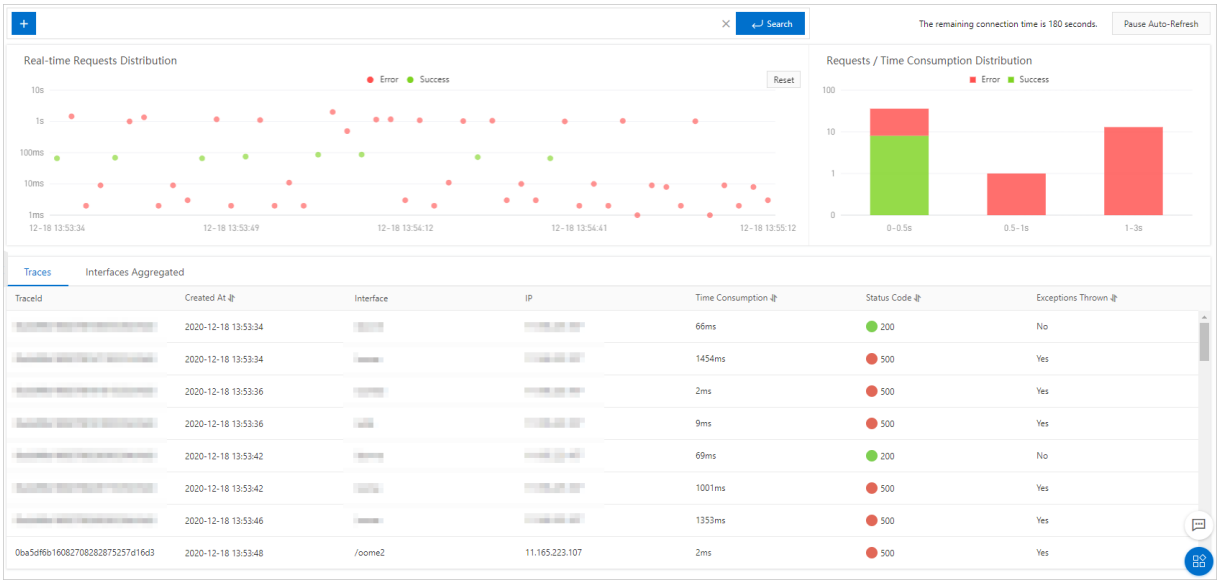
Real-time diagnostics is automatically enabled for 5 minutes and then disabled. To disable real-time diagnostics, click **Terminate Real-time Diagnosis** in the upper-right corner.

View real-time monitoring data

In the **Real-time Requests Distribution** and **Requests by Response Time** sections, you can view the statistics of the last 1,000 requests captured as of the current point in time.



In the chart of the **Real-time Requests Distribution** section, select a time range. Data of the selected time range can be set as visible. The chart shows data only within this time range. Click **Reset** in the upper-right corner of the chart and the default view can be restored.



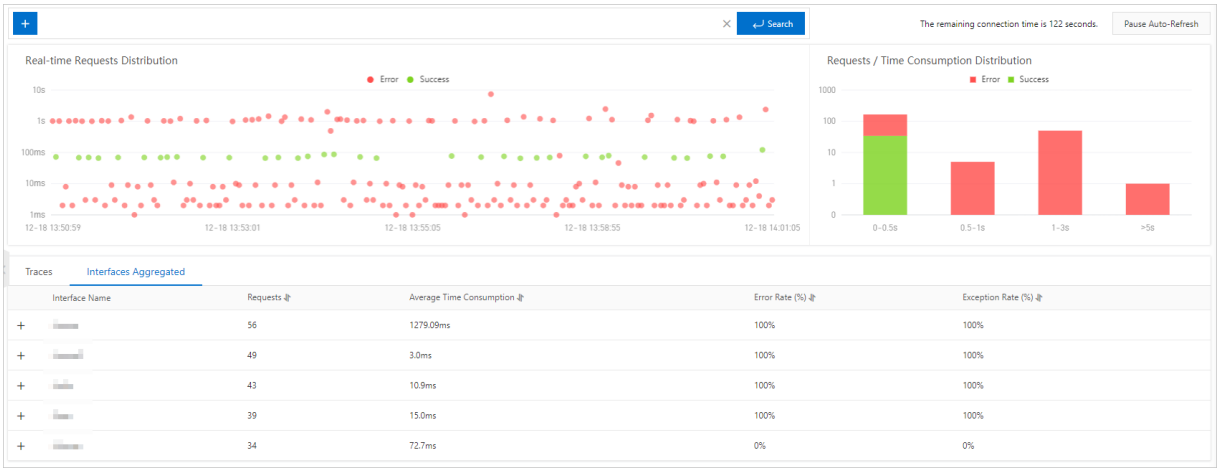
Filter monitoring data

You can filter request monitoring data displayed on the page by operation name or IP address.

1. Click the **+** icon above the **Real-time Requests Distribution** section.
2. Select an API operation or IP address from the drop-down list and click **Search**.
   Only the request monitoring data of the selected operation is displayed on the page.

View information of traces

On the **Traces** and **Interfaces Aggregated** tabs, you can view information of all traces captured in the corresponding period. Click a trace ID to access the **Link Invocation** page. Use the local method stack waterfall chart and thread profiling to identify the causes of exceptions.



Related information

- Trace query
- Analyze errors in code by using ARMS thread profiling

3.10.2. Thread profiling

The thread profiling feature provides statistics on the CPU time consumption at the thread level and the number of threads per type. ARMS records and aggregates the method stacks of threads every 5 minutes, which helps you review the code execution process and find out thread problems. When the CPU utilization of a cluster is high or a large number of slow methods are detected, the thread profiling feature can be used to find out the thread or method that consumes the most CPU resources.

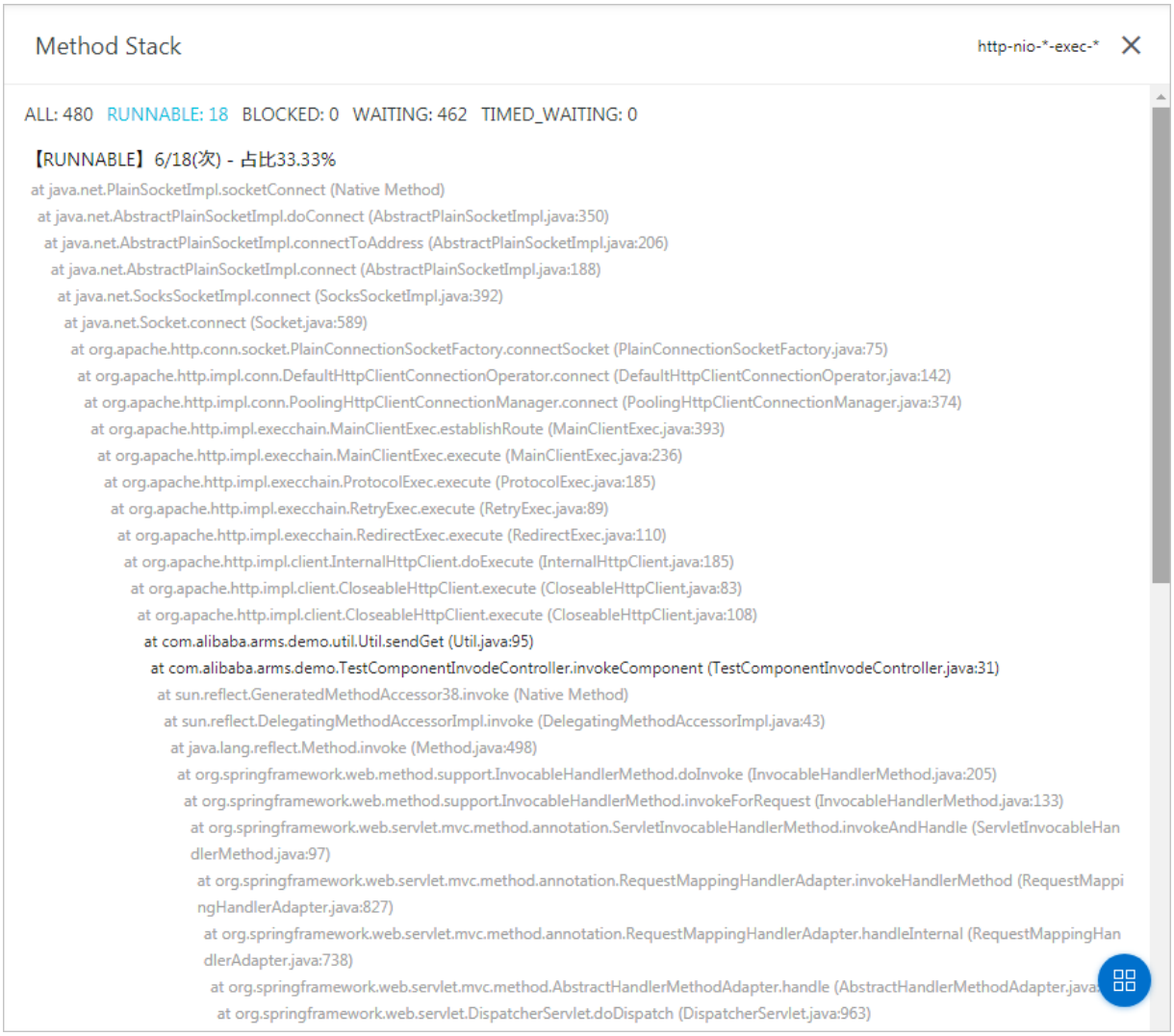
Procedure

1. Log on to the ARMS console .
2. In the left-side navigation pane, choose **Application Monitoring > Applications**. In the top navigation bar, select a region.
3. On the **Applications** page, click the name of the application.
4. In the left-side navigation pane, choose **Application Diagnosis > Threads Profiling**.

Perform thread profiling

On the **Threads Profiling** page, all threads of the application are listed on the left. You can detect abnormal threads based on statistics in the **CPU Time Consumption (ms)** section. Select an abnormal thread and analyze the changes of the CPU time consumption and thread count based on the graphs in the **CPU Time Consumption (ms)** and **Thread Count** sections. For example, you can analyze whether the total number of threads per minute is overlarge.

You can also click **Method Stack** to view the method stack that is actually running within a specified period of time. For example, you can view the method stack of the threads in the BLOCKED state and optimize the specified code block to reduce CPU utilization.



> **Note** If no data appears after you click **Method Stack**, click **Application Settings** in the left-side navigation pane. On the page that appears, click the **Custom Configuration** tab and check whether **Thread Profiling Method Stack** in the **Thread settings** section is turned on. If Thread Profiling Method Stack is turned off, the method stack information cannot be recorded. If Thread Profiling Method Stack is turned on, the method stack information is collected every 5 minutes.

Related information

- Analyze errors in code by using ARMS thread profiling

3.11. Application Settings

3.11.1. Custom configuration

Some common settings of application monitoring, such as the sampling rate of traces, agent switch, and slow SQL threshold, can be directly configured on the **Custom Configuration** tab.

Prerequisites

Create an application monitoring job

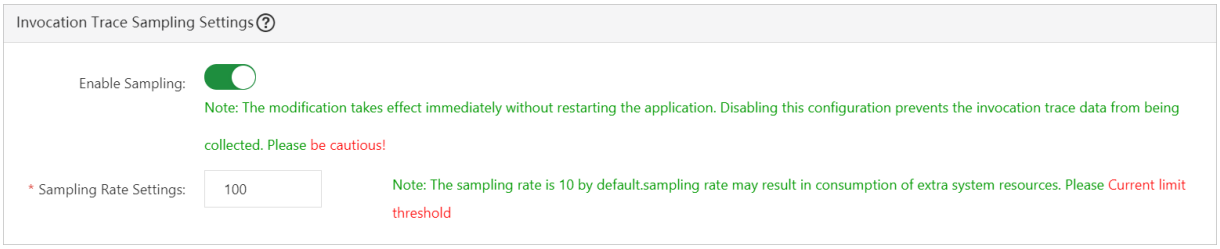
Procedure

1. Log on to the ARMS console .
2. In the left-side navigation pane, choose **Application Monitoring > Applications** and select a region in the top navigation bar.
3. On the **Applications** page, click the name of the application.
4. In the left-side navigation pane, click **Application Settings**. On the page that appears, click the **Custom Configuration** tab.
5. Configure the custom parameters and click **Save** in the lower part of the page.

Configure trace sampling settings

In the **Invocation Trace Sampling Settings** section, you can turn on or off the sampling, and set the sampling rate. You need to enter only the number of the percentile in the **Sampling Rate Settings** field. For example, if you enter *10*, the sampling rate is 10%.

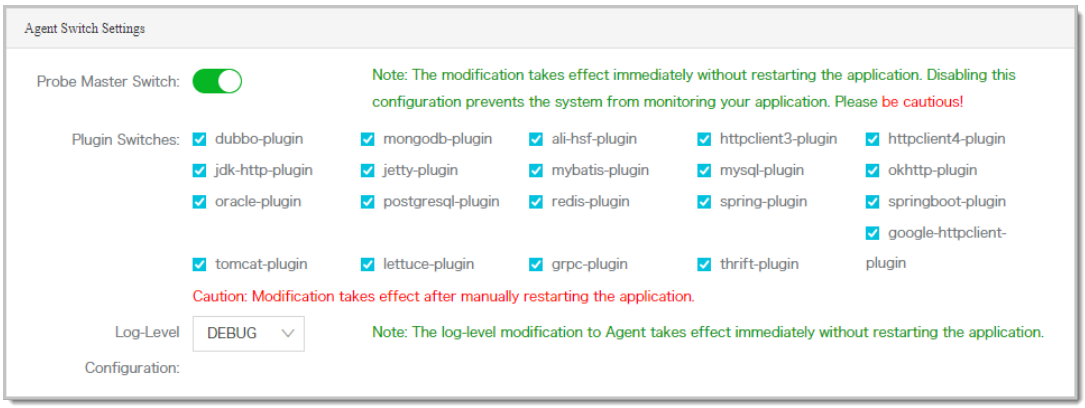
> **Notice** The modification takes effect immediately. You do not need to restart the application. If sampling is turned off, the trace data is not captured. Proceed with caution.



Configure the agent switch and log level

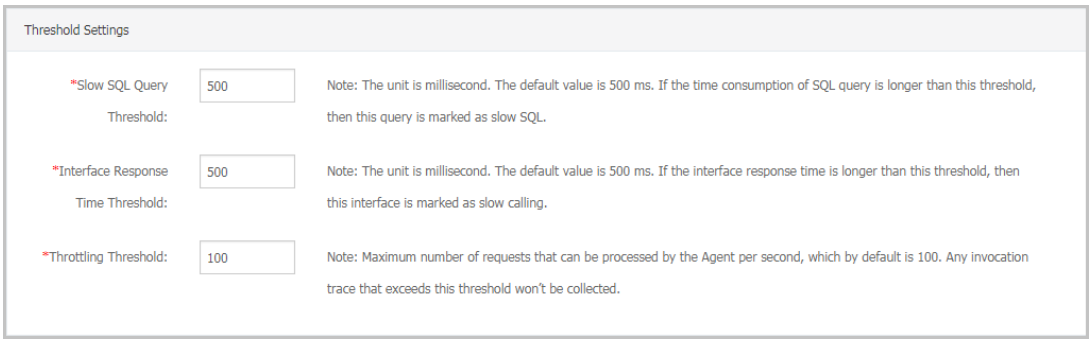
In the **Agent Switch Settings** section, you can turn on or off the probe master switch and other plug-in switches, and configure the log level.

> **Notice** The modifications to the master switch of the agent and log level take effect immediately and do not require you to restart the application. If the master switch of the agent is turned off, Application Real-Time Monitoring Service (ARMS) cannot monitor your applications. Proceed with caution. To make changes to each plug-in switch take effect, you must manually restart the application.



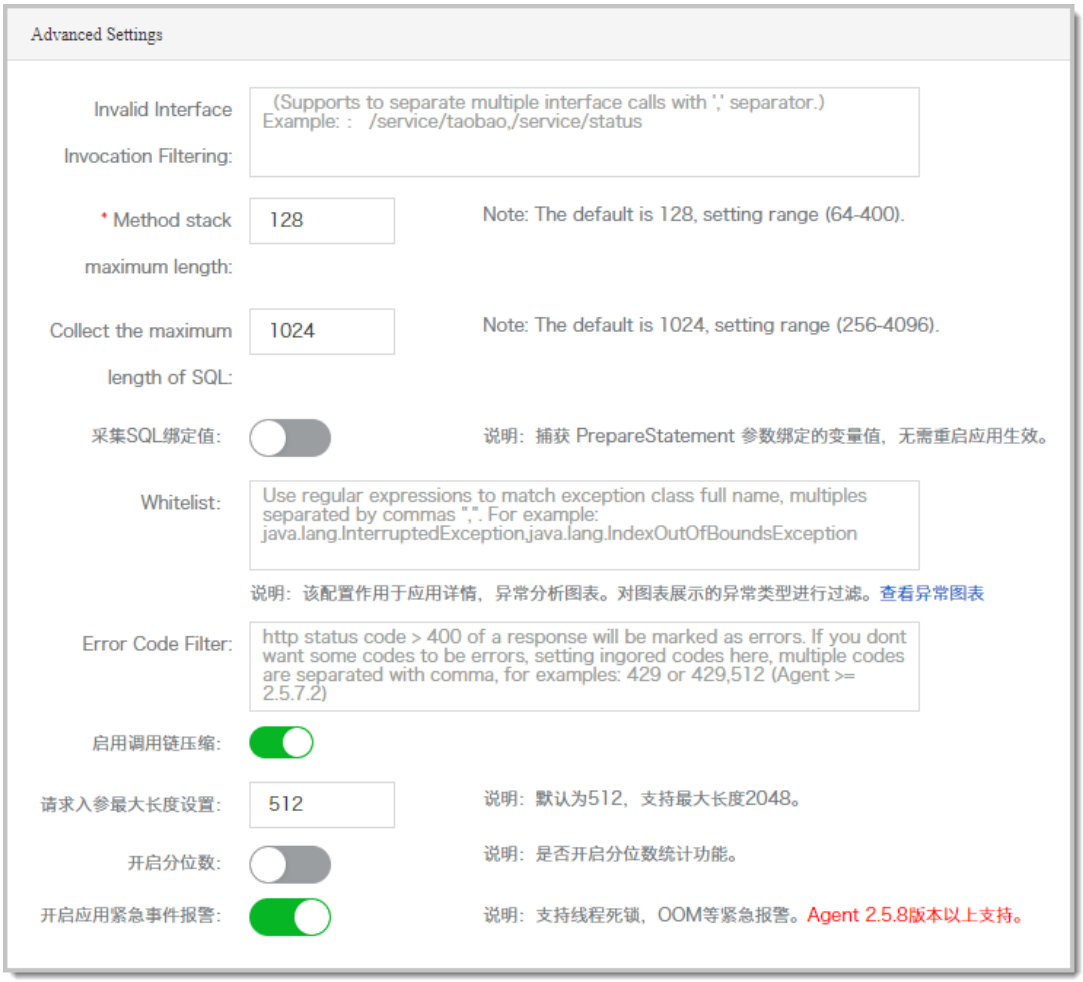
Configure threshold settings

In the **Threshold Settings** section, you can set the slow SQL query threshold, interface response time threshold, and throttling threshold.



Configure advanced settings

In the **Advanced Settings** section, you can set the interface to be filtered and the maximum length of the method stack.

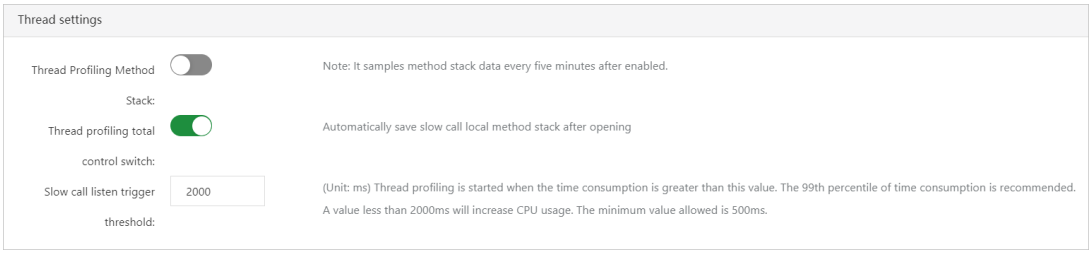


- **Invalid Interface Invocation Filtering**: Enter an interface whose call status does not need review. Then, this interface is hidden from the **Interface Invocation** page.
- **Method stack maximum length**: Default value: 128. Maximum value: 400. Unit: entry.
- **Stack Depth to Distinguish Exceptions**: The stack depth that is used to distinguish exceptions of a same type. This parameter is typically set to the call depth of the first difference.
- **Collect the maximum length of SQL**: Default value: 1024. Minimum value: 256. Maximum value: 4096. Unit: character.
- **Capture SQL Bound Variables**: Specifies whether to capture the variable value bound with PrepareStatement. This parameter takes effect without restarting the application.
- **Original SQL**: SQL statements are only truncated.
- **Exception Filtering**: The exception that you enter is not displayed in the chart on the Application Details and Exception Analysis tabs.
- **Errors Filtering**: By default, HTTP status codes greater than 400 are counted as errors. You can specify the error codes that are greater than 400 but that you do not want to be counted as errors.
- **New Trace Format**: Specifies whether to use a new storage format that sort traces by time. This parameter is enabled by default.
- **Trace Compression**: Specifies whether to simplify duplicated calls such as for loops. This parameter takes effect without restarting the application.
- **Maximum Request Parameter Length**: Default value: 512. Maximum value: 2048. Unit: character.
- **Show Percentiles**: Specifies whether to turn on quantile statistics.

- **Enable application emergency alert**: Specifies whether to enable alerts for emergencies such as thread deadlocks and out-of-memory (OOM). The agent version must be 2.5.8 or later.
- **RabbitMQ Consumer**: Specify the class name of a consumer or the name of the class that contains an anonymous internal consumer. You can then view the trace of the customer. Separate multiple names with commas (,).

Configure thread settings

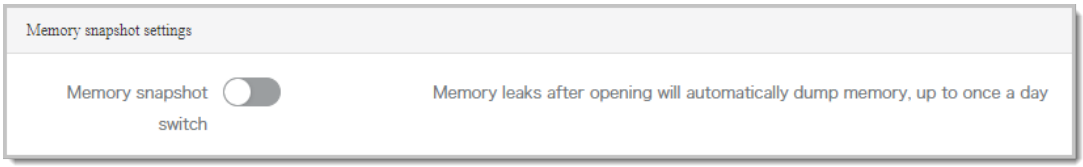
In the **Thread Settings** section, you can enable or disable thread diagnosis method stack and thread profiling master. You can also set the trigger threshold of the slow call listener.



Note The listener is started only when the service call response time exceeds the threshold (1,000 ms by default) and lasts until the call ends or the consumed time exceeds 15 seconds. We recommend that you set the threshold to the 99th percentile of the call response time. For example, if 100 calls are listed in ascending order by response time, the time consumed by the 99th one is the 99th percentile.

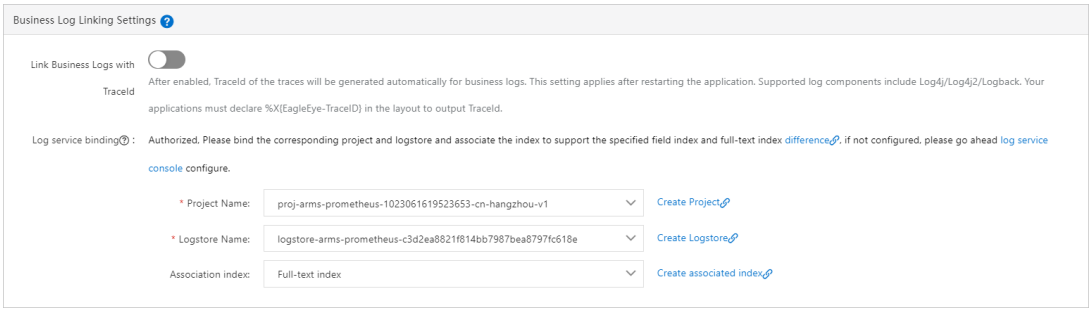
Configure memory snapshot settings

In the **Memory Snapshot Settings** section, you can enable or disable memory snapshots. If you enable it, a memory dump (at most one time a day) is created when memory leaks occur.



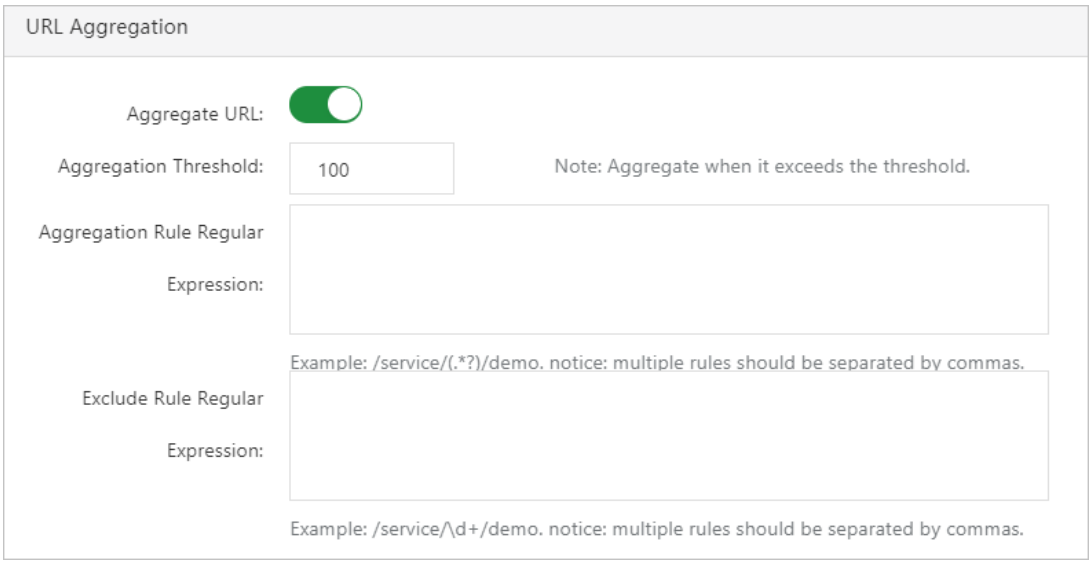
Associate a business log with the trace ID

In the **Business Log Association Settings** section, you can set whether to associate the business log of an application with the trace ID. For more information, see Associate trace IDs with business logs.



Configure URL convergence rules

In the **URL Convergence Settings** section, you can enable or disable the convergence feature. You can also set the convergence threshold, convergence rules and troubleshooting rules. URL convergence means that similar URLs are displayed together as a single object. For example, URLs prefixed with /service/demo are displayed as an object. The convergence threshold is the minimum number of URLs to trigger URL convergence. For example, when the threshold is 100, URLs converge only when 100 URLs meet the regular expression of the rules.



Business monitoring settings

In the **Business Monitoring Settings** section, you can enable or disable business monitoring and configure HTTP encoding.

Related information

- Trace query
- API monitoring
- Analyze errors in code by using ARMS thread profiling
- Memory snapshot

3.11.2. Add custom methods for monitoring

To monitor any methods or APIs that are not automatically detected by the Application Real-Time Monitoring Service (ARMS) agent, you can add custom methods for monitoring in ARMS Application Monitoring.

Prerequisites

Create an application monitoring job

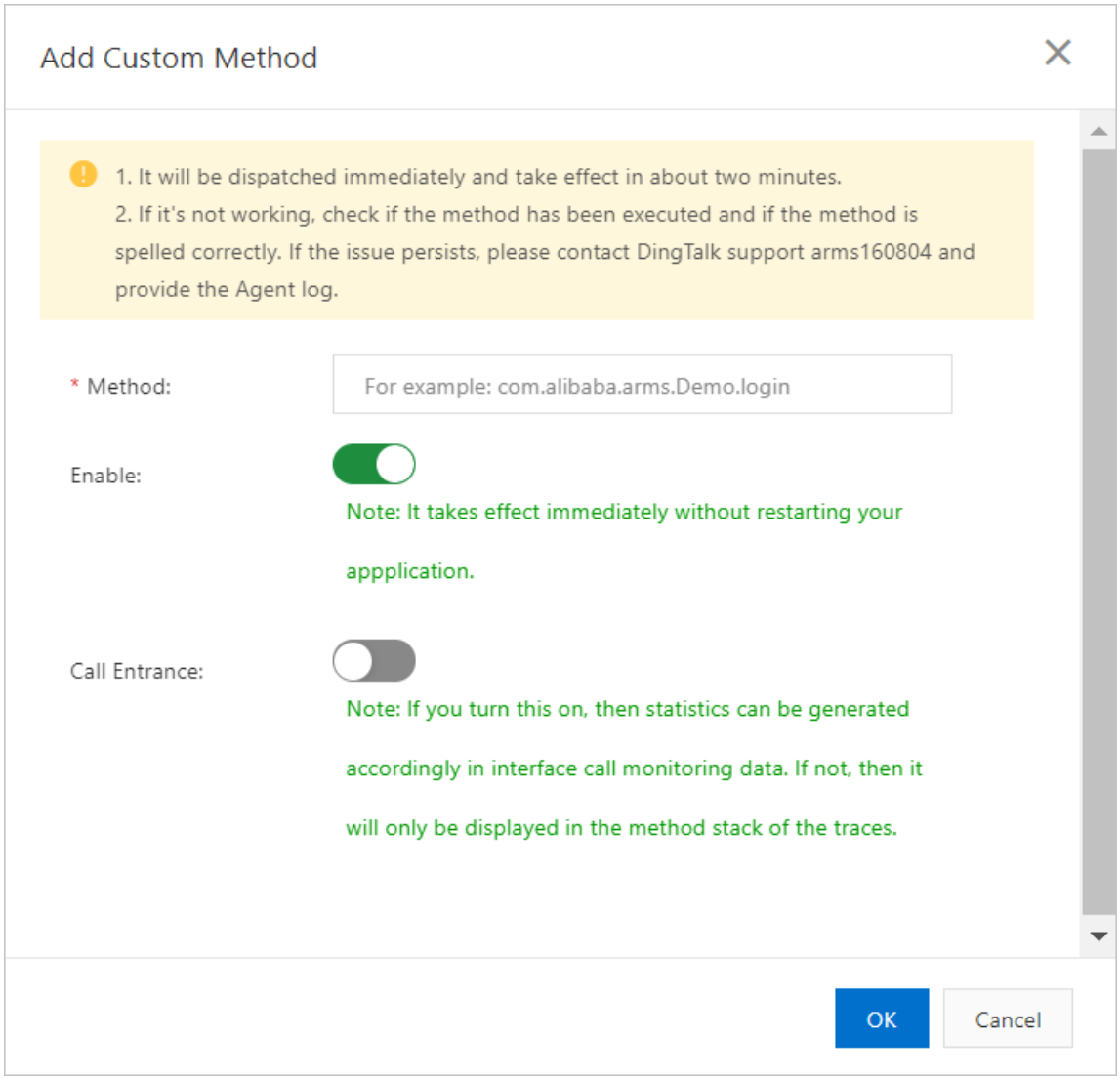
Portal

1. Log on to the ARMS console .
2. In the left-side navigation pane, choose **Application Monitoring > Applications**.
3. On the **Applications** page, click the name of the target application.
4. In the left-side navigation pane, click **Application Settings**. On the page that appears, click the **Monitoring Method Customization** tab.

Add custom methods for monitoring

1. On the **Monitoring Method Customization** tab, click **Add Method** in the upper-right corner.
2. In the **Add Custom Method** dialog box, create a custom method and click **OK**.

Parameter	Description
Method	The name of the method to be monitored. It must be unique.
Enable	After you enable this feature, you can monitor this method and the method is displayed in the local method stack. For more information, see Related operations. By default, the feature is enabled. **Note** ARMS can dynamically enable or disable this feature, without restarting the application.
Call Entrance	After this feature is enabled, you can query businesses based on traces, and the corresponding APIs are displayed in the **API call** module. For more information, see API monitoring. By default, this feature is disabled.



After a custom method is added for monitoring, it is automatically displayed in the method list.

Related information

- Trace query
- API monitoring

3.11.3. Delete an application

When you no longer use Application Real-Time Monitoring Service (ARMS) to monitor your application and want to delete the application from ARMS, you can delete it on the Application Settings page.

Procedure

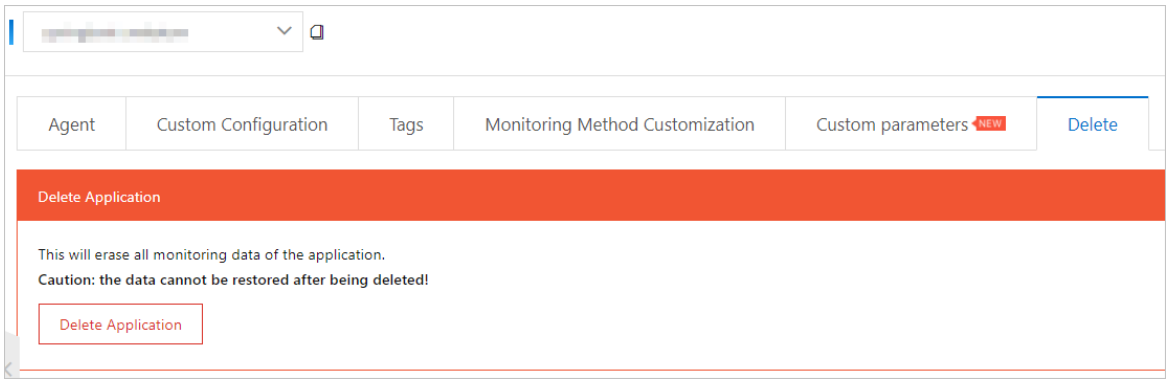
1. Log on to the ARMS console .
2. In the left-side navigation pane, choose **Application Monitoring > Applications**. In the upper part of the Applications page, select a region.
3. On the **Applications** page, click the name of the required application.
4. In the left-side navigation pane, click **Application Settings**. On the page that appears, click the **Custom Configuration** tab.

5. On the **Custom Configuration** tab, turn off **Probe Master Switch** in the **Agent Switch Settings** section. Click **Save**.

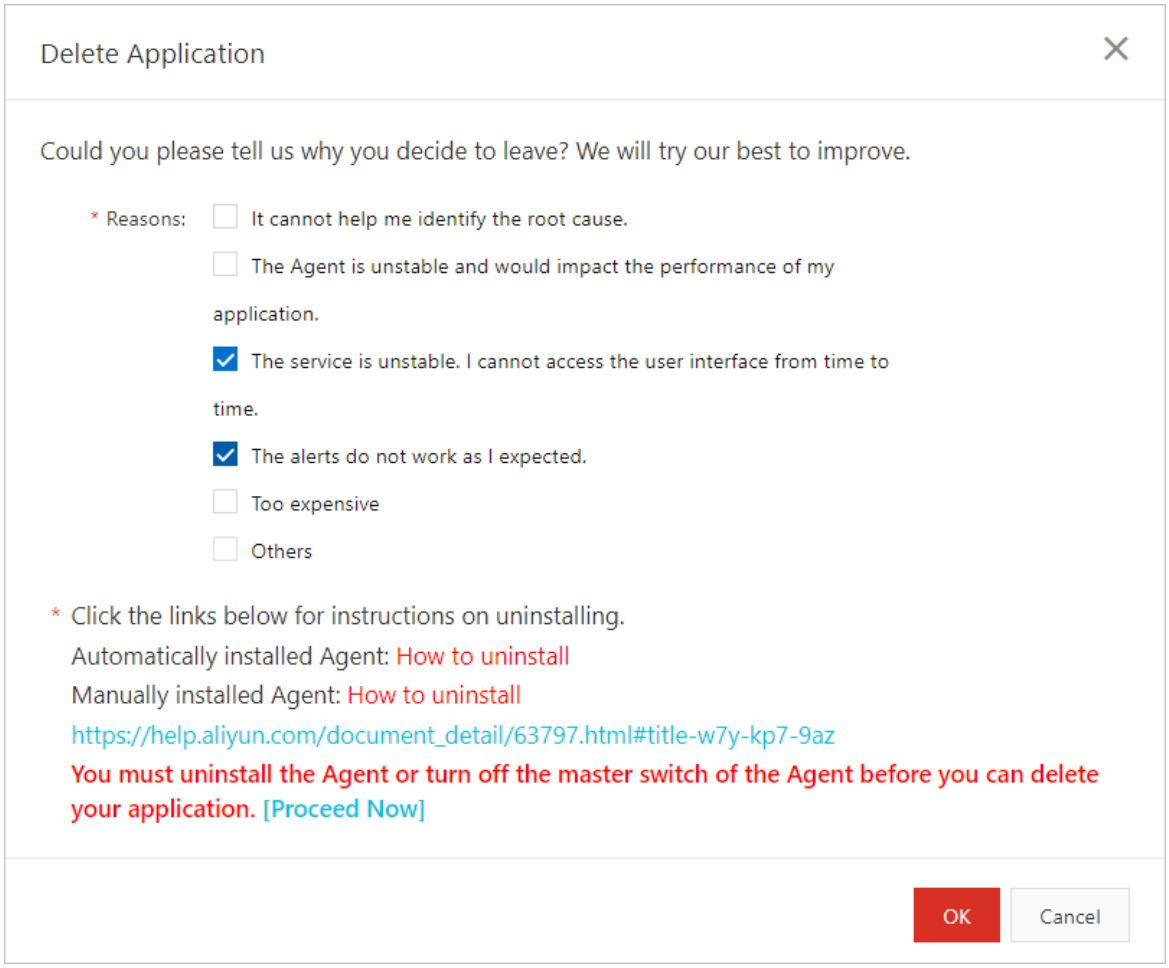
   > **Notice** The modification takes effect immediately without the need to restart the application. After you turn off Probe Master Switch, the system cannot monitor your application, and no fees are incurred. Proceed with caution when you perform this operation.

6. Uninstall the ARMS agent. For more information, see FAQ about ARMS agent uninstallation.
7. After the ARMS agent is uninstalled, click the **Delete** tab on the **Application Settings** page.

   > **Warning** This operation clears all monitoring data of the application, and the monitoring data cannot be restored.



8. On the **Delete** tab, click **Delete Application**. In the **Delete Application** dialog box, set **Reasons** and click **OK** to permanently delete the application.



Result

You can find that the deleted application no longer appears on the **Applications** page.

Related information

- FAQ

4.Tutorials

4.1. Use trace sampling policies

This topic describes a variety of trace sampling policies that are supported by Application Real-Time Monitoring Service (ARMS). You can select appropriate trace sampling policies based on your scenarios so that you can obtain the trace data that you want at a low cost.

Trace sampling is suitable for high-traffic applications that have a large number of visits. Trace sampling can help you record the most valuable trace data at a low cost and a low performance overhead. The basic principle of trace sampling is to preferentially record the traces that you are most concerned about and most likely to access. ARMS provides the following trace sampling policies:

- Trace feature-based sampling
- Business feature-based sampling
- O&M feature-based sampling
- Time feature-based sampling

> **Note** You can use the preceding trace sampling policies in combination to fully meet your personalized sampling requirements.

Trace feature-based sampling

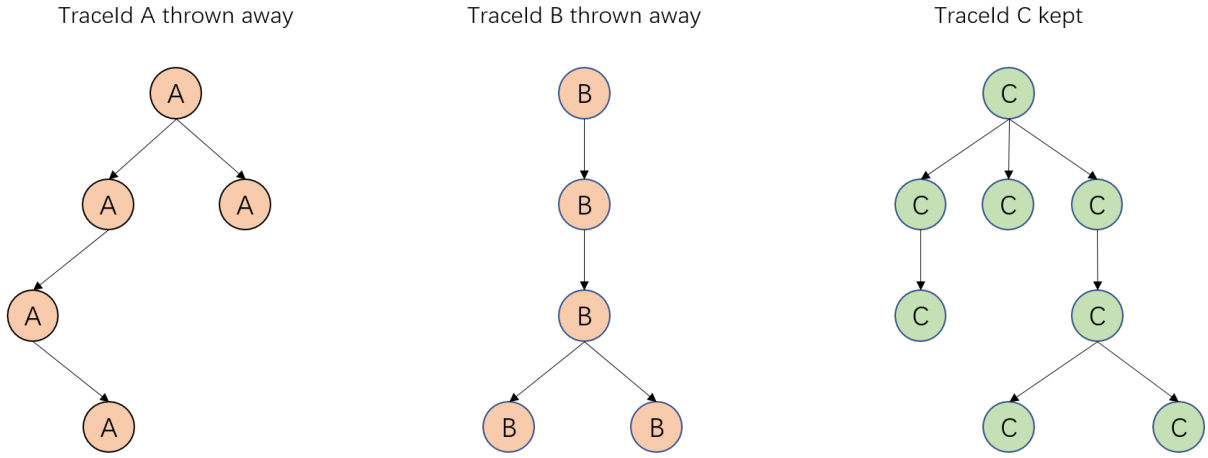
Trace feature-based sampling refers to the sampling based on the attributes of traces, such as time consumption and status. ARMS supports fixed-rate sampling and sampling based on thread profiling for slow calls.

Fixed-rate sampling records a specific proportion of trace data based on the ordinal number of TraceId. For example, if the fixed rate is 10%, one out of every 10 pieces of trace data is recorded. Fixed-rate sampling avoids incomplete trace data. The data of an entire trace is retained or discarded.

Fixed-rate sampling applies to the following scenarios:

- During the peak hours of stress testing or big promotions, the traffic volume is high. If full trace logs are reported, the client performance may be degraded. To avoid this situation, we recommend that you reduce the fixed sample rate to a value between 1% and 10%.
- On regular days, the cost of network bandwidth is high because full trace logs are reported. In this case, you can consider adjusting the fixed sample rate as needed.

The following figure shows how fixed-rate sampling works.



You can perform the following steps to configure fixed-rate sampling:

1. Log on to the ARMS console .
2. In the left-side navigation pane, choose **Application Monitoring > Applications** and select a region in the top navigation bar.
3. On the **Applications** page, click the name of the application.
4. In the left-side navigation pane, click **Application Settings**. On the page that appears, click the **Custom Configuration** tab.
5. In the **Invocation Trace Sampling Settings** section, turn on or turn off the switch of trace sampling and specify a sample rate. In the **Sampling Rate Settings** field, enter the number of the percentile. For example, if you enter *10*, the sample rate is 10%.

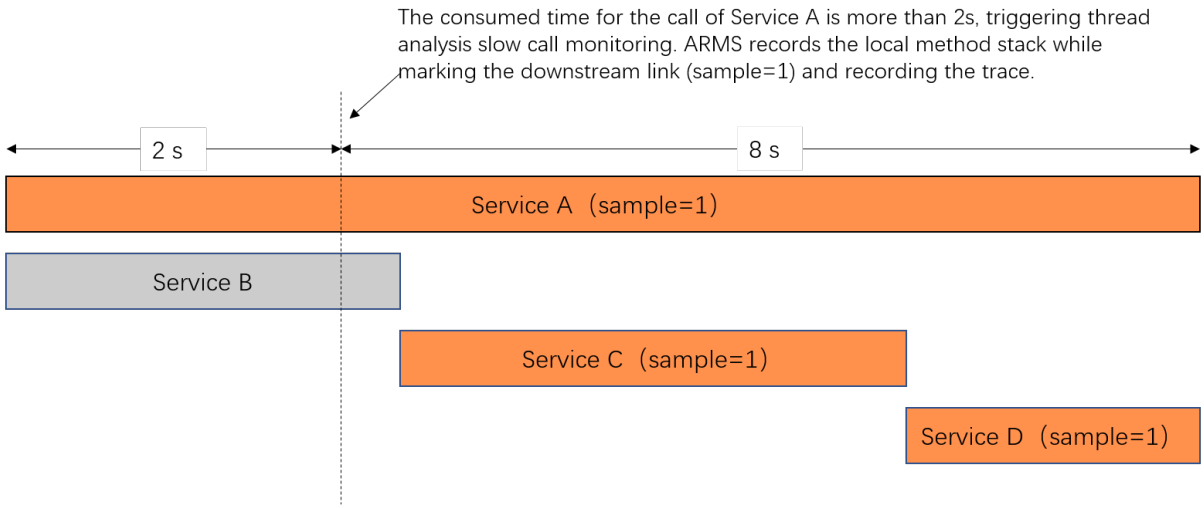
> **Notice** The modification takes effect immediately. You do not need to restart the application. If sampling is turned off, the trace data is not captured. Proceed with caution.

Sampling based on thread profiling for slow calls records data of traces in which thread profiling is triggered to listen on slow calls. After thread profiling is enabled, ARMS transfers a sampling mark to the downstream at the same time when ARMS records a slow call. ARMS also retains the downstream traces of this slow call. However, the number of listening threads of thread profiling is limited to avoid compromised client performance. If multiple slow calls occur at the same time, only some slow calls that meet the conditions and their downstream traces are recorded.

Sampling based on thread profiling for slow calls applies to the following scenarios:

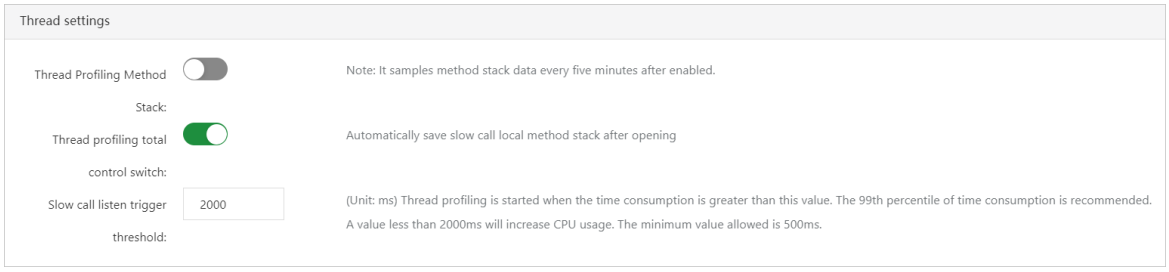
- The system encounters occasional slow calls. For example, the response time of services surges from 0.5s to 10s at night. Assume that you enable thread profiling in advance (the default trigger threshold is 2s). ARMS automatically records the native method stacks of slow calls that are complete within 2s to 10s in the request and the downstream traces of the slow calls.
- The system responds at a slow speed during the peak hours of flash sales or periodic big promotions. In this case, thread profiling records the native method stacks of slow calls that reach the trigger threshold and the downstream traces of the slow calls.

The following figure shows how sampling based on thread profiling for slow calls works.



You can perform the following steps to configure sampling based on thread profiling for slow calls:

1. Log on to the ARMS console .
2. In the left-side navigation pane, choose **Application Monitoring > Applications** and select a region in the top navigation bar.
3. On the **Applications** page, click the name of the application.
4. In the left-side navigation pane, click **Application Settings**. On the page that appears, click the **Custom Configuration** tab.
5. In the **Thread settings** section, turn on or turn off the switch of the thread diagnostics method stack and the master switch of thread profiling. Specify a threshold to trigger slow call listening.



> **Note** The listener is started only when the service call response time exceeds the threshold (1,000 ms by default) and lasts until the call ends or the consumed time exceeds 15 seconds. We recommend that you set the threshold to the 99th percentile of the call response time. For example, if 100 calls are listed in ascending order by response time, the time consumed by the 99th one is the 99th percentile.

Fixed-rate sampling

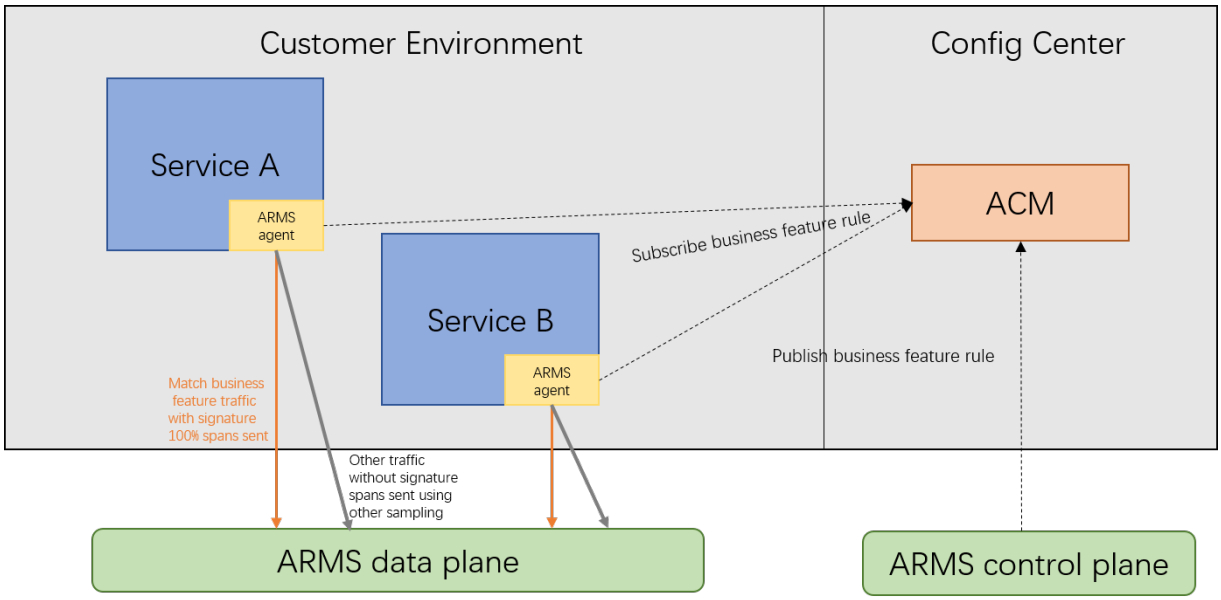
Sampling based on thread profiling for slow calls

Business feature-based sampling

Business feature-based sampling refers to the sampling based on the business traffic features of applications. ARMS allows you to configure rules based on the HTTP traffic features of entry applications. You can filter Header, Method, Cookie, and Parameter information when you extract business features so that the conditions for matching business features can be met in various scenarios. After the switch of full collection is turned on, the trace data that meets the business conditions is given priority to be fully collected. The configuration immediately takes effect, and can be dynamically modified during runtime.

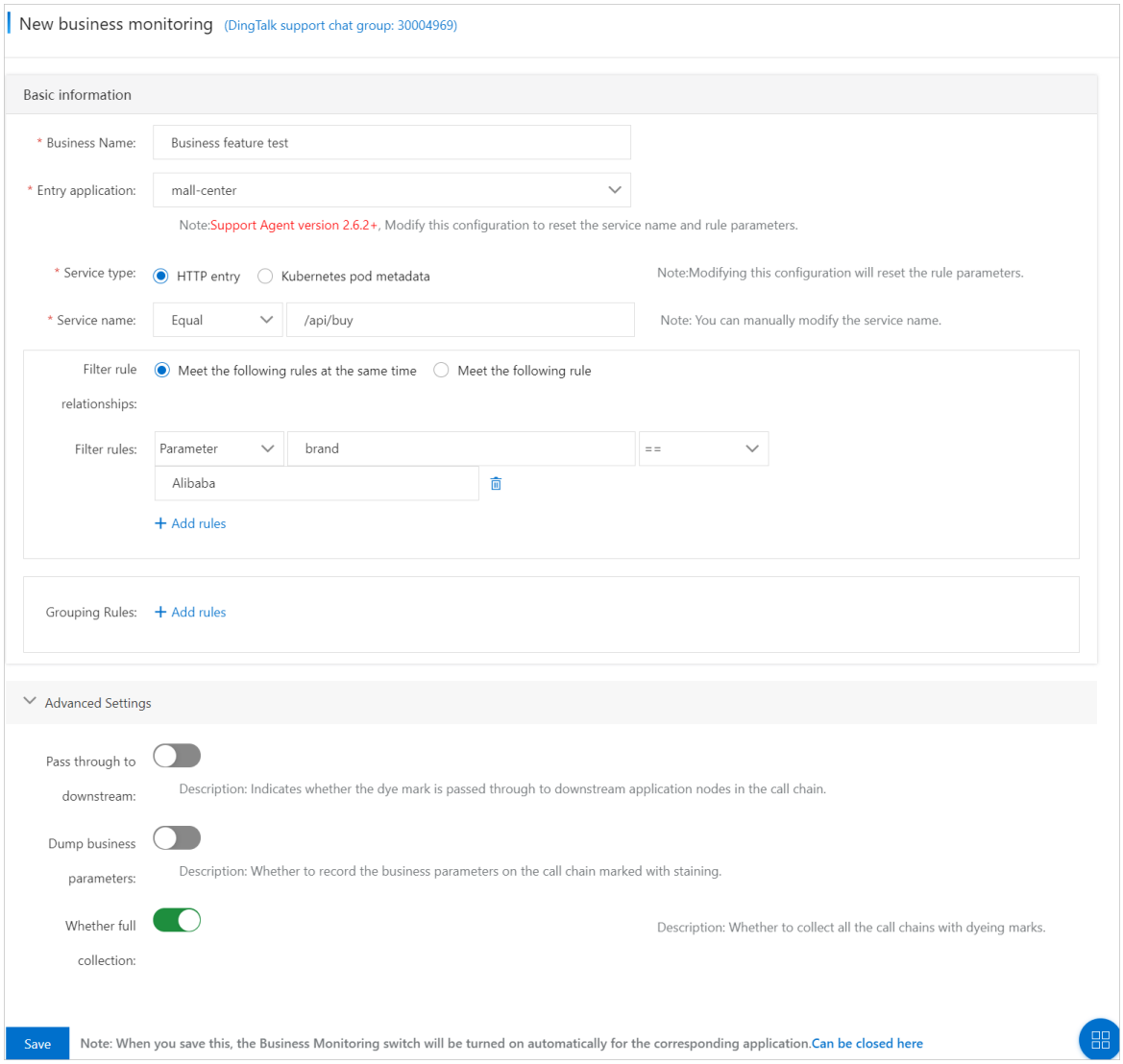
Business feature-based sampling applies to the following scenarios:

The following figure shows how business feature-based sampling works.



You can perform the following steps to configure sampling based on business features:

1. Log on to the ARMS console .
2.
3.
4. In the **Basic information** section on the **New business monitoring** page, set related parameters. Click **Advanced Settings**. In the **Advanced Settings** section, set related parameters. Then, click **Save**.



Parameter	Description	Example
Business Name	Required. The name of the business transaction task.	Business feature test

Parameter	Description	Example
Entry application	Required. The **Entry application** drop-down list displays all the Java applications that have the ARMS agent installed. After you select the required application, ARMS automatically detects the version number of the agent. **Note** You can use the business transaction feature only after you upgrade the ARMS agent to a version later than 2.6.2. If the detected agent version is not later than 2.6.2, upgrade the agent first. For more information, see Update the ARMS agent for Java applications.	mall-center
Service type	Required. The type of the service. ARMS supports only **HTTP entry**. This service type is suitable for scenarios where business traces are dyed based on HTTP traffic features.	**HTTP entry**

Parameter	Description	Example
Service name	Required. The API name that is provided by the application. ARMS automatically detects a list of APIs that are recently provided by this application based on the **entry application** you specify for your choices. If the recommended APIs do not meet your requirements, you can edit the APIs. The following four types of matching patterns of **Service name** are supported: ◦ **Equal**: matches the business API that uses the value of the **Service name** parameter as the API name. This pattern is the default matching pattern. ◦ **Start equal**: matches the business APIs whose names are prefixed with the value of the **Service name** parameter. If you need to monitor the business APIs that have the same prefix, select this pattern. ◦ **Contains**: matches the business APIs whose names contain the value of the **Service name** parameter. If your application provides a large number of business APIs, you can select this pattern to quickly monitor the business APIs that you need. ◦ **End =**: matches the business APIs whose names are suffixed with the value of the **Service name** parameter. This pattern is suitable for typical web frameworks that end with the .do and .action configurations. ◦ **Pattern matching**: matches dynamic URI paths and supports matching rules for Ant-style path patterns. This allows you to monitor and analyze the URIs of a	**Equal** /api/buy

Parameter	Description	Example
	specific type of patterns.	
Filter rule relationships	You can select **Meet the following rules at the same time** or **Meet the following rule**.	**Meet the following rules at the same time**
Filter rules	Optional. Further filter the business APIs that you specify. To specify **Filter rules**, you must specify a parameter to be matched (**Parameter**, **Cookie**, **Method**, **PathVariable**, or **Header**), a key value to be matched, a matching pattern (**==**, **!=**, or **contains**), and a threshold. Among the preceding information, the **PathVariable** option appears for the parameter to be matched only if you select **Pattern matching** for the **Service name** parameter and the input string includes braces {} as placeholders. You can set multiple filter rules. The logical relationship between multiple filter rules is determined by the **Filter rule relationships** parameter that you specify.	Assume that your application provides the /api/buy?brand=*** URL and you want to monitor the API calls of brand=Alibaba. You can set the **Filter rules** parameter to **Parameter** brand == Alibaba in the form.
Whether full collection	Specifies whether to collect all the traces that have dye labels.	Yes

O&M feature-based sampling

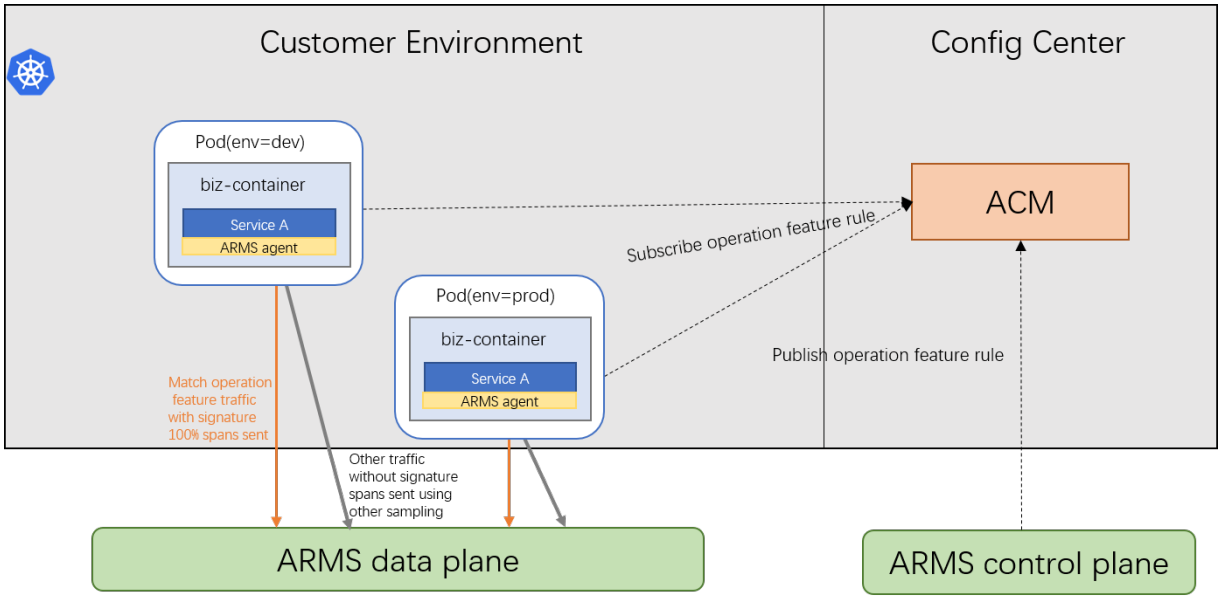
O&M feature-based sampling refers to the sampling based on the O&M features of applications, such as the deployment environment and the network environment. ARMS supports sampling based on the Kubernetes Pod Metadata features of applications. ARMS allows you to configure O&M feature rules based on the Kubernetes Pod Metadata features of entry applications. ARMS can automatically identify the metadata information of pod instances of applications, such as the label, annotation, pod name, and namespace, and filter the metadata information. This meets the conditions for matching O&M features in various scenarios. After the switch of full collection is turned on, the trace data that meets the O&M feature conditions is given priority to be fully collected. The configuration immediately takes effect, and can be dynamically modified during runtime.

O&M feature-based sampling applies to the following scenarios:

- Specific sampling settings are required based on the difference of information about application agents, such as languages, versions, environment variables, and startup parameters.
- Specific sampling settings are required based on the difference of information about servers where applications are deployed, such as specifications, models, regions, and zones.
- Specific sampling settings are required based on the difference of application deployment environments, such as the development, testing, pre-release, and production isolation requirements.
- Specific sampling settings are required based on the difference between deployment modes of

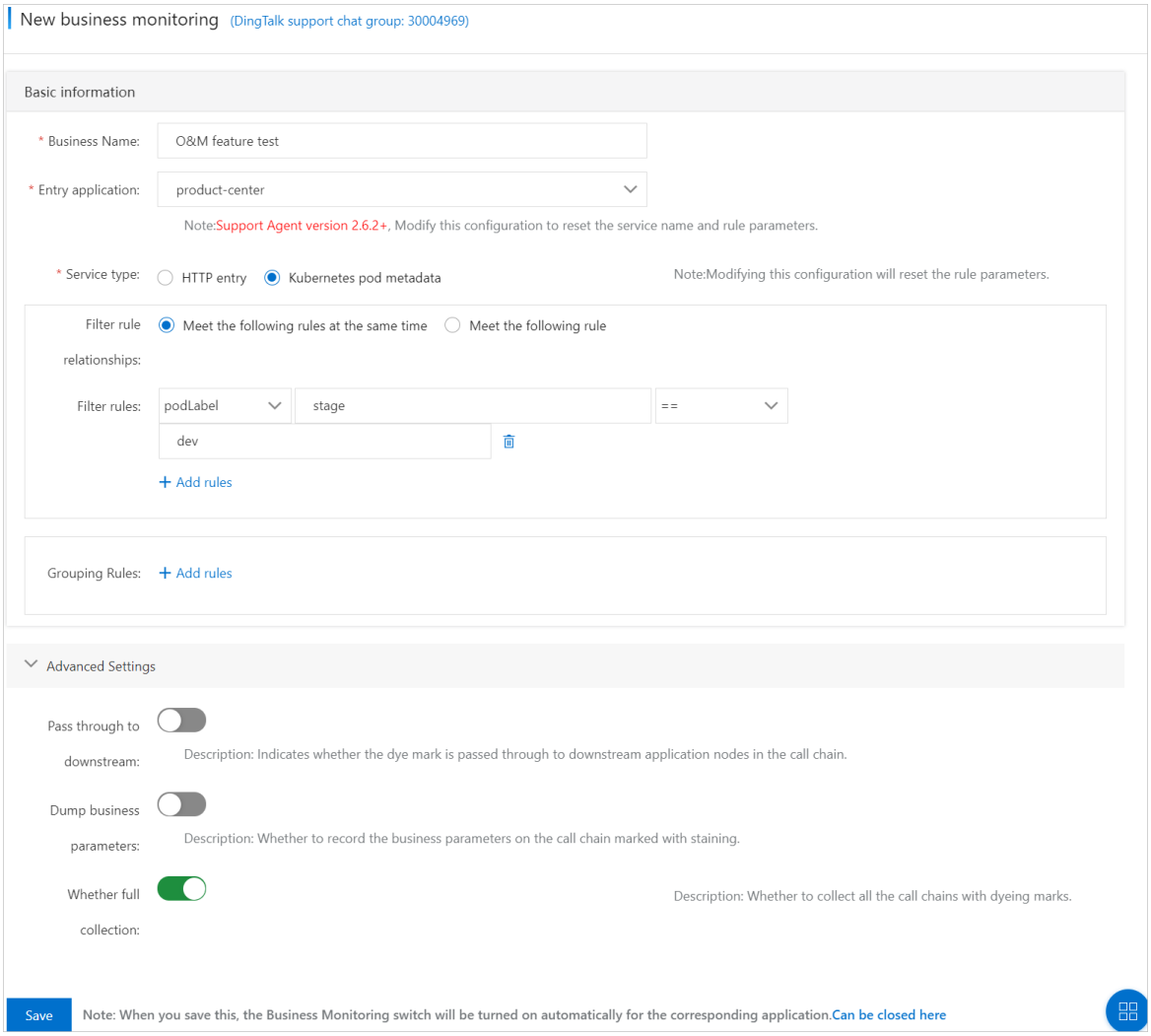
network environments where applications reside, such as physical machines, virtual machines, and containers. Alternatively, specific sampling settings are required based on the difference between the classic network and a VPC.

The following figure shows how O&M feature-based sampling works.



You can perform the following steps to configure O&M feature-based sampling:

1. Log on to the ARMS console .
2.
3.
4. In the **Basic information** section on the **New business monitoring** page, set related parameters. Click **Advanced Settings**. In the **Advanced Settings** section, set related parameters. Then, click **Save**.



Parameter	Description	Example
Business Name	Required. The name of the business transaction task.	O&M feature test

Parameter	Description	Example
Entry application	Required. The **Entry application** drop-down list displays all the Java applications that have the ARMS agent installed. After you select the required application, ARMS automatically detects the version number of the agent. **Note** You can use the business transaction feature only after you upgrade the ARMS agent to a version later than 2.6.2. If the detected agent version is not later than 2.6.2, upgrade the agent first. For more information, see Update the ARMS agent for Java applications.	product-center
Service type	Required. The type of the service. ARMS supports only **Kubernetes Pod Metadata**. This service type is suitable for scenarios where business traces are dyed based on the environment features of Kubernetes pods.	**Kubernetes Pod Metadata**.
Filter rule relationships	You can select **Meet the following rules at the same time** or **Meet the following rule**.	**Meet the following rules at the same time**
Filter rules	Optional. Further filter the business APIs that you specify. To specify **Filter rules**, you must specify a parameter to be matched (**podLabel**, **podAnnotation**, **podName**, **podNamespace**, **podUID**, **podIp**, **nodeName**, **hostIp**, or **podServiceAccount**), a matching pattern (**==**, **!=**, or **contains**), and a threshold. You can set multiple filter rules. The logical relationship between multiple filter rules is determined by the **Filter rule relationships** parameter that you specify.	Assume that your application is deployed in the dev, test, staging, and prod environments. The Pod label stage is used for environment differentiation. You want to monitor the dev environment and collect all traces. In this case, you can set the **Filter rules** parameter to **podLabel** stage == dev in the form.

Parameter	Description	Example
Whether full collection	Specifies whether to collect all the traces that have dye labels.	Yes

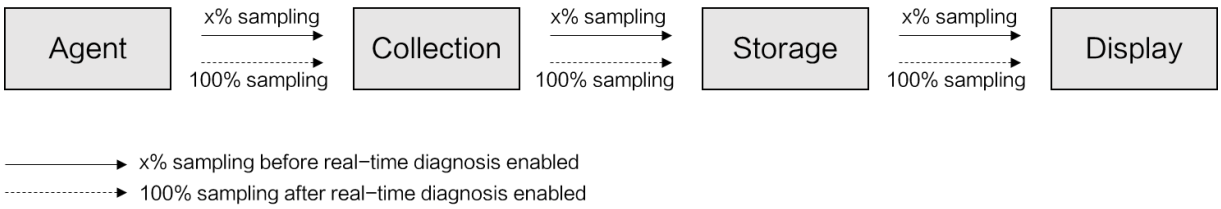
Time feature-based sampling

Time feature-based sampling refers to the sampling based on the time features of applications, such as online diagnostics and offline analysis. Requirements on sample rates vary based on diagnostic scenarios. ARMS supports temporary full collection in online diagnostic scenarios and helps you quickly locate online problems. After real-time diagnostics is enabled, ARMS continuously monitors the application for 5 minutes and reports all the trace data during this period. Then, you can start from the trace that has performance problems, and use features, such as the waterfall charts of method stacks and thread profiling, to identify the causes of the problems.

Time feature-based sampling applies to the following scenarios:

You need to closely monitor the application performance for a short period of time, for example, when you release an application or perform stress testing on the application.

The following figure shows how time feature-based sampling works.



You can perform the following steps to configure time feature-based sampling:

1. Log on to the ARMS console .
2. In the left-side navigation pane, choose **Application Monitoring > Applications** and select a region in the top navigation bar.
3. On the **Applications** page, click the name of the application.
4. In the left-side navigation pane, choose **Application Diagnosis > Real-time Diagnosis**.

The first time you go to the **Real-time Diagnosis** page, real-time diagnostics is automatically enabled. To enable real-time diagnostics in other cases, click **Activate Real-time Diagnosis** in the upper-right corner of the page.
Real-time diagnostics is automatically enabled for 5 minutes and then disabled. To disable real-time diagnostics before it is automatically disabled, click **Pause Auto-Refresh** in the upper-right corner.

4.2. Analyze errors in code by using ARMS thread profiling

Application Real-Time Monitoring Service (ARMS) thread profiling is a code-level diagnosis tool. It can automatically capture stack snapshots of slow calls and restore the code execution process.

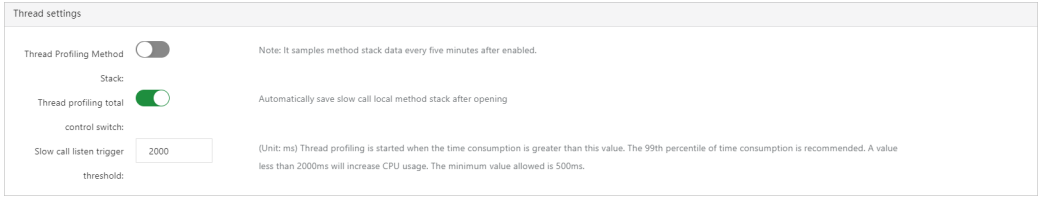
Scenario

- ARMS thread profiling can quickly locate problematic code during periods of high traffic such as sales events.
- If large amounts of slow calls occur in the system, ARMS thread profiling can automatically save the first scene.
- If occasional slow calls cannot recur because the business is too complex, ARMS thread profiling can restore the real code execution process.

Set thread profiling parameters

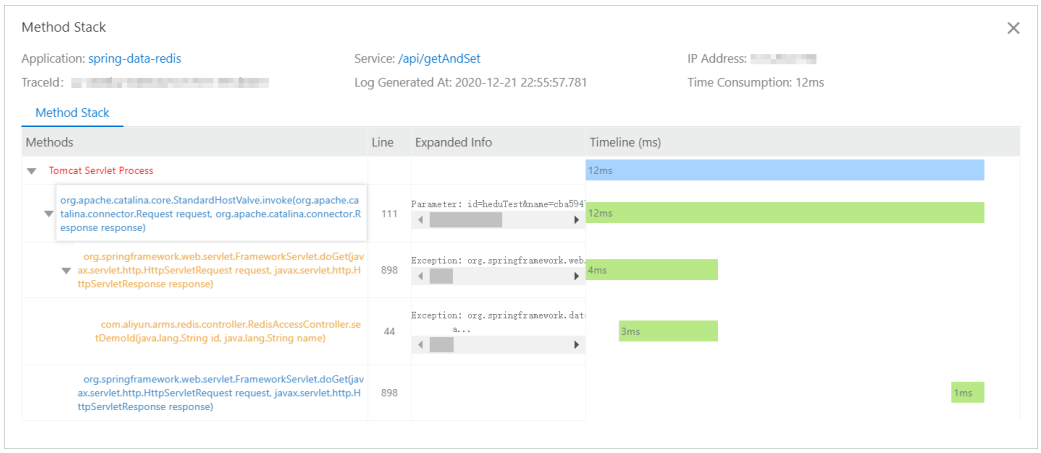
1. Log on to the ARMS console .
2. In the left-side navigation pane, choose **Application Monitoring > Applications** and select a region in the top navigation bar.
3. On the **Applications** page, click the name of the application.
4. In the left-side navigation pane, click **Application Settings**. On the page that appears, click the **Custom Configuration** tab.
5. In the **Thread Settings** section, you can turn on or turn off the thread profiling total control switch and set the slow call listen trigger threshold.

> **Note**
>
> - The listener starts only when the service call response time exceeds the threshold (2,000 ms by default) and it lasts until the call ends or the consumed time exceeds 15 seconds.
> - We recommend that you set the threshold to the 99th percentile of the call response time. For example, if 100 calls are listed in ascending order by response time, the time consumed by the 99th one is the 99th percentile.



View thread profiling details by using interface snapshots

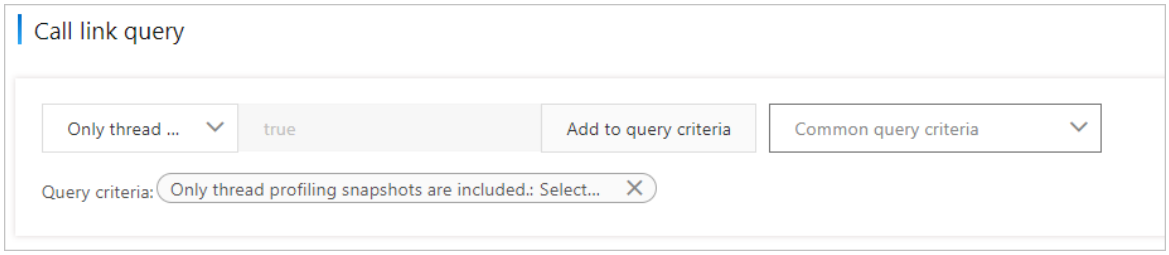
1. In the left-side navigation pane, choose **Application Monitoring > Applications** and select a region in the top navigation bar.
2. On the **Applications** page, click the name of the application.
3. In the left-side navigation pane, click **Interface Invocation**, select the interface on the right side of the page, and click the **Interface Snapshot** tab.
4. On the **Interface Snapshot** tab, click a TraceId link.
   The **Traces** tab appears.
5. In the **Thread Profiling** column, click the magnifier icon.
   The **Thread Profiling** dialog box appears.



> **Note**
>
> - The actual response time is the amount of time it takes for the service call to execute and is not affected by thread profiling.
> - The listening response time is the amount of time consumed by thread profiling. Typically, the listening response time is approximately the actual response time minus the listening threshold for slow calls.

View thread profiling details by using trace query

1. In the left-side navigation pane of the console, choose **Application Monitoring > Invocation Trace Query**.
2. In the **Parameter Name** drop-down list of the **Invocation Trace Query** tab, select **Only thread profiling snapshots are included.** and click **Search**.



3. Click a TraceId link in the search results.
   The **Traces** tab appears.
4. In the **Thread Profiling** column, click the magnifier icon.

The **Thread Profiling** dialog box appears.

FAQ

- Q: What is the actual response time?
  A: The actual response time is the amount of time it takes for the service call to execute and is not affected by thread profiling.
- Q: What is the listening response time?
  A: The listening response time is the amount of call execution time consumed by thread profiling. To minimize listening pressure, thread profiling listens only for the execution time that exceeds the listening threshold of slow calls. The default threshold is 2 seconds. For example, assume that thread profiling listens for a slow call that consumes 5 seconds and only listens to the interval between 3 and 5 seconds. If a call consumes 1.8 seconds, the listener does not listen to the call.
- Q: Why is the listening response time shorter than the actual response time? Why does the listener miss some slow calls that exceed the listening threshold?
  A:
  - Thread profiling listens only for the execution time of a call after the listening threshold is exceeded. Typically, the listening response time is approximately the actual response time minus the listening threshold for slow calls.
  - If the system has a large number of slow calls at the same time, the listener may miss some slow calls after the listening threshold is triggered because the number of listening threads is limited. In this case, the listening response time is shorter than the actual response time and the listener may miss the calls.
  - To ensure that slow calls that exceed five seconds are listened, thread profiling configures independent listening threads for these slow calls. In this case, the listening response time is approximately the actual response time minus five seconds.

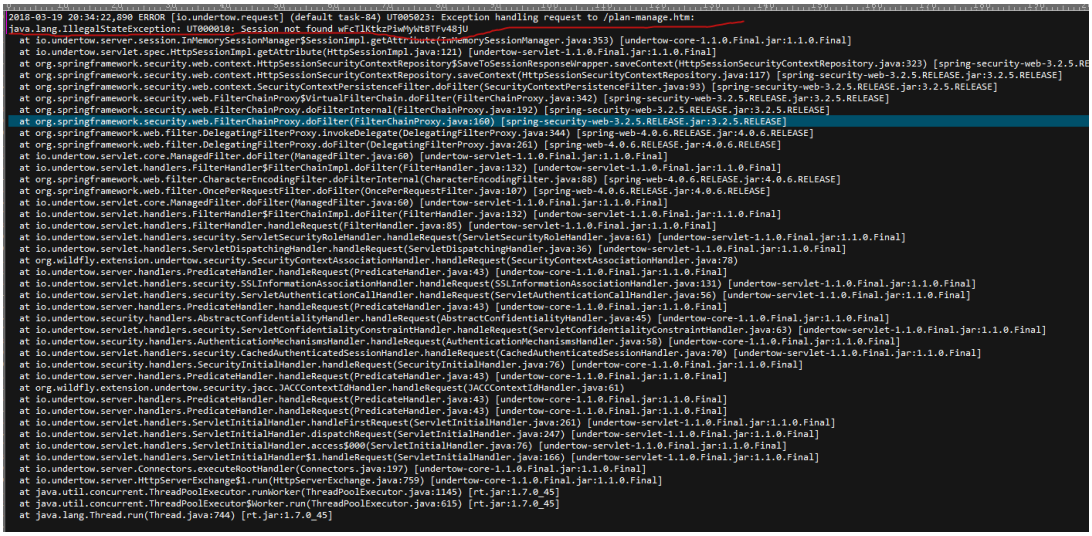
4.3. Diagnose errors on the server

It is challenging to analyze the causes of web page errors, which are one of the most common problems of Internet applications. After the Application Real-Time Monitoring Service (ARMS) agent is installed on an application, the ARMS agent can automatically capture, collect, count, and track exceptions without the need to modify the application code. You can use the ARMS agent to accurately locate all exceptions in the application and perform online diagnostics.

Problem description

Web page errors, particularly those that take the form of a 5xx error, are one of the most common problems of Internet applications. 5xx errors usually occur on the server side. The server side has the most complex business logic and is the most error-prone part of the entire network-request link. Errors on the server side prove to be the most challenging for cause analysis. O&M engineers or development engineers often need to log on to the server to view logs and find the causes.

Example: Common error logs of Java applications



For applications that have less complex logic and short uptimes, the operation to log on to the server to view logs can solve most of these problems. However, this traditional method is often useless in the following scenarios:

- You want to know the time and frequency of a specific type of error in a distributed application cluster.
- A system has been running for a long time, but you do not care about the residual exceptions. You want to know only about new exceptions today compared with yesterday, and new exceptions after the release of the system compared with the period before the release of the system.
- You view the web requests and relevant parameters associated with an exception.
- Customer Services provides the number of an order that a user fails to place for cause analysis of the failure.

Solution

Install the ARMS agent on the application. The ARMS agent can automatically capture, collect, count, and track exceptions without the need to modify the application code. The ARMS agent presents a clear picture of various errors.

Step 1: Install the ARMS agent

The application can be monitored in all aspects only after you install the ARMS agent. Select one of the following methods to install the ARMS agent.

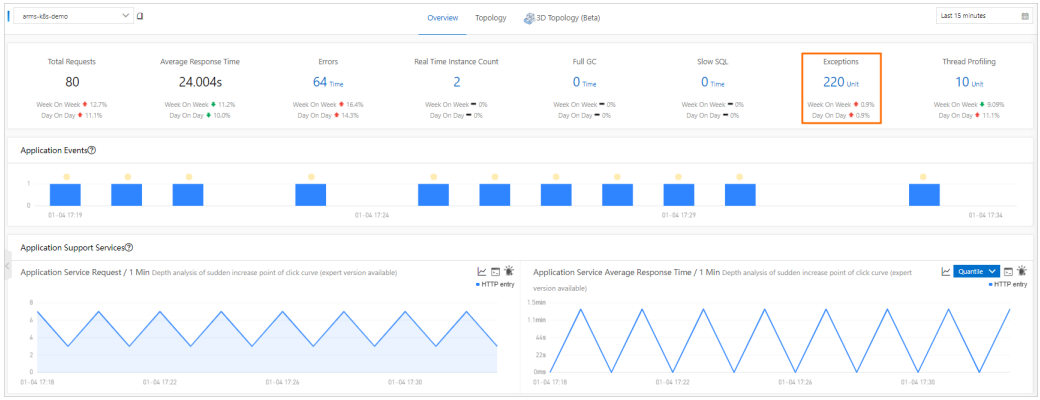
- For more information about how to install the ARMS agent for a Java application, see Manually install the ARMS agent for a Java application.
- For more information about how to install the ARMS agent for a PHP application, see Install the ARMS agent for a PHP application.
- For more information about how to install the ARMS agent for an application in Enterprise Distributed Application Service (EDAS), see Enable ARMS to monitor an EDAS application.
- For more information about how to install the ARMS agent for an application in a Container Service Kubernetes cluster, see Install the ARMS agent for a Java application deployed in Container Service for Kubernetes.
- For more information about how to install the ARMS agent for an application in an open source Kubernetes cluster, see Install the ARMS agent for an application deployed in an open source Kubernetes environment.

Step 2: View statistics of application exceptions

The installed ARMS agent collects and shows the average response time and number of requests, errors, real-time instances, full garbage collection (GC) events, slow SQL queries, exceptions, and slow calls of the application within the selected period of time. The ARMS agent also shows how these metrics change on a day-over-day and week-over-week basis. Perform the following steps to view the statistics of application exceptions:

1. Log on to the ARMS console .
2. In the left-side navigation pane, choose **Application Monitoring > Applications**.
3. In the top navigation bar, select the region where your application is deployed.
4. On the **Applications** page, click the name of your application.
5. On the **Application Overview** page, click the **Overview** tab. On the Overview tab, the total number of exceptions and how the number changes from the previous day and previous week are displayed in the lower part.

   Count of exceptions



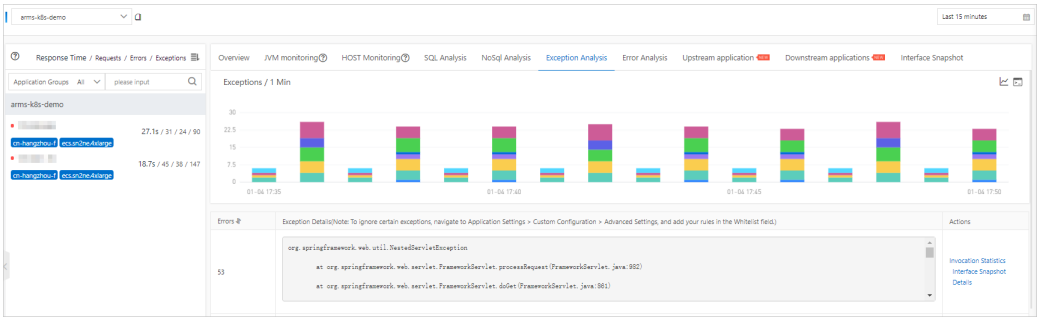
6. Scroll to **Exception Type** in the **Statistics Analysis** section at the bottom of the **Overview** tab. Here you can view the number of times for which each type of exception occurs.

   The number of times each type of exception occurs

Exception	Occurrence/Percentage
org.springframework.web.util.NestedServletException at org.springframework.web.servlet.FrameworkServlet.processRequest(FrameworkServlet.java:982) at org.springframework.web.servlet.FrameworkServlet.doGet(FrameworkServlet.java:861) Caused by: java.lang.NullPointerException	51 / (21.61%)
java.lang.NullPointerException at com.alibaba.arms.demo.util.Util.sendGet(Util.java:108) at com.alibaba.arms.demo.TestComponentInvodeController.invokeComponent(TestComponentInvodeController.java:31)	47 / (19.92%)
java.net.UnknownHostException at java.net.InetAddress.getAllByName0(InetAddress.java:1281) at java.net.InetAddress.getAllByName(InetAddress.java:1193)	46 / (19.49%)
java.net.UnknownHostException at java.net.Inet4AddressImpl.lookupAllHostAddr(Native Method) at java.net.InetAddress$2.lookupAllHostAddr(InetAddress.java:929)	46 / (19.49%)

7. In the left-side navigation pane, click **Application Details**. On the Application Details page, click the **Exception Analysis** tab in the right-side pane to view the exception statistics chart, count of errors, and exception stack.

   Exception Analysis tab



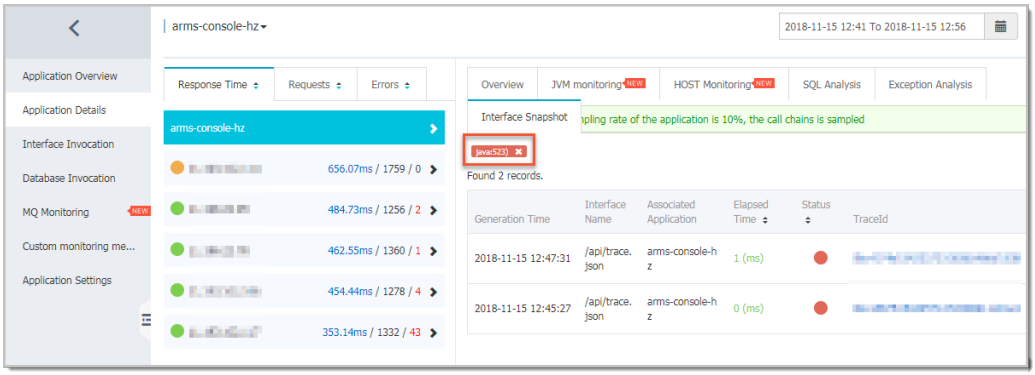
Step 3: Diagnose causes of exceptions

The statistics of application exceptions are insufficient for locating the causes of exceptions. The exception stack in the log contains the code snippet for a call. However, it does not contain the complete upstream and downstream information and request parameters of this call. The bytecode enhancement technology of the ARMS agent allows you to capture complete upstream and downstream call snapshots for exceptions, and the compromise to performance is small. Then, you can identify the specific causes of exceptions.

1. On the **Exception Analysis** tab, find the type of exception that you want to diagnose, and click **Interface Snapshot** in the **Actions** column.
   On the **Interface Snapshot** tab, the call trace information related to this exception type is displayed.
2. On the **Interface Snapshot** tab, click the TraceId of a problem call.

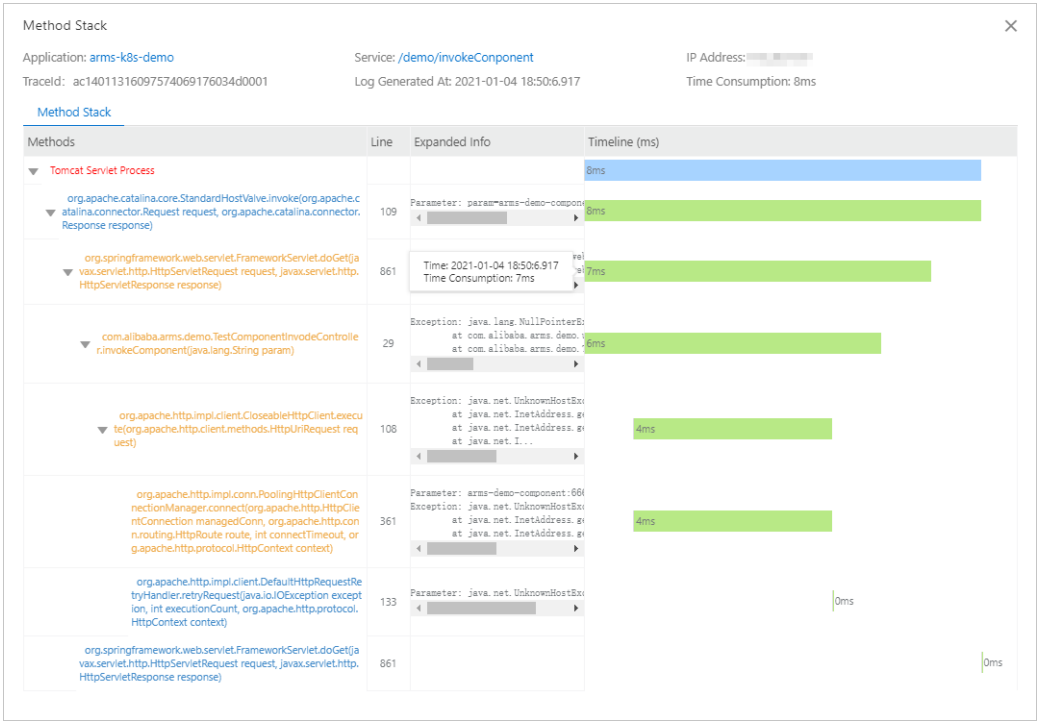
> **Note** For more information about how to find the trace of a problem call, see Trace query.

Interface Snapshot tab



3. On the page that appears, view the trace information about the problem call. In the **Method Stack** column, click the magnifier icon to view the method stack that is called. This way, you can obtain the context information about this problem call.

Complete trace information of the problem call



At this point, the causes of the exception are found. This effectively helps you with the subsequent code optimization. You can also return to the **Interface Invocation** tab to view other problem calls in the list and solve related exceptions one by one.

What to do next

To prevent passive diagnostics after an exception occurs, you can also use the alert feature of ARMS to create an alert for a specific API or all APIs. This ensures that the O&M team receives a notification immediately after an exception occurs. For more information about how to create an alert, see Create ARMS alerts.

References

- Create an application monitoring job
- API monitoring
- Trace query
- Analyze errors in code by using ARMS thread profiling
- Create ARMS alerts

4.4. Diagnose application access problems

The locating and troubleshooting of the causes for application access problems present many challenges. Application Monitoring of Application Real-Time Monitoring Service (ARMS) provides a set of solutions, such as thread profiling, tracing diagnosis, and API monitoring. These solutions help you quickly and accurately locate all slow calls in an application and solve the application access problems.

Analysis

Website freezing and slow webpage loading are among the most common problems of Internet applications. Troubleshooting and solving these problems is complex and takes a long time. Here are the major reasons:

- Overly long application tracing

- The cause can be any failure at any stage of the long trace. The failure can happen between the front-end webpage and the back-end gateway, or between the web application server and the back-end database.
- Applications using the microservice architecture have even more complex traces. Different components of applications might be maintained by different teams and persons, which makes troubleshooting more difficult.

- Incomplete or low-quality logs
  - Most methods of troubleshooting online problems rely on application logs. However, the locations of problems are often unpredictable, and "slow response" often occurs. To find the true causes of "slow response", you need to print the log in every place where errors can occur and record each call. The cost is very high for doing so.
- Insufficient monitoring
  - Rapid business development and fast application iteration lead to frequent API changes of applications and increased dependencies. This further contributes to the deterioration of the quality of code. Applications need a comprehensive monitoring system that automatically monitors every API of the application and records abnormal calls.

Solution

After being installed, the ARMS agent can use ARMS application monitoring features, such as thread profiling, trace diagnosis, API monitoring, to monitor all slow calls. This does not change the code of your application.

Prerequisites

Make sure you have installed the ARMS agent for your applicartion.

Step 1: View the statistics of slow SQLs

The installed ARMS agent collects and shows the application's total requests, average response time, count of errors, real-time instances, number of full garbage collection (GC) activities, slow SQLs, exceptions, and slow calls within the selected period. The agent also shows how these metrics change when compared with their counterparts in the previous day and previous week. Follow these steps to view the statistics of slow SQLs.

1. In the left-side pane of the ARMS console, choose **Application Monitoring > Applications**.
2. On the **Applications** page, click the name of your application.
3. On the **Application Overview** page, the total number of exceptions and how these changed from the previous day and previous week are displayed at the top of the **Overview Analysis** tab. In this example, there are 42 times of slow SQLs.

Step 2: Find and locate slow APIs

On the **Interface Invocation** page, ARMS shows all APIs provided by the monitored application, and the number of calls and consumed time on this API. Slow APIs are marked to help you quickly locate them.

1. Click **Interface Invocation** in the left-side pane of the ARMS console.
2. In the **Interface Selection** section of the **Interface Invocation** page, select the slow API that is called the most frequently. View detailed information of the API on the right.

Step 3: View and locate the problem code

After locating a slow API, you need to find the problem code to eliminate the problem. A snapshot is a complete record of an entire trace. The snapshot includes the code and time consumption of each call, and helps you precisely locate the problem code.

1. Click the **Interface Snapshot** tab on the **Interface Invocation** page.
   On the **Interface Snapshot** tab, you can view snapshots of all APIs corresponding to this API.
2. On the **Interface Snapshot** tab, click the TraceId of a trace, and then click the magnifier icon in the **Method Stack** column to view the problem code.
   To find the target trace, see Trace query.
   In this example, most of the time of this 705 ms call is spent in calling the SQL `SELECT * FROM l_employee` .

At this point, the causes for a specific slow call are revealed. This effectively helps you with the subsequent code optimization. You can also return to the **Interface Invocation** page to view other slow calls in the list and solve them one by one.

4.5. Troubleshoot exceptions by using diagnostic reports

It is a complex and time-consuming task to identify and troubleshoot exceptions. Application Real-Time Monitoring Service (ARMS) provides the active diagnostics feature to help you identify exceptions such as long response time.

Step 1: Install an ARMS agent

After you install an ARMS agent, ARMS can conduct all-around monitoring of your applications. Select one of the following methods to install an ARMS agent.

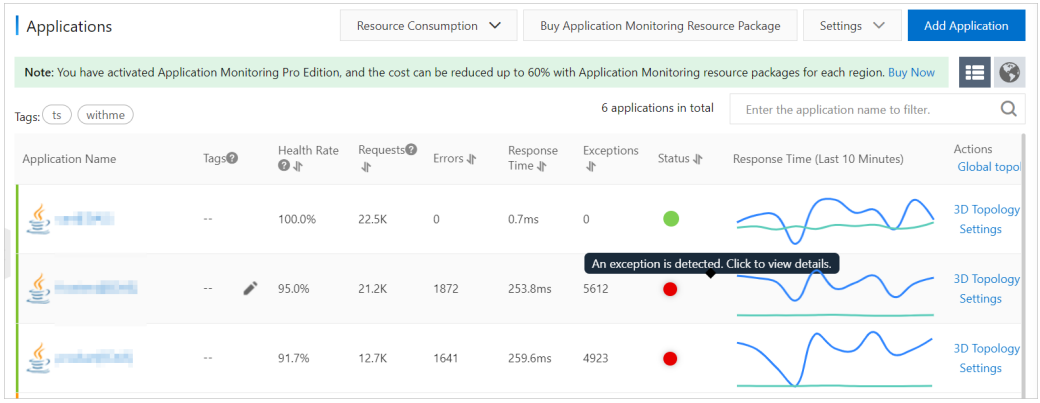
- To install an ARMS agent for Java applications, see Manually install the ARMS agent for a Java application.
- To install an ARMS agent for PHP applications, see Install the ARMS agent for a PHP application.
- To install an ARMS agent for applications in EDAS, see Enable ARMS to monitor an EDAS application.
- To install an ARMS agent for applications in Container Service for Kubernetes, see Install the ARMS agent for a Java application deployed in Container Service for Kubernetes.
- To install an ARMS agent for applications in open source Kubernetes environments, see Install the ARMS agent for an application deployed in an open source Kubernetes environment.

Step 2: View the diagnostic report

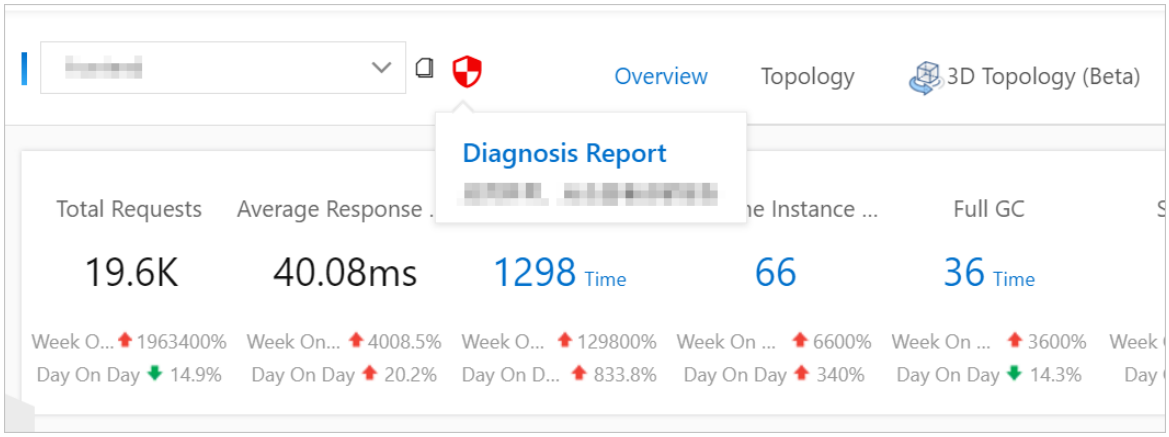
The installed ARMS agent collects and shows the metrics of applications such as Total Requests, Average Response Time, Errors, Real Time Instance Count, Full GC, Slow SQL, Exceptions, and Thread Profiling within the selected period of time. ARMS allows you to view the diagnostic report of troubleshooting metrics. You can also view the diagnostic report of a single metric on the **Application Overview** page.

1. Log on to the ARMS console .
2. In the left-side navigation pane, choose **Application Monitoring > Applications**. In the top navigation bar, select a region.
   On the **Applications** page, if the application has an exception, the **Status** column is displayed in red.
3. On the **Applications** page, move the pointer over the row of the application. The **An exception is detected. Click to view details** message appears. Click the red dot to load the diagnostic

report. After the diagnostic report is loaded, move the pointer over the red dot and click **Diagnostic Report** to go to the **Diagnostic Report** page.



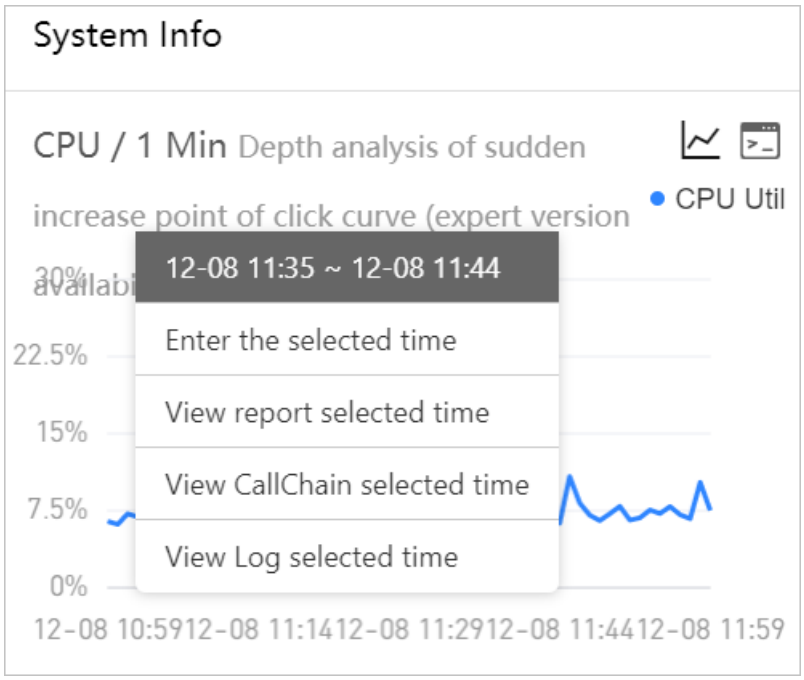
You can also click the name of the application on the **Applications** page to go to the **Application Overview** page. Move the pointer over the icon on the right of **Application Health Overview**. If the **An exception is detected. Click to view details** message appears, click the icon to load the diagnostic report. After the diagnostic report is loaded, click the icon again to go to the **Diagnostic Report** page.



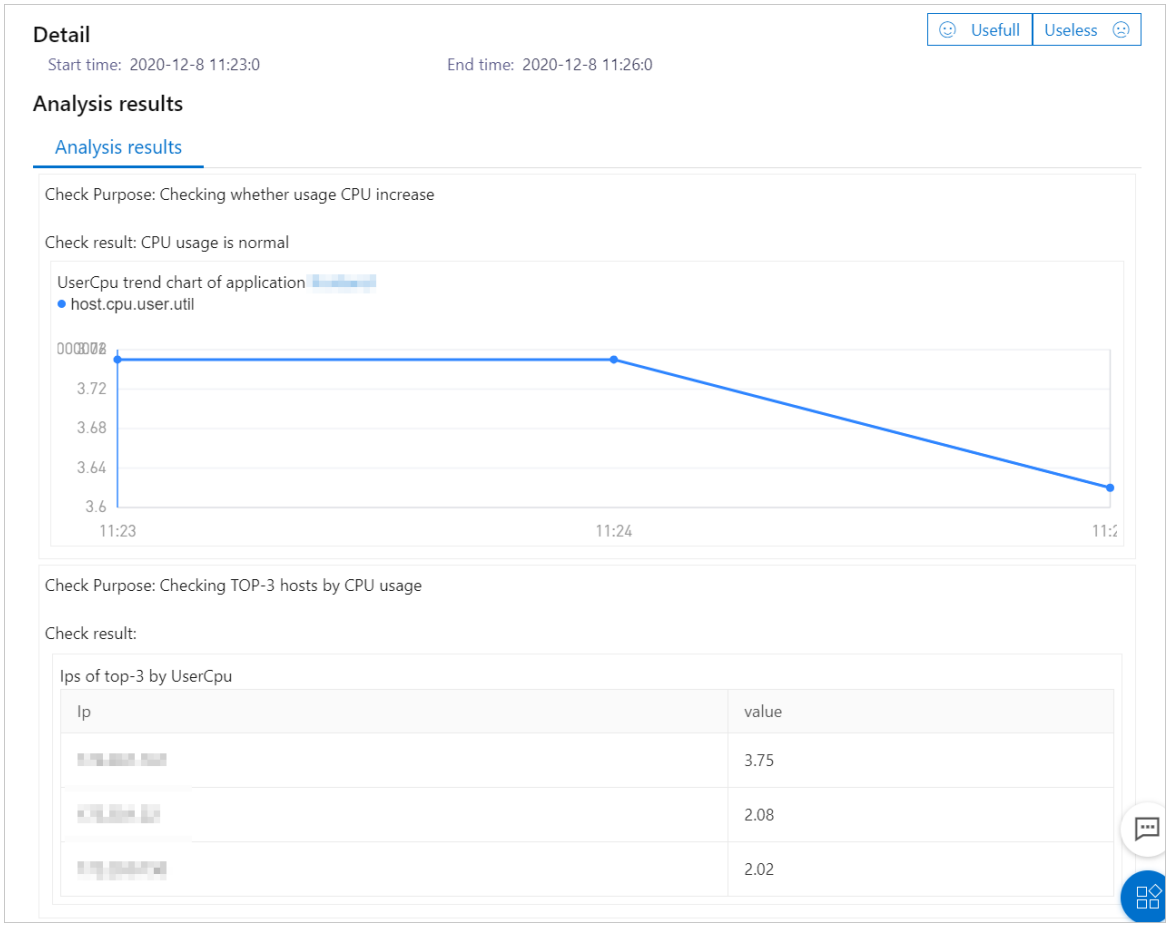
4. On the **Diagnostic Report** page, view the application name, diagnostic time, symptom, error demarcation, root cause analysis, and detection results of metrics.

   Diagnostic report

1. On the **Applications** page, click the name of the application to go to the **Application Overview** page.
2. On the **Application Overview** page, drag the pointer over the curve of a metric and select a time period. Click **View report of selected time** to go to the **Details** page.



3. On the **Details** page, view the detection results of this metric for the selected time range.



View the diagnostic report of all metrics

View the diagnostic report of a single metric

What to do next

To avoid passive diagnostics after an exception occurs, you can also use the alert feature of ARMS to create an alert for API operations. This ensures that the O&M team receives a notification in real time after an exception occurs.

To create an alert, see Create ARMS alerts.

References

- Create an application monitoring job
- API monitoring
- Trace query
- Analyze errors in code by using ARMS thread profiling
- Create ARMS alerts

4.6. Associate trace IDs with business logs

You can associate trace IDs with the business logs of an application. In this way, when an error occurs to the application, you can access the business logs associated with trace IDs to find out and troubleshoot the error.

Prerequisites

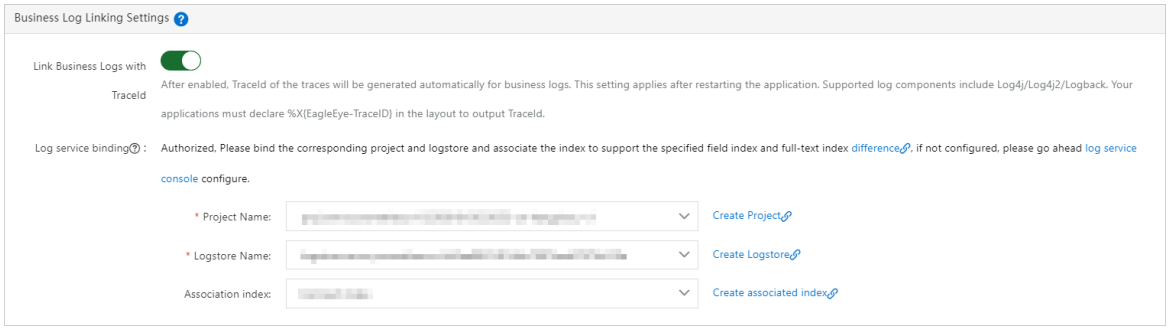
The Application Real-Time Monitoring Service (ARMS) agent is upgraded to version 2.6.1.2 or later. For more information, see FAQ about updating the ARMS agent for Java applications.

Context

In ARMS, trace IDs can be associated with business logs of an application based on the Mapped Diagnostic Context (MDC) mechanism. The Log4j, Log4j 2, and Logback mainstream log frameworks are supported.

Associate trace IDs with business logs

1. Log on to the ARMS console .
2. In the left-side navigation pane, choose **Application Monitoring > Applications**. In the top navigation bar, select a region. On the **Applications** page, click the name of the application.
3. In the left-side navigation pane, click **Application Settings**. On the page that appears, click the **Custom Configuration** tab.
4. On the **Custom Configuration** tab, turn on **Link Business Logs with TraceId** in the **Business Log Linking Settings** section.



> **Note**
>
> - If Link Business Logs with TraceId is turned on, trace IDs are automatically generated in the business logs.

5. Add `%X{EagleEye-TraceID}` to the pattern property of the business log layout. The following figure shows how to add this configuration for the Logback component.

> **Note** For information about how to obtain {EagleEye-TraceID} from the business code, see ARMS SDK.

```
<encoder class="ch.qos.logback.classic.encoder.PatternLayoutEncoder">
    <charset>UTF-8</charset>
    <pattern>%d{yyyy-MM-dd HH:mm:ss.SSS} [%X{EagleEye-TraceID}] [%thread] %-5level %logger{50} - %msg%n</pattern>
</encoder>
```

6. Restart the application.
   If trace IDs are displayed in the business logs of the application, the business logs are associated with the trace IDs, as shown in the following figure.

4.7. Connect applications in a private cloud to ARMS

This topic shows you how to use Gateway to connect applications in a private cloud to Application Real-Time Monitoring Service (ARMS) on Alibaba Cloud.

Scenario

ARMS on Alibaba Cloud cannot directly monitor applications in a private cloud due to a network connection issue. In this case, you can deploy Gateway on the classic network or a virtual private cloud (VPC) of Alibaba Cloud and use Gateway as a proxy to connect the applications to ARMS.

Deployment principle

1. ARMS Gateway cluster A deployed in the DMZ exposes the internal endpoint to Apsara Stack and the private cloud. The ARMS agents deployed in Apsara Stack and WebLogic send the collected monitoring data to the Gateway cluster.
2. The DMZ is connected to Alibaba Cloud by using a leased line. ARMS Gateway cluster B deployed in a VPC on Alibaba Cloud is connected to ARMS Gateway cluster A by using the leased line and acts as a bridge.
3. ARMS Gateway cluster B sends the collected monitoring data to the ARMS server on Alibaba Cloud.
4. ARMS agents installed on Alibaba Cloud, including frontend agents deployed on pages, send data to the ARMS server on Alibaba Cloud.

Requirements on hybrid cloud deployment of ARMS

- Assume that you need to collect data from 500 nodes in Apsara Stack and the private cloud. You must prepare six virtual machines to deploy two Gateway clusters. Each virtual machine is allocated 2 CPU cores and 8 GB memory. Each Gateway cluster contains three virtual machines on which Gateway is deployed.
- The monitoring data that is sent from Gateway to Alibaba Cloud is small in size and occupies less than 1 Mbit/s bandwidth. You can use the bandwidth of the existing leased line and install ARMS agents in Enterprise Distributed Application Service (EDAS) of Apsara Stack and WebLogic to monitor applications, without the need to modify application code.

Connect an application to ARMS

1. Download the Gateway package. In this example, Gateway connects to the ARMS service in the China (Hangzhou) region.
2. Deploy Gateway on a proxy server and run the following command to start Gateway:

   ```
   java -jar arms-gateway-1.7.0.jar
   ```

   > **Note** The version of Java Development Kit (JDK) must be later than JDK 1.7.

   By default, Gateway connects to the ARMS service in the China (Hangzhou) region. To connect Gateway to the ARMS service in another region, you can specify the region by using the -D parameter in the following way:

   ```
   java -jar -Darms.server.endpoint=arms-dc-bj.aliyuncs.com arms-gateway-1.7.0.jar
   ```

- China (Hangzhou): arms-dc-hz.aliyuncs.com
- China (Beijing): arms-dc-bj.aliyuncs.com
- China (Shanghai): arms-dc-sh.aliyuncs.com
- China (Qingdao): arms-dc-qd.aliyuncs.com
- China (Shenzhen): arms-dc-sz.aliyuncs.com

3. Download the agent.
   You can log on to the ARMS console and download the agent on the **Add Application** page. For more information, see Manually install the ARMS agent for a Java application.
4. Decompress the agent installation package.
5. In the arms-agent.config file, change the value of the profiler.collector.ip parameter to the IP address of the proxy server.

   ```
   profiler.collector.ip={IP address of the proxy server}
   ```

6. Start the application. In the ARMS console, choose **Application Monitoring > Applications** in the left-side navigation pane. Check whether monitoring data is reported. If monitoring data is reported, the application is connected to ARMS.

4.8. Identify business exceptions by analyzing traces and logs

The difficulty and low efficiency in identifying business exceptions have always been performance bottlenecks of the application monitoring feature of Application Real-Time Monitoring Service (ARMS). However, you can use the application monitoring feature of ARMS together with traces and logs to efficiently and accurately identify business exceptions. This improves the efficiency of development and diagnostics in a microservices framework.

Prerequisites

- Log Service is activated. Log on to the Log Service console and activate Log Service by following the on-screen instructions.
- A project is created. For more information, see Create a project.
- A Logstore is created. For more information, see Create a Logstore.

Context

Before you identify business exceptions by analyzing traces and logs, you must understand the following terms: metric, tracing, and logging.

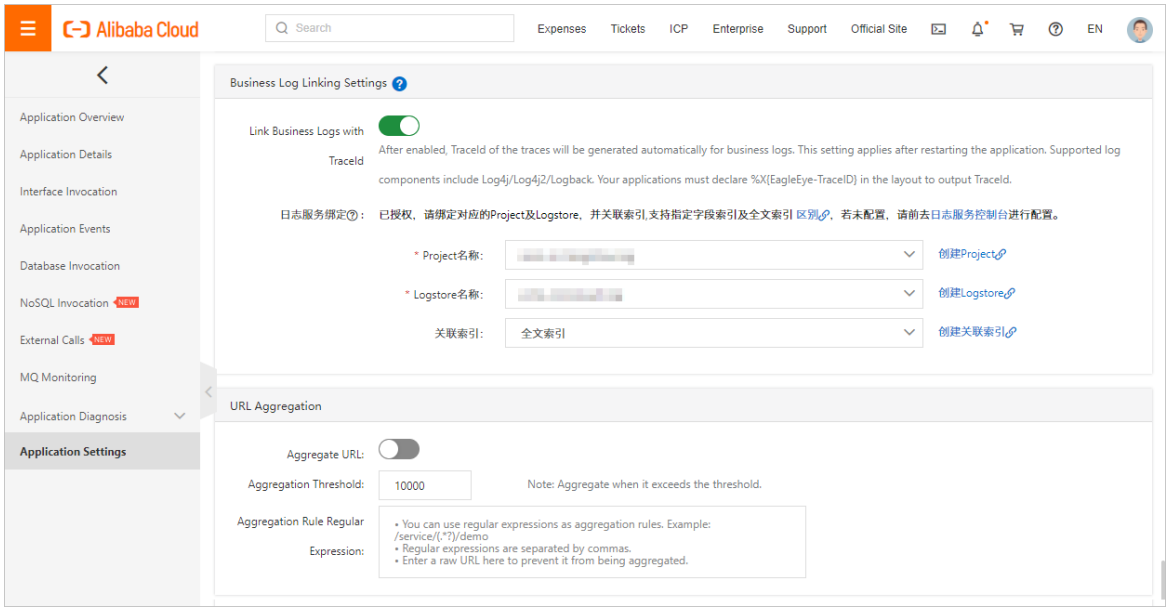
- Metric: The key metrics of an application include Application Service Request, Application Service Average Response Time, and Application Dependent Service Request.
- Tracing: All activities of an application, such as API calls and responses, are recorded in traces.
- Logging: All activities of an application, such as API calls and responses, are recorded in business logs.

When a business exception occurs, the statistical chart for an application metric shows obvious fluctuations. You can roughly analyze the business exception based on the chart. You can also analyze the complete traces and business logs to accurately identify the business exception.

Associate business logs with trace IDs

1. Log on to the ARMS console .

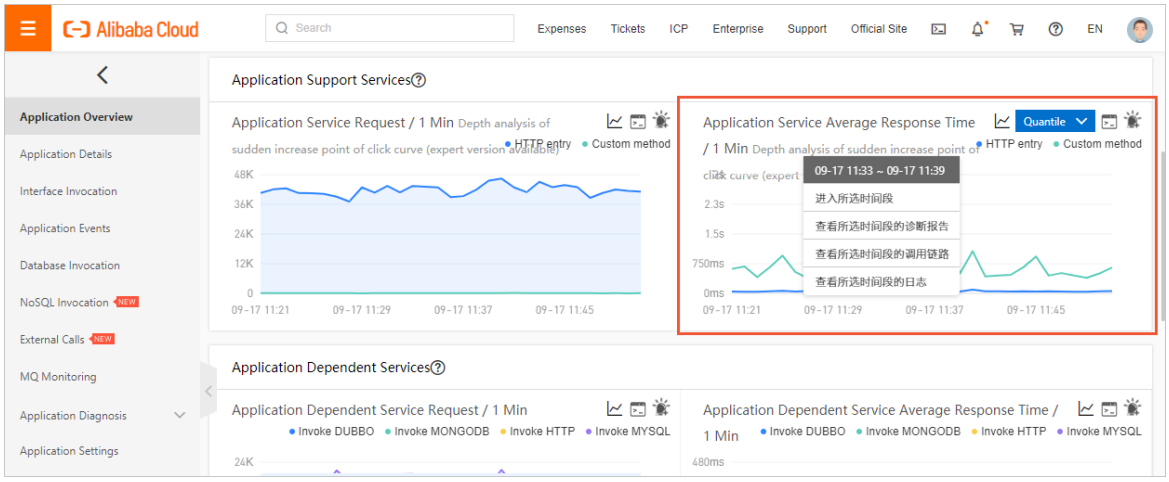
2. In the left-side navigation pane, choose **Application Monitoring > Applications**. In the top navigation bar, select a region. On the **Applications** page, click the name of the application.
3. In the left-side navigation pane, click **Application Settings**. On the page that appears, click the **Custom Configuration** tab.
4. On the **Custom Configuration** tab, turn on **Link Business Logs with TraceId** in the **Business Log Linking Settings** section. Then, specify the project and Logstore that store the business logs to be associated with trace IDs.



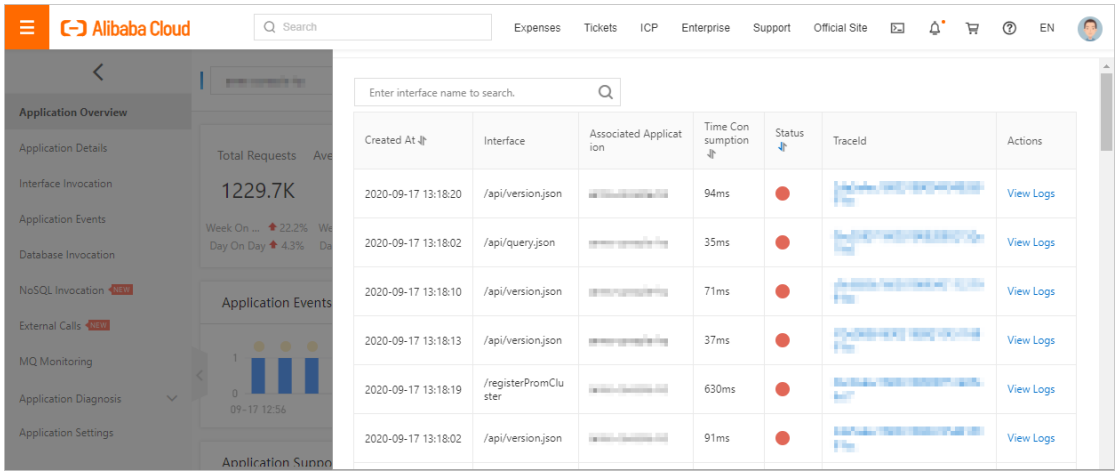
5. On the **Custom Configuration** tab, click **Save** in the lower-left corner.

Troubleshoot business exceptions from the perspective of application metrics

1. Log on to the ARMS console .
2. In the left-side navigation pane, choose **Application Monitoring > Applications**. In the top navigation bar, select a region. On the **Applications** page, click the name of the application.
3. In the left-side navigation pane, click **Application Overview**. On the page that appears, click the **Overview** tab in the upper part and select or set a time range to query in the upper-right corner.The **Overview** tab displays key metrics of the application, including **Application Service Request**, **Application Service Average Response Time**, and **Application Dependent Service Request**.
4. On the **Overview** tab, drag-select a time range on the chart for an application metric.In this example, the **Application Service Average Response Time** metric is used.



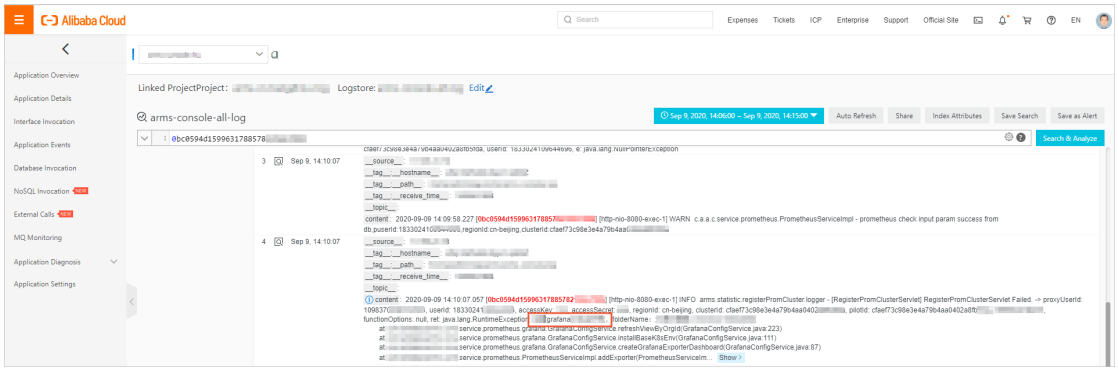
5. View the traces that were generated in the time range selected in Step 4.
   i. Click **View CallChain selected time**.
   ii. In the panel that appears, find the trace record whose status is and click the trace ID in the **TraceId** column.●You can also click **View Logs** in the **Actions** column for the trace record to view the business logs that were generated at the specified time point. Then, you can analyze the cause of the business exception.



   iii. Click the **Traces** tab. In the **Method Stack** column, click the icon.
   iv. On the trace details page, find the error message. You can move the pointer over the error message to view the exception cause.

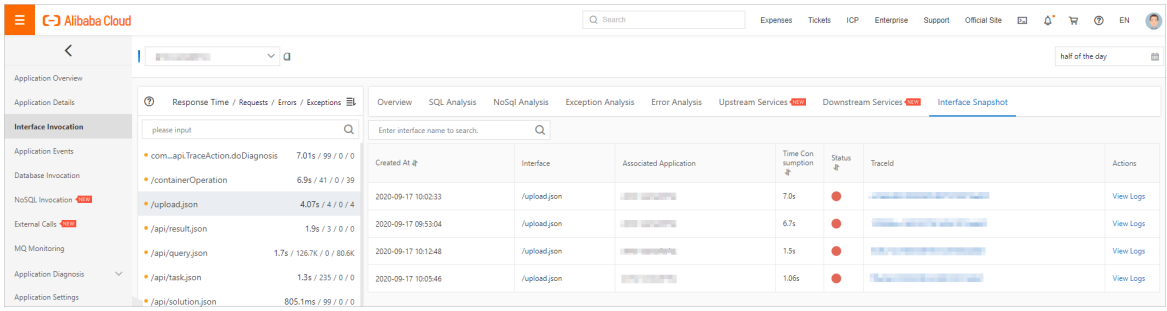
6. View the business logs that were generated in the time range selected in Step 4.
   i. Click **View Log selected time**.

ii. On the log analysis page, find the error message of the business exception and identify the cause of the business exception.

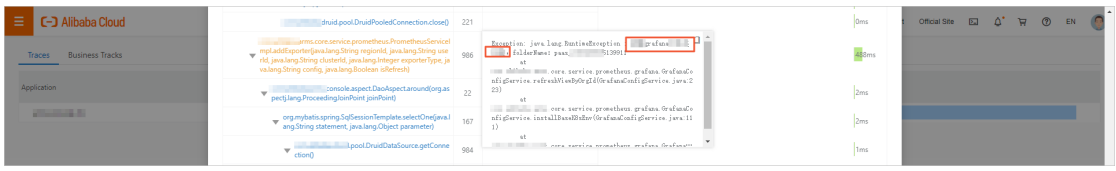


Troubleshoot business exceptions from the perspective of API calls

1. Log on to the ARMS console .
2. In the left-side navigation pane, choose **Application Monitoring > Applications**. In the top navigation bar, select a region. On the **Applications** page, click the name of the application.
3. In the left-side navigation pane, click **Interface Invocation**.
4. On the page that appears, click the API operation that you want in the API operation list and click the **Interface Snapshot** tab on the right.
5. On the **Interface Snapshot** tab, find the API call record whose status is .

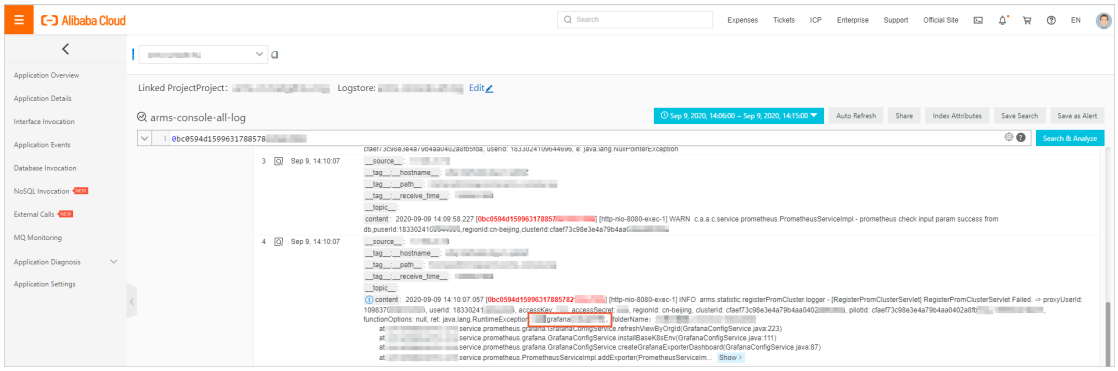


6. View the trace for the API call.
   i. Click the trace ID in the **TraceId** column for the API call record.
   ii. Click the **Traces** tab. In the **Method Stack** column, click the icon.
   iii. On the trace details page, find the error message. You can move the pointer over the error message to view the exception cause.



7. View the logs for the API call.
   i. Click **View Logs** in the **Actions** column for the API call record.

ii. On the log analysis page, find the error message of the business exception and identify the cause of the business exception.



4.9. Embed ARMS console pages in user-created web applications

You can embed Application Real-Time Monitoring Service (ARMS) console pages in user-created web applications. This way, you can view the pages from the applications without the need to switch between systems or to log on to the ARMS console.

Context

The operation to embed ARMS console pages into user-created web applications brings the following benefits:

- Allows you to log on to your own system and browse the application list, application details, and call query pages of the embedded ARMS console pages.
- Hides the top navigation bar and left-side navigation pane of the ARMS console. For more information, see Hide navigation pages.
- Uses Resource Access Management (RAM) to manage permissions on the ARMS console. For example, you can change the full permissions to read-only permissions. For more information, see Grant different permissions to RAM users.

Sample code

To embed ARMS console pages into a user-created web application, download and use the sample code.

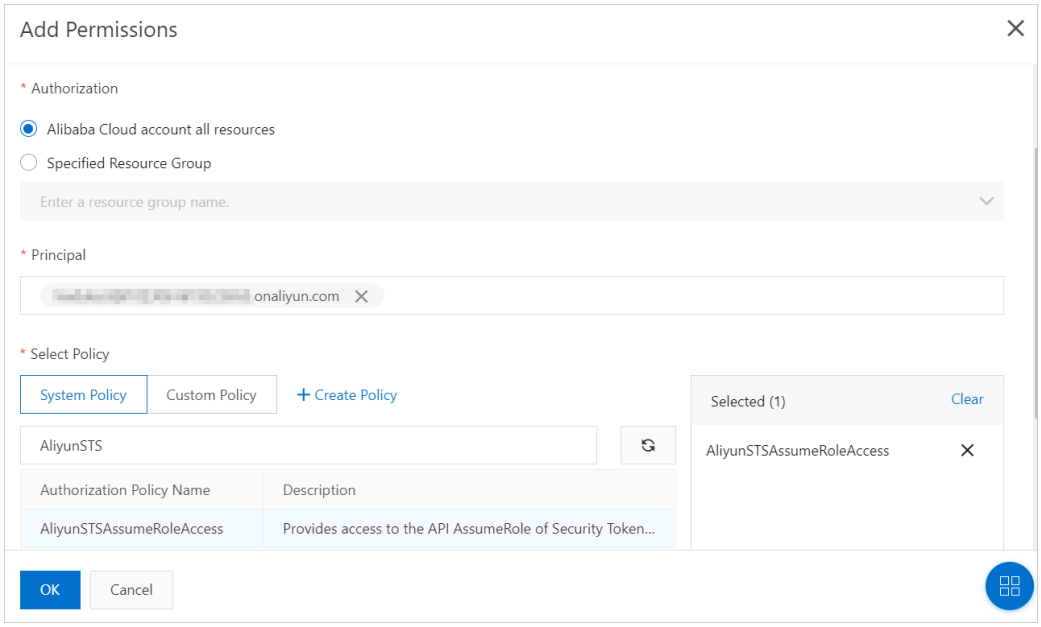
Step 1: Create a RAM user and grant it permissions

Use your Alibaba Cloud account to create RAM users and grant them permissions to call Security Token Service (STS) to assume RAM roles.

1. Log on to the RAM console.
2. In the left-side navigation pane, choose **Identities > Users**.
3. On the **Users** page, click **Create User**.
4. On the **Create User** page, set **Logon Name** and **Display Name** in the **User Account Information** section, select **Programmatic Access** in the **Access Mode** section, and then click **OK**.

> **Notice** RAM automatically generates an AccessKey pair for the RAM user. Then, the RAM user can access ARMS by calling the corresponding API operations. For security reasons, the RAM console allows you to view or download the AccessKey secret only once. Therefore, when you create an AccessKey pair, you must keep your AccessKey secret strictly confidential.

5. In the **Verify by Phone Number** dialog box, click **Get Verification Code**, enter the verification code sent to your mobile phone, and then click **OK**.
6. On the **Users** page, find the created RAM user and click **Add Permissions** in the **Actions** column.
7. In the **Select Policy** section of the **Add Permissions** panel, enter AliyunSTSAssumeRoleAccess in the search box. Click the displayed permission policy to add it to the **Selected** list on the right side. Then, click **OK**.



8. In the **Add Permissions** panel, view the authorization information summary in the **Authorization** section and click **Complete**.

Step 2: Create a RAM role and grant it permissions

Create a RAM role and grant it permissions to access the ARMS console. Then, the RAM user assumes the RAM role to access the ARMS console.

1. Log on to the RAM console.
2. In the left-side navigation pane, click **RAM Roles**.
3. On the **RAM Roles** page, click **Create RAM Role**.
4. In the **Create RAM Role** panel, perform the following operations:
   i. In the **Select Role Type** step, select **Alibaba Cloud Account** for **Trusted entity type** and click **Next**.
   ii. In the **Configure Role** step, enter a role name in the **RAM Role Name** field and click **OK**.
   iii. In the **Finish** step click **Add Permissions to RAM Role**.
5. In the **Select Policy** section of the **Add Permissions** panel, click System Policy or Custom Policy and enter the keyword of the policy that you want to add in the search box. Click the displayed policy to add it to the **Selected** list on the right side. Then, click **OK**.You can grant the following

ARMS permissions to a RAM role as needed:

- AliyunARMSFullAccess: the full access permissions on ARMS
- AliyunARMSReadOnlyAccess: the read-only permissions on ARMS

6. In the **Add Permissions** panel, view the authorization information summary in the **Authorization** section and click **Complete**.

Step 3: Obtain the temporary AccessKey pair and STS token

Log on to the user-created web application, and then call the AssumeRole operation of STS on the web server to obtain the temporary AccessKey pair and STS token. They are the temporary identity. For more information about the AssumeRole operation, see AssumeRole.

You can call the AssumeRole operation by using one of the following methods:

- Use OpenAPI Explorer.
- Use SDK for Java.

SDK for Java is used in the example.

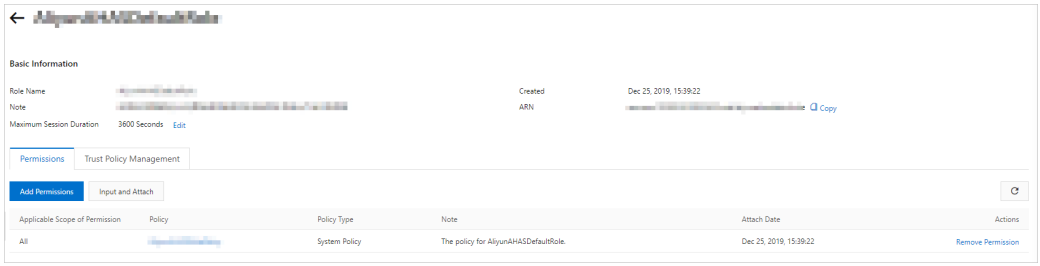
Set the following parameters when you use SDK for Java:

```
String accessId = "<yourAccessKeyId>"; // The AccessKey ID of the RAM user.
String accessKey = "<yourAccessKeySecret>"; // The AccessKey secret of the RAM user.
String roleArn = "<roleArn>"; // The Alibaba Cloud Resource Name (ARN) of the RAM role.
```

The AccessKey ID and AccessKey secret of the RAM user are obtained when the RAM user is created.

Perform the following steps to obtain the ARN of the RAM role:

1. Log on to the RAM console.
2. In the left-side navigation pane, click **RAM Roles**.
3. In the lower part of the **RAM Roles** page, click the name of the RAM role whose ARN you want to obtain.
4. On the page that appears, copy the value of **ARN** in the **Basic Information** section.



Step 2: Obtain the logon token

After you call the AssumeRole operation of STS to obtain the temporary AccessKey pair and STS token, call the GetSigninToken operation to obtain the logon token.

> **Notice** The temporary STS token may contain special characters. Before you use the token, use the URL encoding method to encode the special characters.

Sample request

```
http://signin.aliyun.com/federation?Action=GetSigninToken
  &AccessKeyId=<The temporary AccessKey ID that is returned by STS>
  &AccessKeySecret=<The temporary AccessKey secret that is returned by STS>
  &SecurityToken=<The temporary token that is returned by STS>
  &TicketType=mini
```

Step 3: Generate a logon-free URL

Use the obtained logon token and URL of the ARMS console page that you want to embed to generate a logon-free URL. This allows you to access the ARMS console page from your user-created web application.

> **Note** The temporary token is valid for 3 hours. We recommend that you configure the URL in the user-created web application to generate a new logon token on each request.

1. In the ARMS console, obtain the URL of the console page that you want to embed.

   The following example is the URL of the **Applications** page for the China (Hangzhou) region:

   ```
   https://arms.console.aliyun.com/apm?iframeMode=true&demo=1&pid=ac346dab-419d-48f5-b06a-e1c331c5c93e&regionId=cn-shanghai#/ac346dab-419d-48f5-b06a-e1c331c5c93e/home
   ```

   > **Note**
   > - The URL must be the console address of ARMS Application Monitoring. ARMS Browser Monitoring is not supported.
   > - To hide the top navigation bar and the left-side navigation pane of the ARMS console, set `iframeMode` to true in the search section of the URL.

2. Use the logon token and the URL of the ARMS console page to generate a logon-free URL for the page.

   Sample request

   ```
   http://signin.aliyun.com/federation?Action=Login
     &LoginUrl=<A URL that returns HTTP status code 302 and redirects you to the user-created website>
     &Destination=<The URL of the ARMS console page>
     &SigninToken=<The obtained logon token>
   ```

5.References

5.1. Java components and frameworks supported by ARMS

This topic describes third-party Java components and frameworks supported by Application Real-Time Monitoring Service (ARMS). If the components or frameworks used by the application that you want to monitor are not supported by ARMS, you must configure a universal Filter interceptor to collect monitoring data.

Java components and frameworks supported by ARMS

Component	JDK 1.7	JDK 1.8
Dubbo	2.5.X+	2.5.X+
Feign	Not supported	9.X+
Google HTTP Client	1.10.X+	1.10.X+
GRPC-Java	1.15+	1.15+
HttpClient 3	3.X+	3.X+
HttpClient 4	4.X+	4.X+
Hystrix	1.5.X+	1.5.X+
JDK HTTP	1.7.X+	1.7.X+
Jetty	8.X+	8.X+
Lettuce	4.0+	4.0+
MariaDB	1.3+	1.3+
MemCached	2.8+	2.8+
MongoDB	3.7+	3.7+
MyBatis	3.X+	3.X+
MySQL JDBC	5.0.X+	5.0.X+
OKHttp	2.X+	2.X+
Oracle JDBC	10.2.X+	10.2.X+
PostgreSql JDBC	9.4+	9.4+
Reactor	Not supported	3.X+

Component	JDK 1.7	JDK 1.8
Reactor Netty	Not supported	0.9+
Redis	2.X+	2.X+
Resin	3.0+	3.0+
RxJava	2.X+	2.X+
Spring	4.X+	4.X+
Spring Boot	1.3.X+	1.3.X+
Spring Cloud Gateway	Not supported	5.0.0.RELEASE+
Spring WebFlux	Not supported	5.0.0.RELEASE+5.0.0.RELEASE+
SQLServer JDBC	6.4+	6.4+
Thrift	0.8+	0.8+
Tomcat	7.X+	7.X+
Undertow	1.3X+	1.3X+
WebLogic	12.X+	12.X+

Configure a universal Filter interceptor to collect data

If the components or frameworks of an application are not supported by ARMS, you can configure a universal Filter interceptor to collect data. Perform the following operations:

1. Import *arms-sdk-1.7.1.jar* to the pom.xml file.

```
<dependency>
  <groupId>com.alibaba.arms.apm</groupId>
  <artifactId>arms-sdk</artifactId>
  <version>1.7.1</version>
</dependency>
```

> **Note** If you cannot retrieve pom.xml, download arms-sdk-1.7.1.jar.

2. Configure the ARMS Filter interceptor in *web.xml*.

```
<filter>
  <filter-name>EagleEyeFilter</filter-name>
  <filter-class>com.alibaba.arms.filter.EagleEyeFilter</filter-class>
</filter>
<filter-mapping>
  <filter-name>EagleEyeFilter</filter-name>
  <url-pattern>/*</url-pattern>
</filter-mapping>
```

3. Log on to the ARMS console .
4. Install the ARMS agent for the Java application. For more information, see Manually install the ARMS agent for Java applications.
5. Restart the application for the configuration to take effect.

5.2. PHP components and frameworks supported by ARMS Application Monitoring

This topic lists the third-party PHP components and frameworks supported by Application Real-Time Monitoring Service (ARMS) Application Monitoring.

PHP components and frameworks supported by ARMS Application Monitoring

Item	Version
PHP version	PHP 5.4, 5.5, 5.6, 7.0, 7.1, and 7.2 NTS
Nginx	php-fpm
Apache	apache2handler
Runtime environment of the ARMS agent for PHP applications	Glibc-2.12 and later versions

5.3. Versions of the ARMS agents

This topic describes the version history of the ARMS agents for Java and PHP.

Versions of the ARMS agent for Java

Version	Release date	Revision
2.7.1.1	August 14, 2020	• NoSQL monitoring is supported. • Routing of microservice tags is supported. • Compression for N + 1 invocations is supported. • The network connection issue of Finance Cloud is fixed and the memory usage is optimized.

Version	Release date	Revision
2.7.1	July 16, 2020	• The latest version of the Jedis plug-in is supported to solve the issue that topology maps are not recognized by ApsaraDB for Redis clusters.
2.7.0	May 20, 2020	• Subfeatures of microservices are supported.
2.6.2	May 20, 2020	• Business monitoring is supported.
2.6.1.2	March 19, 2020	• Microservice authentication is supported. • Graceful disconnection for microservices is supported.
2.6.1.1	March 16, 2020	• Components such as Spring Cloud Gateway and Spring Webflux are supported.
2.6.1	February 14, 2020	• Features such as obtaining microservice metadata are supported.
2.6.0.2	January 2, 2020	• Exception analysis of the new version is supported. • The Thrift plug-in issue is fixed.
2.6.0	December 17, 2019	• Asynchronous trace is supported. • The invocation parameters of Dubbo and HSFProvider are recorded. • Several existing plug-in issues are fixed.
2.5.9.5	November 28, 2019	• The jfinal-undertow plug-in is supported. • Several bugs are fixed, such as the failure to obtain Dubbo thread profiling data.

Version	Release date	Revision
2.5.9.3	November 25, 2019	• Tracing services are integrated into ARMS. • Several bugs are fixed and the agent performance is optimized.
2.5.9	September 6, 2019	• The denial of service (DoS) vulnerability of FastJson is fixed. • The logic that is used to obtain the IP address of network interface controller is modified.
2.5.8	August 2, 2019	• The dual-state alert feature is supported. This feature is used to configure alert rules for metrics with the only two states: yes or no. • Chinese DM database plug-ins are supported.
2.5.7.2	July 30, 2019	• JVM metaspace metrics are supported. • HTTP status codes to be ignored can be customized. By default, status codes greater than 400 are counted as errors. You can also customize a threshold greater than 400. For more information, see [Related topic].
2.5.7	July 11, 2019	The FastJson version of dependencies is upgraded to eliminate security vulnerabilities.

Version	Release date	Revision
2.5.6.1	June 28, 2019	• Dubbo and MariaDB plug-ins are supported. • Bound SQL values can be obtained by custom configurations. The variable values bound to PrepareStatement can be captured. The variable values take effect without the need to restart the application. For more information, see [Related topic]. • Memory is optimized and several bugs are fixed. • Log4j log dependency is removed to avoid conflicts.
2.5.6	June 7, 2019	• Quantile statistics is supported. • Features of the ARMS agent are optimized and several bugs are fixed.
2.5.5	June 3, 2019	• HSF and HTTP calls are supported. • Features of the ARMS agent are optimized and several bugs are fixed.
2.5.3	March 15, 2019	• Thread metrics of application can be reported while the applications are running. • The Spring-Data-Redis plug-in is supported. • The Druid database connection pool plug-in is supported.

Version	Release date	Revision
2.5.2	February 21, 2019	• The number of file handles can be collected. • Instantaneous values for garbage collection (GC) time and for the number of GC operations can be reported. • The maximum length of request parameters can be customized. For more information, see [Related topic].
2.5.1	January 14, 2019	• Trace compression is supported. For more information, see [Related topic]. • Application monitoring jobs can be created without the need to use the ARMS console. • Features of the ARMS agent are optimized and several bugs are fixed.
2.5.0	December 28, 2018	• Agent can be connected without the need to restart the application. • Host monitoring is improved and the Windows system can be monitored. • Spring-webflux is supported. • Features of the ARMS agent are optimized and several bugs are fixed.
2.4.6	October 26, 2018	• The Google Remote Procedure Call (gRPC), Thrift, and XMemcached plug-ins are supported. • Topology views of API operation calls are supported. • Topology views that cover the frontend and backend are supported.

Version	Release date	Revision
2.4.5	September 17, 2018	• The Lettuce plug-in (JRE 1.8+) is supported. • The MongoDB plug-in is supported. • Exception details can be captured.
2.4.4	August 6, 2018	• Application thread profiling data can be reported. • Memcached caching is supported. • Exception filtering can be customized. For more information, see [Related topic].
2.4.3.1	June 29, 2018	• WebLogic servers are supported. • Undertow servers are supported. • Memory usage by the ARMS agent is optimized. • The time required to start and load the ARMS agent is shortened. • The problem that JVM monitoring and host monitoring metrics cannot be reported is eliminated.
2.4.3	May 18, 2018	• The monitoring metrics of Message Queue for RocketMQ (RocketMQ) can be captured. • Monitoring methods can be customized. • The problem of frequent log output in throttling scenarios is solved. • The maximum length of the method stack can be customized. For more information, see [Related topic]. • The sampling feature is optimized. Abnormal traces are excluded.

Version	Release date	Revision
2.4.2	April 19, 2018	• Custom configuration details can be read. • Trace information can be retrieved by using SDKs. • JVM metrics such as thread, number of GC operations, and duration of GC operations can be collected. • HSF calls can be monitored. • Host monitoring metrics related to CPU, memory, networks, and disks can be collected. • The problem that the **./shutdown.sh** process may be stuck in the Tomcat environment is eliminated.
2.4.1	March 24, 2018	• JVM monitoring, such as reporting of heap memory and non-heap memory is supported. • PlayFrameWork 1.4.4 is supported. • Parameters such as the sampling rate, agent switch, log level, and threshold can be customized. For more information, see [Related topic].
2.4.0	February 14, 2018	• The PostgreSQL database is supported. • ARMS can be connected with Alibaba Cloud Elastic Compute Service (ECS) instances in each region over the internal network. • ARMS application monitoring is available for commercial use.

Versions of the ARMS agent for PHP

Version	Release date	Revision

Version	Release date	Revision
2.0.3	August 19, 2019	• The Predis plug-in is supported. • Bugs of the Curl plug-in are fixed.
2.0.2	July 31, 2019	• Issues in the transmission of network modules under high concurrency are fixed. • The logic for transmission and re-connection is redesigned. • Memory usage is reduced. • Several bugs are fixed.
2.0.1	July 23, 2019	• The Arms-agent process is used as the daemon. • The Redis and MongoDB plug-ins are supported. • Several bugs are fixed.
2.0.0	July 5, 2019	• A new and more reliable network model is used. • The display of exception information is optimized. • The memory usage is optimized. • Several bugs are fixed.

Version	Release date	Revision
1.1.0	April 30, 2019	• The GCC 4.4.7 environment is supported. • The TCP connection heartbeat is introduced. • Host monitoring bugs are fixed. • **php -m** command can be run to show the ARMS version. • DNS is used to resolve the domain names of collectors. • Features of the ARMS agent are optimized and several bugs are fixed. **Notice** We recommend that you immediately upgrade the ARMS agent from version 1.x.x to 2.x.x.
1.0.1	March 15, 2019	• Laravel 5.x is supported. • The PDO plug-in is supported. • Unnecessary logs are removed. • Several bugs are fixed. **Notice** We recommend that you immediately upgrade the ARMS agent from version 1.x.x to 2.x.x.

5.4. Key statistical metrics

This topic describes the meanings of key statistical metrics on each page in application monitoring of Application Real-Time Monitoring Service (ARMS).

Terms

The following terms are used in this topic:

- Apdex
  Application Performance Index (Apdex) is an internationally accepted standard for evaluating application performance. In Apdex, the user experience of an application can be classified into three levels:
  - Satisfied (0 to T)
  - Tolerating (T to 4T)

- Frustrated (greater than 4T)

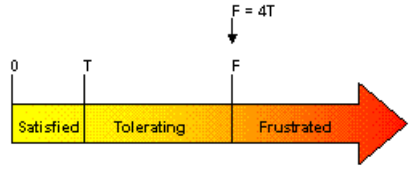
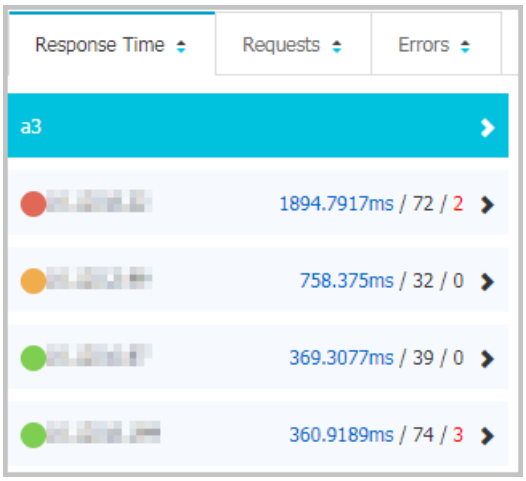


Image source: apdex.org
The following formula is used to calculate the Apdex score:

```
Apdex = (Satisfied samples + Tolerating samples/2)/Total samples
```

ARMS uses the average response time of an application in calculation, and defines T at 500 ms.

- instance
  An instance is a machine where the monitored application is deployed. The granularity of an instance is JVM. In the following figure, "a3" is an application, and each row under a3 is a machine where a3 is deployed. Each machine is an instance.



Statistics pages

- **Applications**
  In the left-side navigation pane, choose **Application Monitoring > Applications** to view the health rate, requests, errors, response time, exceptions, status, and response time in the last ten minutes of instances.
- **Application Overview**
  On the Applications page, click the name of an application to go to the **Application Overview** page. You can select menus in the upper part of the page to view statistics in other dimensions.
  - **Overview Analysis**
    - Services provided by the application: requests and average response time
    - Services the application depends on: requests, average response time, the number of instances, and HTTP status code
    - System information: CPU, memory, and load
    - Statistical analysis: slow call analysis, average response time, exception type, and times of occurrence

- **Topology Graph**
  - Application topology
  - Instance health status: Green indicates Normal, yellow indicates Alerting, and red indicates Severe.
  - Type of call:

Type of call	Description	Remarks
Local API call	Local API operation calls	API operation call
HTTP entry point	The entry point of the application called by the client by using HTTP	Service entry call
Dubbo call	Calls generated by Dubbo consumers	Service entry call
HSF call	Calls generated by HSF consumers	Service entry call
HTTP call	HTTP calls initiated by this application to other services	Inter-service call
HSF provision	Calls generated by HSF providers	Inter-service call
Dubbo provision	Calls generated by Dubbo providers	Inter-service call
MySQL call	Calls initiated for operating on MySQL	Database call
Oracle call	Calls initiated for operating on Oracle	Database call
Redis call	Calls initiated for operating on Redis	Database call

  - Instance IP: The IP addresses of all instances where the application is deployed.
  - The number of requests per minute, the response time, and the error rate of the application.
- **3D Topology**
  - QPS: the queries per second
  - RT(ms): the response time in milliseconds
  - ErrQps: the error queries per second

- **Application Details**
  This page shows details of calling the current application. Click different tabs to view the detailed analysis of different dimensions, such as instance response time, the number of requests, the number of errors, instance overview, SQL analysis, exception analysis, and interface snapshot.
- **Interface Invocation**

This page shows the statistical information of API operations provided by the current application. Click different tabs to view the detailed analysis of different dimensions, such as instance response time, the number of requests, the number of errors, instance overview, SQL analysis, exception analysis, and interface snapshot.

- **Database Invocation**
  This page shows application-related database invocation information. Click different tabs to view the detailed analysis of different dimensions, such as instance response time, the number of requests, the number of errors, instance overview, SQL analysis, and exception analysis.

Key statistical metrics on related tabs

- Response Time: The average response time of applications and instance calls, or the average execution response time of database operations
- Requests: The number of requests to call applications or instances, or the number of times database operations are performed
- Errors: The number of incorrect application or instance calls, or the number of abnormal executions in database operations
- **Overview**

Reported field	Description
Requests	The number of requests to call applications or instances, or the number of times database operations are performed
Response Time	The average response time of applications and instance calls, or the average execution response time of database operations
Error Rate	(The number of abnormal application or instance calls, or the number of abnormal executions in database operations)/Number of requests

- **SQL Analysis**

Reported field	Description
SQL Call Statistics	The column chart and the left-side Y axis show the number of database requests. The line chart and the right-side Y axis show the database response time.
Average Time Consumption	The average amount of time consumed for this database call
Number of Calls	The number of times this type of database is called

- **Exception Analysis**

Reported field	Description
Exception Statistics	The column chart shows the number of exceptions of the application, instance, and database.
Exception Type	The types of collected exceptions.
Exception Details	The detailed information of exceptions
Average Time Consumption	The average amount of time consumed by this incorrect call
Errors	The number of times this exception type has occurred

- **Interface Snapshot**

Reported field	Description
Elapsed Time	Amount of time consumed to call the API of an application or instance
Status	The return status of the API call of an application or instance. Green indicates a success response, and red indicates an exception.
TraceId	The index ID of an application or instance call. You can click the ID to go to the details page of this trace.

5.5. ARMS SDK

This topic describes how to use Application Real-Time Monitoring Service (ARMS) SDK to dynamically obtain TraceId and its properties in the service code.

Prerequisites

- An application monitoring task is created in the ARMS console, and the ARMS agent for application monitoring is installed and started in the Java program. For more information, see Manually install the ARMS agent for a Java application.
- arms-sdk-1.7.3.jar is introduced to the program.

```
<dependency>
  <groupId>com.alibaba.arms.apm</groupId>
  <artifactId>arms-sdk</artifactId>
  <version>1.7.3</version>
</dependency>
```

> **Note** If you cannot obtain the pom.xml file, download arms-sdk-1.7.3.jar.

Obtain TraceId and RpcId

You can run the following code to obtain TraceId and RpcId:

```
Span span = Tracer.builder().getSpan();
String traceId = span.getTraceId();
String rpcId = span.getRpcId();
```

Pass through a custom tag baggage

To pass through a custom tag, you must add and obtain the tag from the service code. The following section describes the procedure:

1. Add the baggage tag to the service code.

```
Map<String, String> baggage = new HashMap<String, String>();
baggage.put("key-01", "value-01");
baggage.put("key-02", "value-02");
baggage.put("key-03", "value-03");
Span span = Tracer.builder().getSpan();
span.withBaggage(baggage);
```

2. Obtain the baggage tag from the service code.

```
Span span = Tracer.builder().getSpan();
Map<String, String> baggage = span.baggageItems();
```

Set a custom tag tag for a span

A custom tag for a span applies to only the current span and is not passed to other spans. You must add and obtain the tag from the service code. The following section describes the procedure:

1. Add a custom tag tag for a span in the service code. Multiple tags can be added.

```
Span span = Tracer.builder().getSpan();
// Add a tag to the Span.
span.setTag("tag-key1", "tag-value1");
span.setTag("tag-key2", "tag-value2");
```

2. Obtain the tag from the service code.

```
Span span = Tracer.builder().getSpan();
// Inspect the Span's tags.
Map<String, String> tags = span.tags();
```

Query traces based on custom tags baggage and tag

The span tags baggage and tag can be used to query traces by tag.

- Items on a baggage can be passed to the downstream and are generally used to color business data. We recommend that you do not set a large number of tag items.
- Items on a tag apply to only the current span. You can set multiple items.

1. Log on to the ARMS console.
2. In the left-side navigation pane, choose **Application Monitoring > Invocation Trace Query** and select a region in the top navigation bar.
3. On the **Invocation Trace Query** page, select a **Parameter Name**, specify the custom tag in the **Parameter Value** field, and click **Query**.
4. In the **Invocation Trace Query** list, click the **TraceID** of the destination trace.

5. On the **Invocation Trace** page, move the pointer over **Service Name**. The information of tags that belong to the current span is displayed. baggage items are automatically added to tags of each span.

6.Update the ARMS agent for Java applications

This topic describes how to update the ARMS agent for Java applications in Enterprise Distributed Application Service (EDAS), Container Service, and other environments.

How do I update the ARMS agent for a Java application in EDAS?

To update the ARMS agent for a Java application in EDAS, redeploy the application.

How do I update the ARMS agent for a Java application in Container Service?

To update the ARMS agent for a Java application in Container Service, restart the pod of the application.

How do I update the ARMS agent for other Java applications than those in EDAS and Container Service?

To update the ARMS agent for other Java applications than those in EDAS and Container Service, uninstall the agent and then install it again.

For more information about how to uninstall the agent, see FAQ about uninstalling the ARMS agent.

7.FAQ

This topic provides answers to commonly asked questions about the application monitoring feature of Application Real-time Management Service (ARMS).

Overview

- FAQ about manually installing the ARMS agent for Java applications
  - Is the ARMS agent compatible with the agents of other Application Performance Management (APM) products, such as Pinpoint?
  - What do I do if OutOfMemoryError is reported when I start an application after the ARMS agent is installed?
  - How do I test network connectivity?
  - How do I check whether the ARMS agent is successfully installed?
  - Why is no monitoring data displayed in the ARMS console after I install an ARMS agent?
  - What do I do if no IP address or an incorrect IP address is displayed after the ARMS agent is installed?
  - How do I troubleshoot common errors contained in the log files of the ARMS agent stored in the ArmsAgent/log folder?
  - How do I deploy multiple instances on a single machine?
- FAQ about installing the ARMS agent for Java applications with one click
  - What do I do when getcwd errors are reported when I run the script to access a Java application?
  - Where do I view the logs after I install the ARMS agent with one click?
- FAQ about installing the ARMS agent for Java applications deployed on ECS instances with one click
  - What do I do if the ARMS agent cannot be installed?
  - What do I do if the information about the processes of the ECS instance is inaccurate after the ARMS agent is installed?
  - What do I do if I cannot enable ARMS application monitoring for a process on an ECS instance?
- FAQ about installing the ARMS agent for Java applications in Container Service for Kubernetes (ACK) clusters
  - Why is there no data displayed in Application Monitoring after the ARMS agent is installed on a Java application in an ACK cluster?
- FAQ about installing the ARMS agent for Java applications in open-source Kubernetes environments
  - What do I do if the application cannot be started?
  - How do I view the logs of the ARMS agent?
- FAQ about modifying the names of Java applications without reinstalling the ARMS agent
  - How do I modify the name of a common Java application on which the ARMS agent is manually installed?
  - How do I modify the name of a Java application deployed in an ACK cluster?
  - How do I modify the name of a Java application deployed in Enterprise Distributed Application Service (EDAS)?
- FAQ about uninstalling the ARMS agent

- How do I uninstall the ARMS agent that is manually installed?
- How do I uninstall the ARMS agent that is installed with one click?
- How do I uninstall the ARMS agent installed on a Java application deployed on an ECS instance?
- How do I uninstall the ARMS agent installed on a Java application in an ACK cluster?
- How do I uninstall the ARMS agent installed on a Java application in an open-source Kubernetes environment?
- How do I uninstall the ARMS agent installed on a Java application in Docker?
- How do I uninstall the ARMS agent installed on a PHP application?
- How do I uninstall the ARMS agent installed on a PHP application in an ACK cluster?

- Other FAQ
  - What do I do if the data of my application with the OpenFeign component is incomplete in ARMS?

Is the ARMS agent compatible with the agents of other Application Performance Management (APM) products, such as Pinpoint?

The ARMS agent is incompatible with the agents of other APM products. APM is implemented using bytecode instrumentation based on the ASM framework. If you install two agents, bytecode instrumentation is performed twice on the code. Agents developed by different vendors use different code to implement bytecode instrumentation. Therefore, if you install multiple agents, performance issues may occur due to code conflicts. We recommend that you do not install the agents of other APM products.

[Back to the top]

What do I do if OutOfMemoryError is reported when I start an application after the ARMS agent is installed?

Add the corresponding heap memory parameters to the start command to increase the memory of the JVM. In the following example, the initial value of heap memory size (Xms) is 512 MB and the maximum value of heap memory size (Xmx) is 2 GB.

> **Note** Adjust the values based on your actual requirement. In other environments such as Tomcat, add this parameter to JAVA_OPTS of the configuration file.

```
-Xms512M
-Xmx2048M
```

If the `OutOfMemoryError: PermGen space` error is reported, add the following parameters to the start command:

```
-XX:PermSize=256M
-XX:MaxPermSize=512M
```

If the `OutOfMemoryError: metaspace` error is reported, add the following parameters to the start command:

```
-XX:MetaspaceSize=256M
-XX:MaxMetaspaceSize=512M
```

[Back to the top]

How do I test network connectivity?

Before you install the ARMS agent, make sure that the 8883, 8443, and 8442 ports can be accessed. You can run the **Telnet** command to check whether the target host is connected to the ARMS server network. For example, to test the connectivity to the China (Shenzhen) region, log on to the host on which the application is deployed and run the following commands:

```
telnet arms-dc-sz.aliyuncs.com 8883
telnet arms-dc-sz.aliyuncs.com 8443
telnet arms-dc-sz.aliyuncs.com 8442
```

Note Replace the domain names in the preceding commands with the actual endpoints of ARMS application monitoring. Note that the endpoint of ARMS application monitoring varies depending on if you access it from classic networks and the Internet or VPCs.

Endpoints of ARMS application monitoring

Region	Endpoint for access from classic networks and the Internet	Endpoint for access from VPC
China (Hangzhou)	arms-dc-hz.aliyuncs.com	arms-dc-hz-internal.aliyuncs.com
China (Beijing)	arms-dc-bj.aliyuncs.com	arms-dc-bj-internal.aliyuncs.com
China (Shanghai)	arms-dc-sh.aliyuncs.com	arms-dc-sh-internal.aliyuncs.com
China (Qingdao)	arms-dc-qd.aliyuncs.com	arms-dc-qd-internal.aliyuncs.com
China (Shenzhen)	arms-dc-sz.aliyuncs.com	arms-dc-sz-internal.aliyuncs.com
China (Zhangjiakou)	arms-dc-zb.aliyuncs.com	arms-dc-zb-internal.aliyuncs.com
China (Hong Kong)	arms-dc-hk.aliyuncs.com	arms-dc-hk-internal.aliyuncs.com
Singapore	arms-dc-sg.aliyuncs.com	arms-dc-sg-internal.aliyuncs.com
Regions for Alibaba GovCloud	arms-dc-gov.aliyuncs.com	arms-dc-gov-internal.aliyuncs.com
Finance Cloud of China (Hangzhou)	arms-dc-hz-finance.aliyuncs.com	arms-dc-hz-finance-internal.aliyuncs.com

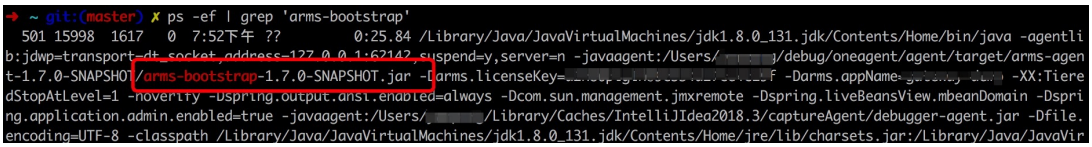
[Back to the top]

How do I check whether the ARMS agent is successfully installed?

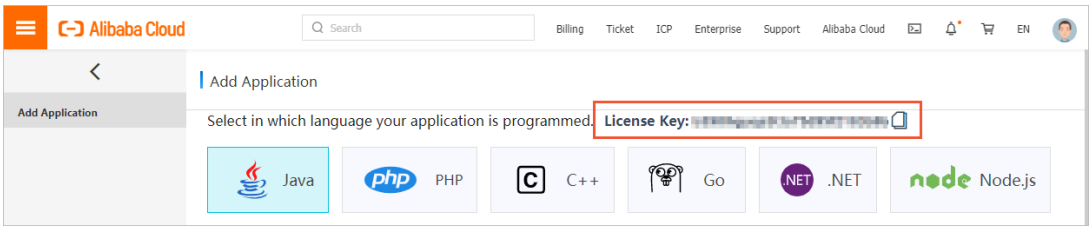
Run the following **ps** command to check whether the ARMS agent is successfully installed based on parameters in the start command.

```
ps -ef | grep 'arms-bootstrap'
```

The results shown in the following figure indicates that the ARMS agent is successfully installed.



The value of Darms.licenseKey in the command must be the same as the license key value displayed on the **Add Application** page in the ARMS console.



[Back to the top]

Why is no monitoring data displayed in the ARMS console after I install an ARMS agent?

1. If the log of the ARMS agent contains `send agent metrics. no metrics.`, check whether your application is continuously accessed by external requests, including HTTP requests, HSF requests, and Dubbo requests and whether the development framework is supported by the ARMS agent. For more information about third-party components and frameworks supported by the ARMS agent, see Overview.
2. Check whether the selected time range for query is correct. Select the past 15 minutes as the time range for query and check whether monitoring data is displayed.
3. If you start the ARMS agent by running the `-jar` command, check the setting of the command and make sure that the -javaagent parameter is before `-jar`. The following command provides an example on how to add parameters to the start command:

   ```
   java -javaagent:/{user.workspace}/ArmsAgent/arms-bootstrap-1.7.0-SNAPSHOT.jar -Darms.licenseKey=xxx -Darms.appName=xxx -jar demoApp.jar
   ```

4. If the logs stored in *ArmsAgent/log/* contains the "LicenseKey is invalid." error, check whether the region of your application is the same as that of the ARMS agent.
5. After your application is started, if the log folder does not exist in the *ArmsAgent* folder, it indicates that *arms-bootstrap-1.7.0-SNAPSHOT.jar* fails to be loaded. Check whether the permissions of the ArmsAgent folder are correct.
6. If the following error is reported when your application is started, check whether the *arms-bootstrap-1.7.0-SNAPSHOT.jar* package and the corresponding permissions are correct.

   ```
   Error opening zip file or JAR manifest missing: /root/ArmsAgent/arms-bootstrap-1.7.0-SNAPSHOT.jar
   Error occurred during initialization of VM
   agent library failed to init: instrument
   ```

7. If no monitoring data is displayed, compress the logs of the ARMS agent for Java applications in the *ArmsAgent/log* folder into a compressed file, and contact the DingTalk account `arms160804` for support.

8. Check the JDK version. If the JDK version is 1.8.0_25 or 1.8.0_31, you may fail to install the agent. We recommend that you upgrade the JDK or contact the DingTalk account `arms160804`.

[Back to the top]

What do I do if no IP address or an incorrect IP address is displayed after the ARMS agent is installed?

1. Run the **ifconfig -a** command to check the network configuration of the current machine. If the machine uses multiple network interface controllers (NICs), the IP address obtained by the ARMS agent may be inconsistent with the actual IP address due to network configurations.
2. You can solve the problem by using one of the following methods:
   - Configure the `-DEAGLEEYE.LOCAL.IP=10.XX.XX.XX` parameter of the JVM.

     > **Note** Replace `10.XX.XX.XX` with the actual IP address.

   - Configure the ARMS agent to obtain the value of the `-DNETWORK.INTERFACE=eth0` parameter, in which `eth0` indicates the NIC name.

[Back to the top]

How do I troubleshoot common errors contained in the log files of the ARMS agent stored in the ArmsAgent/log folder?

If the "LicenseKey is invalid" error is contained in the logs, perform the following operations to troubleshoot the error:

1. Make sure that the application is created in ARMS and the LicenseKey that you specified when the ARMS agent is installed is correct.
2. ARMS supports multiple regions. Therefore, check whether the download URL of the ARMS agent is in the same region as your application.

[Back to the top]

How do I deploy multiple instances on a single machine?

To deploy multiple instances of an application on a single machine, configure the -Darms.agentId parameter to specify the JVM process to connect. This parameter indicates a logical number. Examples: 001 and 002. The following command provides an example on how to deploy multiple instances on a single machine:

```
java -javaagent:/{user.workspace}/ArmsAgent/arms-bootstrap-1.7.0-SNAPSHOT.jar -Darms.licenseKey=<LicenseKey> -Darms.appName=<AppName> -Darms.agentId=001 -jar demoApp.jar
```

[Back to the top]

What do I do when getcwd errors are reported when I run the script to access a Java application?

The following error message is returned after you run the script to access the Java application with one click:

```
shell-init: error retrieving current directory: getcwd: cannot access parent directories: No such file or directory Error occurred during initialization of VM java.lang.Error: Properties init: Could not determine current working directory. at java.lang.System.initProperties(Native Method) at java.lang.System.initializeSystemClass(System.java:1119)
```

This may because that the current directory is accidentally deleted when the script is running. To solve this problem, run the `cd` command and then run the script again.

[Back to the top]

Where do I view the logs after I install the ARMS agent with one click?

By default, the logs are stored in */root/.arms/supervisor/logs/*. If no logs are stored in this folder, run `ps -ef |grep arms` to view the folder where the logs are stored.

[Back to the top]

What do I do if the ARMS agent cannot be installed?

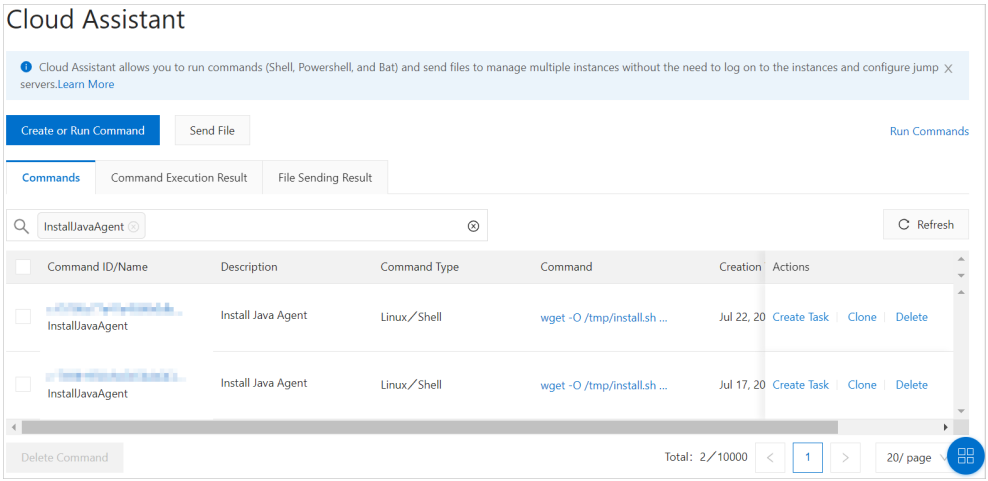
1. Make sure that your ECS instance can access the download URL of the ARMS agent over the Internet in the region in which the ECS instance is located.

Region	Download URL in Internet
China (Hangzhou)	`http://arms-apm-hangzhou.oss-cn-hangzhou.aliyuncs.com/install.sh`
China (Shanghai)	`http://arms-apm-shanghai.oss-cn-shanghai.aliyuncs.com/install.sh`
China (Qingdao)	`http://arms-apm-qingdao.oss-cn-qingdao.aliyuncs.com/install.sh`
China (Beijing)	`http://arms-apm-beijing.oss-cn-beijing.aliyuncs.com/install.sh`
China (Shenzhen)	`http://arms-apm-shenzhen.oss-cn-shenzhen.aliyuncs.com/install.sh`
Singapore	`http://arms-apm-ap-southeast.oss-ap-southeast-1.aliyuncs.com/cloud_ap-southeast-1/install.sh`

2. Make sure that your ECS instance can access the ARMS console.
   - ARMS console for regions in China
   - ARMS console for the Singapore region
3. Log on to the ECS console and perform the following operations:
   i. In the left-side navigation pane, choose **Maintenance & Monitoring > ECS Cloud**

Assistant.

ii. On the **Cloud Assistant** page, click the **Commands** tab, select **Command Name** in the search box, enter `InstallJavaAgent`, and press the Enter key.



> **Note** If no result is returned for the search, contact the DingTalk account `arms160804`.

iii. On the **Cloud Assistant** page, click the **Command Execution Result** tab and enter the ID of the `InstallJavaAgent`. In the search results, click **View** in the **Actions** column corresponding to the command and check whether the `command` is successfully executed. If the command is not successfully executed, troubleshoot the errors based on the execution results. For example, if the problems occur because the disk of the ECS instance is full or the ARMS agent is not installed, you can clear the disk or install the ARMS agent. If the errors cannot be resolved, send the execution results to the DingTalk account `arms160804` for support.

[Back to the top]

What do I do if the information about the processes of the ECS instance is inaccurate after the ARMS agent is installed?

If the information about the processes of the ECS instance is not displayed or is inaccurate after the ARMS agent is installed, click the - icon on the left side of the ECS instance and click the + icon to refresh the data. If the problem retains, contact the DingTalk account `arms160804` for support.

[Back to the top]

What do I do if I cannot enable ARMS application monitoring for a process on an ECS instance?

On the ECS instance, check whether the `/root/.arms/supervisor/logs/arms-supervisor.log` file contains errors. If the log file contains errors, troubleshoot the error based on the error message. If the error cannot be solved, contact the DingTalk account `arms160804` for support.

[Back to the top]

Why is there no data displayed in Application Monitoring after the ARMS agent is installed on a Java application in an ACK cluster?

1. Log on to the Alibaba Cloud Container Service for Kubernetes console.

2. In the left-side navigation pane, choose **Clusters**. On the **Clusters** page, click **Applications** in the Actions column corresponding to the cluster in which the Java application is deployed.
3. At the top of the **Pods** tab, select the namespace in which your application resides. Click **Edit** next to the application.
4. In the **Edit YAML** dialog box, check whether the YAML file contains initContainers.
   - If initContainers is not contained in the YAML file, the pod has not been injected to arms-init-container. Perform Step 5.
   - If initContainers is contained in the YAML file, the pod has been injected to arms-init-container. Perform Step 8.
5. At the top of the **Pods** tab, set Namespace to **arms-pilot**. Check whether any pods whose names contain the **arms-pilot** prefix exist in the Pod list.
   - If pods whose names contain the prefix exist, perform Step 6.
   - If pods whose names contain the prefix do not exist, install arms-pilot from the application market. For more information, see Install the ARMS agent for Java applications in Container Service for Kubernetes.
6. On the **Deployments** or **StatefulSets** tab, choose **More > View in YAML** in the **Actions** column. In the **Edit YAML** dialog box, check whether the YAML file contains the following annotations.

   ```
   annotations:
     armsPilotAutoEnable: 'on'
     armsPilotCreateAppName: <your-deployment-name>
   ```

   - If the YAML file contains the annotations, perform Step 7.
   - If the YAML file does not contain the annotations, add the preceding annotations to the *spec > template > metadata* section in the **Edit YAML** dialog box, replace `<your-deployment-name>` with your application name. Then, click **Update**.
7. On the **Pods** tab, click **Logs** next to the required pod to check whether the pod logs of arms-pilot reports an STS error in the `"Message":"STS error"` format.
   - If the error is reported, authorize the cluster of the application and restart the pod of the application. For more information, see Install the ARMS agent for Java applications in Container Service for Kubernetes.
   - If the error is not reported, contact the DingTalk account arms160804 for support.
8. On the **Pods** tab, click **Edit** next to the required pod. In the **Edit YAML** dialog box, check whether the YAML file contains the following javaagent parameter:

   ```
   -javaagent:/home/admin/.opt/ArmsAgent/arms-bootstrap-1.7.0-SNAPSHOT.jar
   ```

   - If the YAML file contains the parameter, find the pod of the application on the **Pods** page, and click **Terminal** next to the target pod to access the **command line** page. Run the following command to check whether any log file name is suffixed with .log. Then, contact the DingTalk account arms160804.

     ```
     cd /home/admin/.opt/ArmsAgent/logs
     ```

   - If the YAML file does not contain the parameter, contact the DingTalk account arms160804 for support.

[Back to the top]

What do I do if the application cannot be started?

Run the following command to view the arms-pilot-system logs and troubleshoot the problem based on the logs:

```
kubectl logs -f {arms-pilot-arms-pilot-XXX} -n arms-pilot-system
```

[Back to the top]

How do I view the logs of the ARMS agent?

On the worker of the ACK cluster, view the */home/admin/.opt/ArmsAgent/logs/xxxx.log* files.

[Back to the top]

How do I modify the name of a common Java application on which the ARMS agent is manually installed?

Common Java applications indicate Java applications other than those deployed on ECS instances. If you manually install the ARMS agent on the application, the directory of the agent is specified by you during the installation.

You can check the version of the ARMS agent by viewing the Version file in the directory of the agent. For example, `2.5.8_cf020486_20190816150025` indicates that the version of the ARMS agent is 2.5.8 and was released on August 16, 2019.

- If the version of the ARMS agent is earlier than 2.5.8.1, uninstall the agent and install it again. You can specify a new application name when you reinstall the agent.
  - For more information about how to uninstall the ARMS agent that is manually installed, see How do I uninstall the ARMS agent that is manually installed?
  - For more information about how to uninstall the ARMS agent that is installed with one click, see How do I uninstall the ARMS agent that is installed with one click?
  - For more information about how to uninstall the ARMS agent installed on applications in ECS instances, see How do I uninstall the ARMS agent installed on applications in ECS instances?
- If the version of the ARMS agent is 2.5.8.1 or later, you can perform the following steps to modify the name of the Java application without reinstalling the ARMS agent.

  > **Note** ARMS agents downloaded after August 20, 2019 support this feature.

  1. Run one of the following commands to download and decompress Supervisor from the Internet or VPC.

     > **Note** Replace the download URL in the commands with the URL for the region in which the Java application is located.

     - Download Supervisor from the Internet

       ```
       wget http://arms-apm-hangzhou.oss-cn-hangzhou.aliyuncs.com/ArmsSupervisor.zip -O ArmsSupervisor.zip
       unzip ArmsSupervisor.zip
       ```

     - Download Supervisor from VPC (when the Internet download URL is unavailable)

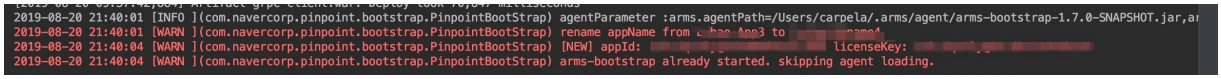
```
wget http://arms-apm-hangzhou.oss-cn-hangzhou-internal.aliyuncs.com/ArmsSupervisor.zip -O ArmsSupervisor.zip
unzip ArmsSupervisor.zip
```

2. Run the following command to modify the name of the Java application:

```
cd ArmsSupervisor
./attach.sh </path/to/ArmsAgent/arms-bootstrap-1.7.0-SNAPSHOT.jar> <PID> <NewAppName> <LicenseKey>
```

   - *</path/to/ArmsAgent/arms-bootstrap-1.7.0-SNAPSHOT.jar>*: the path of the arms-bootstrap-1.7.0-SNAPSHOT.jar file.
   - *<PID>*: the ID of the process. You can run the `jps/ps` command to obtain the version ID.
   - *<NewAppName>*: the new application name.
   - *<LicenseKey>*: the license key of the application monitored by ARMS, which can be obtained from the ARMS console.

If the standard output shown in the following figure is displayed.



[Back to the top]

How do I modify the name of a common Java application on which the ARMS agent is installed with one click?

If you install the ARMS agent with one click, the agent is installed in the *~/.arms/supervisor/agent* directory. Note that the account you use must be the same as the application account.

Perform the following steps to modify your application name:

1. Log on to the machine where the application is located and run the following command:

```
cd ~/.arms/supervisor
./cli.sh <LicenseKey> <NewAppName>
```

   - *<LicenseKey>*: LicenseKey of the application monitored by ARMS, which can be obtained from the ARMS console.
   - *<NewAppName>*: new application name.

2. Select the proper process from all Java processes.

   > **Note** If only one process is available, this process is selected by default.

3. Open the *~/.arms/attach.info* file, change the application name to the new name, and save the file.

   > **Warning** Do not add content such as spaces when you modify the file. Otherwise, the modification may fail because it does not match the new application name modified in the preceding steps.

   Wait a moment after your application name is modified. Monitoring data of the application is reported using the new name rather than the previous name.

How do I change the name of a Java application deployed on ECS instances?

If your Java application is deployed on an ECS instance, the ARMS agent directory is */.arms/agent*.

Perform the following steps to modify your application name:

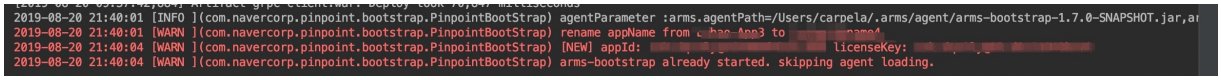
1. Log on to the ECS instance where your application is located and run the following command with the root account:

   ```
   su <USER> -c "./attach.sh /.arms/agent/arms-bootstrap-1.7.0-SNAPSHOT.jar <PID> <NewAppName> <LicenseKey>"
   ```

   - *<PID>*: target process ID, which can be obtained by using the `jps/ps` command.
   - *<NewAppName>*: new application name.
   - *<LicenseKey>*: LicenseKey of the application monitored by ARMS, which can be obtained from the ARMS console.
2. Open the *~/.arms/attach.info* file, change the application name to the new name, and save the file.

   Warning Do not add content such as spaces when you modify the file. Otherwise, the modification may fail because it does not match the new application name modified in the preceding steps.

If the standard output shown in the following figure is displayed, it indicates that the application name is modified.



How do I modify the name of a Java application deployed in an ACK cluster?

You can check the version of the ARMS agent by viewing the Version file in the directory of the agent. For example, `2.5.8_cf020486_20190816150025` indicates that the version of the ARMS agent is 2.5.8 and was released on August 16, 2019.

- If the version of the ARMS agent is earlier than 2.5.8.1, uninstall the agent and install it again. You can specify a new application name when you reinstall the agent.
  - For more information about how to uninstall the ARMS agent that is manually installed, see How do I uninstall the ARMS agent that is manually installed?
  - For more information about how to uninstall the ARMS agent that is installed with one click, see How do I uninstall the ARMS agent that is installed with one click?
  - For more information about how to uninstall the ARMS agent installed on applications in ECS instances, see How do I uninstall the ARMS agent installed on applications in ECS instances?
- If the version of the ARMS agent is 2.5.8.1 or later, you can perform the following steps to modify the name of the Java application without reinstalling the ARMS agent.

  Note ARMS agents downloaded after August 20, 2019 support this feature.
-

Change the value of the armsPilotCreateAppName parameter in Deployment and restart the pod.

Wait a moment after your application name is modified. Monitoring data of the application is reported using the new name rather than the previous name.

[Back to the top]

How do I modify the name of a Java application deployed in Enterprise Distributed Application Service (EDAS)?

The names of Java applications deployed in EDAS cannot be modified.

[Back to the top]

How do I uninstall the ARMS agent that is manually installed?

For more information about how to manually install the ARMS agent on Java applications, see Manually install the ARMS agent for a Java application. You can perform the following steps to uninstall the ARMS agent that is manually installed:

1. If you no longer want to use ARMS to monitor your Java applications, delete all parametersrelated to AppName and LicenseKey, which are described in Step 8.
2. Restart the Java application.

[Back to the top]

How do I uninstall the ARMS agent that is installed with one click?

For more information about how to install the ARMS agent with one click, see Install the ARMS agent for a Java application by using scripts. You can perform the following steps to uninstall the ARMS agent that is installed with one click:

1. If you no longer want to use ARMS to monitor your Java applications, run the `jps -l` command to view all processes and find the process ID of `com.alibaba.mw.arms.apm.supervisor.daemon.Daemon` in the returned results.

   In this example, the process ID of `com.alibaba.mw.arms.apm.supervisor.daemon.Daemon` is 62857.

   ```
   ➜ ~ jps -l
   62800 org.apache.catalina.startup.Bootstrap
   62857 com.alibaba.mw.arms.apm.supervisor.daemon.Daemon
   5411
   62799 org.jetbrains.jps.cmdline.Launcher
   67809 sun.tools.jps.Jps
   ```

2. Run the `kill -9 <process ID>` command.

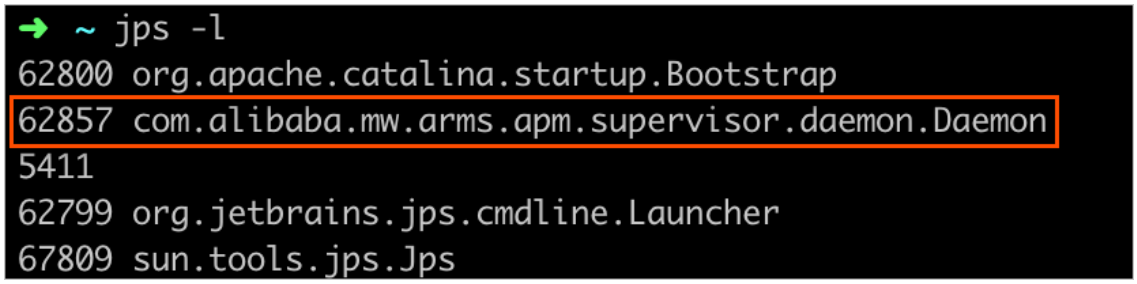
   Example: `kill -9 62857`.
3. Run the `rm -rf /.arms /root/.arms` command.
4. Restart your application.

[Back to the top]

How do I uninstall the ARMS agent installed on a Java application deployed on an ECS instance?

1. If you no longer want to use ARMS to monitor your Java applications, run the `jps -l` command to view all processes and find the process ID of `com.alibaba.mw.arms.apm.supervisor.daemon.Daemon` in the returned results.

   In this example, the process ID of `com.alibaba.mw.arms.apm.supervisor.daemon.Daemon` is 62857.

2. Run the `kill -9 <process ID>` command.

   Example: `kill -9 62857`.

3. Run the `rm -rf /.arms /root/.arms` command.
4. Restart your application.
5.

[Back to the top]

How do I uninstall the ARMS agent installed on a Java application in an ACK cluster?

If you no longer want to use ARMS to monitor your Java applications in an ACK cluster, perform the following steps to uninstall the ARMS agent:

1. Log on to the Alibaba Cloud Container Service for Kubernetes console .
2. In the left-side navigation pane, click **Clusters**. On the **Clusters** page, click **Applications** in the **Actions** column corresponding to the cluster that contains the Java application from which you want to uninstall the ARMS agent.
3. In the left-side navigation pane, select **Releases**.
4. On the **Helm** tab, select the release name **arms-pilot** of the ARMS agent, and click **Delete** in the **Actions** column.
5. In the **Delete** dialog box, click **OK**.
6. Restart your business pod.

[Back to the top]

How do I uninstall the ARMS agent installed on a Java application in an open-source Kubernetes environment?

1. If you no longer want to use ARMS to monitor your Java applications in an open-source Kubernetes environment, run the following command to uninstall arms-pilot:

   ```
   helm del --purge arms-pilot
   ```

2. Restart your business pod.

[Back to the top]

How do I uninstall the ARMS agent installed on a Java application in Docker?

1. If you no longer want to use ARMS to monitor your Java applications in a Docker cluster, delete the *Dockerfile* content edited in Step 1 in the "Install the ARMS agent for applications in Docker" topic.
2. Run the **docker build** command to construct the image.
3. Run the **docker run** command to start the image.

[Back to the top]

How do I uninstall the ARMS agent installed on a PHP application?

If you no longer want to use ARMS to monitor your PHP applications that are not deployed in an ACK cluster, perform the following steps to uninstall the ARMS agent:

1. Delete the following four lines from the *php.ini* file:

```
[arms]
extension=<php_extension_dir>/arms.so
arms.trace_exception=true
arms.config_full_name=/<php-agent-dir>/arms-agent.conf
```

2. Restart your PHP application.

[Back to the top]

How do I uninstall the ARMS agent installed on a PHP application in an ACK cluster?

If you no longer want to use ARMS to monitor your PHP applications that are deployed in an ACK cluster, perform the following steps to uninstall the ARMS agent:

1. Log on to the Alibaba Cloud Container Service for Kubernetes console .
2. In the left-side navigation pane, click **Clusters**. On the **Clusters** page, click **Applications** in the **Actions** column corresponding to the cluster that contains the Java application from which you want to uninstall the ARMS agent.
3. In the left-side navigation pane, select **Releases**.
4. On the **Helm** tab, select the release name **arms-pilot** of the ARMS agent, and click **Delete** in the **Actions** column.
5. In the **Delete** dialog box, click **OK**.
6. Restart your business pod.
7.

[Back to the top]

What do I do if the data of my application with the OpenFeign component is incomplete in ARMS?

After your application with the OpenFeign component is connected to ARMS application monitoring, if the data is incomplete and the data of downstream applications cannot be viewed, this may be because that the OpenFeign component enables Hystrix using the RxJava asynchronous framework by default, whereas ARMS does not support asynchronous frameworks.

> **Note** This topic applies to scenarios where the version of the ARMS agent for Java applications is earlier than 2.6.0. The ARMS agent for Java applications of the 2.6.0 version or later already supports asynchronous frameworks.

You can disable Hystrix and enable the OkHttp request class to resolve this problem.

1. Add the following dependencies to the *pom.xml* file.

```
<! -- OK Http supports Feign -->
<dependency>
  <groupId>io.github.openfeign</groupId>
  <artifactId>feign-okhttp</artifactId>
</dependency>
```

2. Add the following content to the SpringCloud configuration file:

```
feign.okhttp.enabled: true
feign.hystrix.enabled: false
```

3. Configure the OkHttp request class.

```
@Configuration
@ConditionalOnClass(Feign.class)
@AutoConfigureBefore(FeignAutoConfiguration.class)
public class FeignClientOkHttpConfiguration {
  @Bean
  public OkHttpClient okHttpClient() {
    return new OkHttpClient.Builder()
        // The connection times out.
        .connectTimeout(20, TimeUnit.SECONDS)
        // The response times out.
        .readTimeout(20, TimeUnit.SECONDS)
        // The write request times out.
        .writeTimeout(20, TimeUnit.SECONDS)
        // Indicates whether to enable automatic reconnection.
        .retryOnConnectionFailure(true)
        // The connection tool.
        .connectionPool(new ConnectionPool())
        .build();
  }
```

[Back to the top]

8.Troubleshooting

8.1. Why is no monitoring data displayed in the ARMS console after I install an ARMS agent?

Condition

After an ARMS agent is installed, no monitoring data is displayed in the ARMS console.

Cause

This problem may occur because the application does not have continuous external access requests, the region of the application is different from that of the ARMS agent, or the permissions on the ARMS agent installation directory are invalid.

Procedure

1. If the log file of the ARMS agent contains "send agent metrics. no metrics.", verify whether the development framework is supported by the ARMS agent and whether your application has continuous external access requests, including HTTP requests, HSF requests, and Dubbo requests. For more information about third-party components and frameworks supported by the ARMS agent, see Java components and frameworks supported by ARMS and PHP components and frameworks supported by ARMS Application Monitoring.
2. Check whether the selected time range to query is correct. Select the last 15 minutes as the query time range and check whether monitoring data is displayed.
3. If you start the ARMS agent by running the `-jar` command, check the configuration of the command line and make sure that the -javaagent parameter is before `-jar`.

   ```
   java -javaagent:/{user.workspace}/ArmsAgent/arms-bootstrap-1.7.0-SNAPSHOT.jar -Darms.licenseKey=xxx -Darms.appName=xxx -jar demoApp.jar
   ```

4. If the log in ArmsAgent/log/ contains the LicenseKey is invalid. error, check whether the region of your application is the same as that of the ARMS agent.
5. If the log subdirectory does not exist in the ArmsAgent directory after your application is started, this is because arms-bootstrap-1.7.0-SNAPSHOT.jar failed to be loaded. Check whether the permissions on the ARMS agent installation directory are valid.
6. If the following error is reported when your application is started, check whether the arms-bootstrap-1.7.0-SNAPSHOT.jar package and the permissions are correct.

   ```
   Error opening zip file or JAR manifest missing : /root/ArmsAgent/arms-bootstrap-1.7.0-SNAPSHOT.jar
   Error occurred during initialization of VM
   agent library failed to init: instrument
   ```

7. If no monitoring data is displayed, pack the log file of the ARMS agent for Java applications in the ArmsAgent/log directory, and contact customer services of ARMS at DingTalk service account `arms160804` for assistance.
8. Check the JDK version.
   - If the JDK version is 1.8.0_25 or 1.8.0_31, you may fail to install the agent. We recommend that you upgrade the JDK or contact customer services of ARMS at DingTalk service account `arms1608`

04 .

- If the JDK version is 1.10.X, download the correct agent installation package from one of the following addresses:
  - China (Hangzhou)
  - China (Shanghai)
  - China (Qingdao)
  - China (Beijing)
  - China (Zhangjiakou)
  - China (Shenzhen)
  - China (Hong Kong)

8.2. Why is no data displayed in Application Monitoring after the ARMS agent is installed on a Java application in a cluster of Container Service for Kubernetes (ACK)?

Condition

No data is displayed in Application Monitoring after the Application Real-Time Monitoring Service (ARMS) agent is installed on a Java application in an ACK cluster.

Cause

The pod of the application is not injected to arms-init-container, the YAML file of the application contains no annotations, or Security Token Service (STS) is not authorized.

Remedy

Procedure

1. Log on to the Alibaba Cloud Container Service for Kubernetes console .
2. In the left-side navigation pane, click **Clusters**. On the **Clusters** page, find the required cluster, and click **Applications** in the **Actions** column.
3. In the upper part of the **Pods** tab, select the namespace in which your application resides. Click **Edit** next to the application.
4. In the **Edit YAML** dialog box, check whether the YAML file contains initContainers.
   - If the YAML file does not contain initContainers, the pod has not been injected to arms-init-container. Perform .
   - If the YAML file contains initContainers, the pod has been injected to arms-init-container. Perform .
5. In the upper part of the **Pods** tab, set Namespace to **arms-pilot**. Check whether any pods that have the **arms-pilot** prefix exist in the Pod list.
   - If any pods that have the arms-pilot prefix exist in the Pod list, perform .

- Otherwise, install arms-pilot in the application market. For more information, see Install the ARMS agent for Java applications in Container Service for Kubernetes.

6. On the **Deployments** or **StatefulSets** tab, choose **More > View in YAML** in the **Actions** column. In the **Edit YAML** dialog box, check whether the YAML file contains the following annotations:

```
annotations:
  armsPilotAutoEnable: 'on'
  armsPilotCreateAppName: <your-deployment-name>
```

   - If the YAML file contains these annotations, perform .
   - If the YAML file does not contain these annotations, add the preceding annotations to the *spec > template > metadata* section in the **Edit YAML** dialog box, replace `<your-deployment-name>` with your application name, and then click **Update**.

7. On the **Pods** tab, click **Logs** next to the required pod to check whether the pod logs of arms-pilot report an STS error in the `"Message":"STS error"` format.
   - If the pod logs of arms-pilot report an STS error, authorize the cluster of the application and restart the pod of the application. For more information, see Install the ARMS agent for Java applications in Container Service for Kubernetes.
   - If the pod logs of arms-pilot do not report an STS error, contact the ARMS DingTalk account arms160804.

8. On the **Pods** tab, click **Edit** next to the required pod. In the **Edit YAML** dialog box, check whether the YAML file contains the following javaagent parameter:

```
-javaagent:/home/admin/.opt/ArmsAgent/arms-bootstrap-1.7.0-SNAPSHOT.jar
```

   - If the YAML file contains the javaagent parameter, find the pod of the application on the **Pods** tab, and click **Terminal** next to the required pod to go to the **Shell** page. Run the following command to check whether any log file name is suffixed with .log, and then contact the ARMS DingTalk accoun arms160804.

```
cd /home/admin/.opt/ArmsAgent/logs
```

   - If the YAML file does not contain the javaagent parameter, contact the ARMS DingTalk account arms160804.