Enterprise Distributed Application Service (EDAS)

Application diagnostics

Application diagnostics

Application Diagnosis

General operations

EDAS provides a container monitoring function – application diagnosis that collects data to help you detect problems (such as memory and class conflicts) of the applications that are deployed and run inside the Tomcat container.EDAS provides the refined statistics function designed for application containers, which collects statistics on the application running instance based on a range of statistical items, including JVM heap memory, non-heap memory, class loader, thread, and Tomcat connector.Similar to infrastructure monitoring, container monitoring (application diagnosis) lists application-specific single-host data.

The differences are as follows:

- The monitored object of infrastructure monitoring is ECS instances, whereas that of container monitoring is the container where applications reside.
- Application diagnosis supports querying of diagnostic information in single-instance mode rather than cluster mode.
- Infrastructure monitoring has latency, whereas container monitoring is near real-time because statistical computing is not performed on collected data (except memory monitoring data).

To view container details, follow these steps:

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

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

Expand the **ECS Instances (Instance ID/Name/IP)** drop-down list and select an ECS instance.

Click tabs to view the monitoring details of the container.

The Application Diagnosis page contains the following tabs:

GC Diagnosis: This tab includes the GC Diagnosis and Memory sections.

GC Diagnosis: This can monitor some performance metrics of the current instance that have GC according to the instances of the current application, and analyze the GC of current instance according to the selected time range.These metrics help you to judge the health status of an instance of the application, that is, check whether the application has memory leakage or a large object.

- The GC policy of the current instance is -XX:+UseParallelGC or -XX:+UseParallelOldGC.
- FGC refers to Full Garbage Collection while YGC refers to Young Garbage Collection.
- When YGC occurs more than six times or FGC occurs more than 10 times within one minute, and the YGC or FGC number is the largest in the selected time range in this one minute, this one minute is regarded as the most active time for YGC/FGC.
- When the total YGC consumed time in one minute exceeds 100 ms or the total FGC consumed time exceeds 300 ms within one minute, and the total YGC/FGC consumed time in this one minute is the longest in the selected time range, this one minute is called the time with the greatest YGC/FGC time consumption.
- Memory before and after recycling refers to the memory occupied by the application.

- Memory: Memory is monitored on a per-instance basis. EDAS provides statistics on the heap memory and non-heap memory of the JVM process of the Tomcat container where the application is located.

Class Loading: provides real-time information about JAR package loading.When a JAR package of the application has version conflict, you can use this function to easily locate the path to which the JAR package is loaded, which lowers troubleshooting costs.

- **Thread**: displays the basic information of all threads of the current JVM process, including the thread ID, status, and name. The statistical fields are native information of JVM.

Connector: The Tomcat connector is indicated by <Connector /> in the XML configuration file of Tomcat. The configuration of each <Connector /> can be considered as a line of pulled information.This view displays the running status of the corresponding connector from the past 10 minutes.

Each connector has a certain number of threads (which forms a thread pool) to process incoming requests. When concurrency or throughput bottlenecks occur, statistics must be collected on the processing status of the connector’ s thread pool. For example, an HTTP connector has the following XML configuration:

<Connector port="8080" protocol="HTTP/1.1" maxThreads="250" .... />

Click **Thread Pool Information** in the Action field next to the connector to view more details.

The preceding figure shows that the application is almost load-free. If the value of “Busy Thread Count” is close to that of “Thread Pool Max. Value” , the system encounters serious concurrency. To resolve this problem, scale up the application or optimize the service code.

Object Memory Distribution: Select **System Class**, **Java Primitive Object Class**, and **Class Loading Related**.Based on the selected three classes, the number of objects, occupied space, and usage percentage of the total system memory are displayed in a pie chart and a list.

Method Tracing: Method tracing is complex. For more information, see Method tracing.

Hot Thread: provides two functions, the retrieval of thread snapshots and the analysis of call statistics. For more information, see the Thread hotspot.

Druid Database Connection Pool Monitor: For more information, see Druid database connection pool monitoring.

Commons Pool: For more information, see Commons Pool monitoring.

Method tracking

Overview

EDAS method tracing helps you quickly troubleshoot problems of running applications. Typical use cases include:

- When you find that it takes a long time to run a service logic, and you want to identify the part of code that causes the time consumption.
- Applications and services are all running smoothly for most of the time. However, a user reports the problem that service response is extremely slow when a specific parameter is passed. In this case, you may need a mechanism to check the code execution related to a specific parameter in a method.
- When a method with complex service logic is executed, you want to have a clear view of the logic and time sequence of service calls in details.

EDAS method tracing is designed to meet the preceding requirements without interfering with code or stopping applications.

EDAS method tracing adopts the JVM bytecode enhancement technique when recording the time consumption and sequence during the entire call process of the selected method, enabling you to check the execution sequence while execution is in progress.

Restrictions

If the following restrictions affect your services or troubleshooting, open a ticket to consult with us so that we can improve some restrictions or configure a whitelist.

In principle, only tracing of service-type classes is supported. Therefore, packages are filtered by name before tracing starts.

Sampled output is adopted to prevent excessive logs due to large amout of method calls. The default policy is to output logs on 10 calls per second for the method.

When you exit and then log on to the EDAS console again, or refresh the screen, historical trace records are lost and previously pulled tracing information is no longer retained.

Automatic stop policy: If method tracing is in inactive state for 10 minutes, EDAS automatically detaches the tracing module and restores the method to the initial state (state prior to enhancement).

Parameter printing: Currently, EDAS only supports printing of basic Java types (string, char, int, float, double, short, and boolean).

If the selected string is too long, EDAS truncates the string to output the first 100 characters.

If the JVM instance restarts during the tracing process, you must stop tracing and then restart it.

Currently, a maximum of 10,000 trace logs can be output. To output more logs, restart the tracing function.

The current version does not support Docker-based applications.

Environment check

Because the method tracing function adopts the JVM bytecode enhancement technique, to ensure normal running of applications, this function is disabled when the environment check fails.

Before the method tracing function starts, EDAS automatically checks that:

1. Ali-Tomcat is in **Running** state.
2. CPU usage is lower than 60%.
3. Available system memory is more than 100 MB.
4. The available space of JVM PermGen or Metaspace is more than 20 MB.

If the environment check fails, we recommend that you clear the alarms and then click **Retry**.

Usage instructions

Log on to the EDAS console.

Select **Applications Management** > **Applications** in the left-side navigation pane and click the name of the application to be checked in the application list.

Click **Application Diagnosis** in the left-side navigation pane of the Application Details page.

Click the **Method Tracing** tab on the application diagnosis page.

Note:

If the **Method Tracing** tab does not appear, follow these steps:

Check that you are using Google Chrome as the web browser, and refresh the page. (We performed tests in Google Chrome only.)

If you log on with a sub-account, check that the sub-account has required permissions.

To check permissions, choose **Applications** > **Application Diagnosis** > **Method Tracing** and **Tool Authorization**.

Before the permission check starts, EDAS performs an environment check on the ECS instance where the application is located. When the environment check dialog box appears, select the checkbox to confirm the precondition and click **Confirm** to start the environment check.

The method tracing page appears after the environment check is successful.

Set tracing parameters.

Note: **Class Name** and **Method Name** are required. Set the two parameters to the class and method you want to trace.

The configuration items are described as follows:

Class Name: Required. Enter a full path name starting with the package path, for example, com.test.alibaba.demo.HelloWorldServlet. EDAS does not support tracing of classes whose names start with the following package paths:

- java.*
- javax.*
- com.google.*
- com.alibaba.*
- com.aliyun.*
- com.taobao.*
- org.apache.*
- org.dom4j.*
- org.springframework.*
- redis.clients.*

After a complete package path is entered, EDAS checks whether the class exists on the selected ECS instance.

- If the entered class exists, it is displayed in the drop-down list. Select the class to continue.
- If the entered class does not exist, the message “Class does not exist” is displayed in the drop-down list.

Method Name: Required. After a class is selected, the system automatically searches for all methods under the class and displays the method list below the textbox.

The icon on the left of each method indicates the modifier of the method.

- public: A green lock
- protected: A yellow key
- private: A red lock
- package: A blue block
- abstract: No icon

Select the method to be traced from the drop-down list and continue.

Exception Tracing Only: The execution of a method either returns a response normally or ends execution due to an exception. If you select "Exception Tracing Only", the tracing results are printed and output only when the method is ended due to an exception.

Print Returned Values: If you select this option, the returned values of the method are printed on the result page. null is output if the return type of the method is void.

After a method is selected, the **Start Tracing** button is available in blue.

Click **Start Tracing** to trace the method. Whenever the method is called, the call information is displayed in the result area.

Note: After method tracing starts, EDAS periodically checks whether the tracing is in active state. If method tracing is in inactive state for 10 minutes, EDAS automatically stops the tracing and restores the method to the initial state.

Check the method call information.

After method tracing starts, EDAS displays the generated call logs in the EDAS console.

On the left of the display area, each record represents the log that is generated each time the method is called.

44-62/150 at the bottom of the table indicates that the browser returns 150 records in total and currently lists the trace records in rows 44 to 62.

The prompt "Press key H to show help information" is displayed at the bottom of the table. Press "H" on the keyboard to display the usage instructions on shortcut keys.

- H: Displays the help document.

- Ctrl+G: Displays the latest data in real time. As the call times increase, it is impossible to render and display all records. Similar to the tail function, Ctrl+G is used to display the latest data once retrieved.
- G: Jumps to a specific record. Search for the trace record in a specific row and select it to display details.
- Ctrl+C or Esc: Ends the command that is being executed.
- Ctrl+H: Goes to the next page to display the next 10 trace records.
- Ctrl+I: Returns to the previous page to display the preceding 10 trace records.
- J or ↓: Selects the next trace record.
- K or ↑: Selects the previous trace record.
- Enter or Double-click: Zooms in or restores the selected trace record.

The right side of the display area displays the details or basic information of the selected record. You can double-click or press **Enter** in the details section to zoom in, or press **Esc** to restore to initial display.

- **Tracing details**: Shows the time consumption and execution sequence for each call. The time in blueis the total time consumed by calling the method.The time in red indicates that the time of the specific call is more than 30% of the total time consumed.
- **Output details**: Shows the exceptions, return values, and input parameters (which are selected using the **More** option) during execution.
- **Method stack details**: Shows the stack information before the traced method is called.

Stop tracing.

After method tracing starts, the **Start Tracing** button changes to **Stop Tracing**. After you click **Stop Tracing**, EDAS restores the traced method to the initial state (state prior to enhancement) and records tracing information in the instance dimension. Next time you go to the method tracing page, the last tracing information is displayed.

If you modify tracing items (for example, method name) after tracing stops and then click **Start Tracing**, the modified information is submitted and tracing starts based on the latest submitted information.

Hot thread

When Druid is used for the database connection pool, the EDAS Druid database connection pool monitoring agent monitors the database connection pool and SQL execution.The monitored data is recorded once every 10 seconds and reset.

Procedure

To use the Druid database connection pool, complete the following steps:

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

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

On the **Application Diagnosis** page, click the drop-down arrow on the right of **ECS Instances (Instance ID/Name/IP)**, and select an ECS instance.

Click the **Druid Database Connection Pool Monitor** tab.

Click **Start Monitoring**.

Information about the database connection pool and SQL execution is displayed.The page is refreshed once every 10 seconds by default.

Note: When you click “Start Monitoring”, if StatFilter provided by the Druid database connection pool is not configured for the application, EDAS automatically adds StatFilter to the application.Given that this may slightly affect the performance, we strongly recommend that you manually add the StatFilter to your application.

Click **Close** to exit monitoring.

Monitoring information description

Monitoring information of the database connection pool

The monitoring information of the database connection pool includes the following:

- Database connection pool monitoring indicators include the database type, driver class, initial connection pool size, and maximum number of connections.
- Information of the database connection pool during running, including the size of available connections, peak size of available connections, and number of active connections.

The following table describes the Druid database connection pool monitoring indicators.

Field	Name	Description
DB Type	DB Type	Indicates the type of the database of the connected data source, for example, MySQL.
Driver Class	Driver Class	Indicates the class of the data driver.
User Name	User Name	Indicates the user of the database connection pool.
Init Size	Init Size	Indicates the initial size of the database connection pool.
Max Active	Max Active	Indicates the maximum number of connections in the database connection pool.
Pool Size	Pool Size	Indicates the number of available connections in the database connection pool.
Maximum Pool Size	Maximum Pool Size	Indicates the maximum number of available connections in the database connection pool.
Active Count	Active Count	Indicates the number of active connections in the database connection pool.

Description of SQL execution information

The SQL execution information consists of the **information of SQL statements executed in the last 10 seconds** and the **information of SQL statements whose maximum execution time exceeds 100 milliseconds**.The monitoring indicators for both of these two parts are the same. They are described in the following table.

Field	Name	Description
SQL	SQL	Indicates the executed SQL statement.
Executed Count	Executed Count	Indicates the number of executions of this SQL statement.
Total Executed Time	Total Executed Time	Indicates the total execution time of this SQL statement.
Maximum Executed Time	Maximum Executed Time	Indicates them maximum execution time of this SQL

		statement.
Maximum Returned Rows	Maximum Returned Rows	Indicates the maximum number of returned rows of this SQL statement.
Monitor Time	Monitor Time	Indicates the recording time of this SQL record.

Druid database connection pool monitoring

When Druid is used for the database connection pool, the EDAS Druid database connection pool monitoring agent monitors the database connection pool and SQL execution.The monitored data is recorded once every 10 seconds and reset.

Procedure

To use the Druid database connection pool, complete the following steps:

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

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

On the **Application Diagnosis** page, click the drop-down arrow on the right of **ECS Instances (Instance ID/Name/IP)**, and select an ECS instance.

Click the **Druid Database Connection Pool Monitor** tab.

Click **Start Monitoring**.

Information about the database connection pool and SQL execution is displayed.The page is refreshed once every 10 seconds by default.

Note: When you click “Start Monitoring”, if StatFilter provided by the Druid database connection pool is not configured for the application, EDAS automatically adds StatFilter to the application.Given that this may slightly affect the performance, we strongly recommend that you manually add the StatFilter to your application.

Click **Close** to exit monitoring.

Monitoring information description

Monitoring information of the database connection pool

The monitoring information of the database connection pool includes the following:

- Database connection pool monitoring indicators include the database type, driver class, initial connection pool size, and maximum number of connections.
- Information of the database connection pool during running, including the size of available connections, peak size of available connections, and number of active connections.

The following table describes the Druid database connection pool monitoring indicators.

Field	Name	Description
DB Type	DB Type	Indicates the type of the database of the connected data source, for example, MySQL.
Driver Class	Driver Class	Indicates the class of the data driver.
User Name	User Name	Indicates the user of the database connection pool.
Init Size	Init Size	Indicates the initial size of the database connection pool.
Max Active	Max Active	Indicates the maximum number of connections in the database connection pool.
Pool Size	Pool Size	Indicates the number of available connections in the database connection pool.
Maximum Pool Size	Maximum Pool Size	Indicates the maximum number of available connections in the database connection pool.
Active Count	Active Count	Indicates the number of active connections in the database connection pool.

Description of SQL execution information

The SQL execution information consists of the **information of SQL statements executed in the last 10**

seconds and the **information of SQL statements whose maximum execution time exceeds 100 milliseconds**.The monitoring indicators for both of these two parts are the same. They are described in the following table.

Field	Name	Description
SQL	SQL	Indicates the executed SQL statement.
Executed Count	Executed Count	Indicates the number of executions of this SQL statement.
Total Executed Time	Total Executed Time	Indicates the total execution time of this SQL statement.
Maximum Executed Time	Maximum Executed Time	Indicates them maximum execution time of this SQL statement.
Maximum Returned Rows	Maximum Returned Rows	Indicates the maximum number of returned rows of this SQL statement.
Monitor Time	Monitor Time	Indicates the recording time of this SQL record.

Commons pool monitoring

When Druid is used for the database connection pool, the EDAS Druid database connection pool monitoring agent monitors the database connection pool and SQL execution.The monitored data is recorded once every 10 seconds and reset.

Procedure

To use the Druid database connection pool, complete the following steps:

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

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

On the **Application Diagnosis** page, click the drop-down arrow on the right of **ECS Instances**

(Instance ID/Name/IP), and select an ECS instance.

Click the **Druid Database Connection Pool Monitor** tab.

Click **Start Monitoring**.

Information about the database connection pool and SQL execution is displayed.The page is refreshed once every 10 seconds by default.

Note: When you click "Start Monitoring", if StatFilter provided by the Druid database connection pool is not configured for the application, EDAS automatically adds StatFilter to the application.Given that this may slightly affect the performance, we strongly recommend that you manually add the StatFilter to your application.

Click **Close** to exit monitoring.

Monitoring information description

Monitoring information of the database connection pool

The monitoring information of the database connection pool includes the following:

- Database connection pool monitoring indicators include the database type, driver class, initial connection pool size, and maximum number of connections.
- Information of the database connection pool during running, including the size of available connections, peak size of available connections, and number of active connections.

The following table describes the Druid database connection pool monitoring indicators.

Field	**Name**	**Description**
DB Type	DB Type	Indicates the type of the database of the connected data source, for example, MySQL.
Driver Class	Driver Class	Indicates the class of the data driver.
User Name	User Name	Indicates the user of the database connection pool.
Init Size	Init Size	Indicates the initial size of the database connection pool.
Max Active	Max Active	Indicates the maximum number of connections in the database connection

		pool.
Pool Size	Pool Size	Indicates the number of available connections in the database connection pool.
Maximum Pool Size	Maximum Pool Size	Indicates the maximum number of available connections in the database connection pool.
Active Count	Active Count	Indicates the number of active connections in the database connection pool.

Description of SQL execution information

The SQL execution information consists of the **information of SQL statements executed in the last 10 seconds** and the **information of SQL statements whose maximum execution time exceeds 100 milliseconds**.The monitoring indicators for both of these two parts are the same. They are described in the following table.

Field	**Name**	**Description**
SQL	SQL	Indicates the executed SQL statement.
Executed Count	Executed Count	Indicates the number of executions of this SQL statement.
Total Executed Time	Total Executed Time	Indicates the total execution time of this SQL statement.
Maximum Executed Time	Maximum Executed Time	Indicates them maximum execution time of this SQL statement.
Maximum Returned Rows	Maximum Returned Rows	Indicates the maximum number of returned rows of this SQL statement.
Monitor Time	Monitor Time	Indicates the recording time of this SQL record.