Sun GlassFish Enterprise Manager DTrace Monitoring 3.0 Beta Installation and Quick Start Guide

Beta

Sun Microsystems, Inc.
4150 Network Circle
Santa Clara, CA 95054
U.S.A.

Part No: 821–1005–05
December 2009

091209@23031

Contents

Sun GlassFish Enterprise Manager DTrace Monitoring 3.0 Beta Installation and Quick Start Guide

Sun GlassFish™ Enterprise Manager DTrace Monitoring enables you to use the DTrace utility of the Solaris™ Operating System (OS) to monitor the performance of Sun GlassFish Enterprise Server. Enterprise Server provides standard DTrace probes to enable you to use all the features of DTrace. DTrace Monitoring is supported only on operating systems that support DTrace, such as the Solaris OS.

Sun GlassFish Enterprise Manager DTrace Monitoring 3.0 Beta Installation and Quick Start Guide explains how to install DTrace Monitoring, including how to enable DTrace monitoring for Enterprise Server. This document also explains how to obtain information about Enterprise Server DTrace probes and provides sample D scripts that illustrate the use of these probes to monitor Enterprise Server. The ability to use DTrace is assumed.

Note – DTrace Monitoring is beta release software. Use this software for evaluation purposes only. Do *not* use this software in production deployments.

The following topics are addressed here:

Downloading and Installing DTrace Monitoring

DTrace Monitoring is available under an evaluation licence from the Sun GlassFish Enterprise Manager DTrace Monitoring download page.

Requirements for Downloading DTrace Monitoring

Before you can download and install DTrace Monitoring, you must have a Sun Online Account (`https://reg.sun.com/whyregister`).

▼ To Install DTrace Monitoring

Before You Begin Ensure that the following prerequisites are met:

- The DTrace Monitoring distribution file is downloaded from the Sun GlassFish Enterprise Manager DTrace Monitoring download page.
- The following software is installed on your machine:
  - Sun GlassFish Enterprise Server v3
  - JDK™ release 7
- Enterprise Server uses JDK release 7.

JDK release 7 is available for download from the Java Early Access Downloads site (`http://java.sun.com/javase/downloads/ea.jsp`).

1. **Change to the directory to which you downloaded the ZIP file.**

   `cd` *download-dir*

   download-dir is the directory where you downloaded the ZIP file.

2. **Unzip the ZIP file.**

   ```
   unzip monitoring_dtrace-3_0-beta.zip
   ```

   When you unzip the file, a directory that is named `monitoring_dtrace` is created within the current directory. This new directory contains a subdirectory that is named `lib`, which contains the `glassfish-dtrace.jar` file.

3. **Copy the `glassfish-dtrace.jar` file to the Enterprise Server `modules` directory.**

   `cp monitoring_dtrace/lib/glassfish-dtrace.jar` *as-install***`/modules`**

   as-install is the directory in which Enterprise Server is installed.

4. **If necessary, start an administrative domain.**

   `asadmin start-domain` *domain*

domain is the name of the administrative domain to start.

5 **Enable support for DTrace monitoring in Enterprise Server.**

```
asadmin set configs.config.server-config.monitoring-service.dtrace-enabled=true
```

DTrace Documentation

For general information about using DTrace, see the following documentation:

- *DTrace User Guide*
- *Solaris Dynamic Tracing Guide*
- The `dtrace`(1M) man page

Provider in the Names of Enterprise Server DTrace Probes

The provider of the Enterprise Server DTrace probes is `glassfish`. DTrace treats the Enterprise Server DTrace probes as process identifier (`pid`) probes and appends the process identifier (ID) of the Enterprise Server process to the provider in the names of these probes. Therefore, you must specify the process ID or the wildcard character * whenever you specify the provider of the Enterprise Server DTrace probes. Typically, you specify the provider in probe names in D scripts or in the `-P` option of the `dtrace` command.

Enterprise Server add-on components that are developed by an external vendor can also provide DTrace probes. The provider in the names of the probes for an add-on component need not be `glassfish`, but is determined by the developer of the add-on component.

Obtaining Information About Enterprise Server DTrace Probes

Information about Enterprise Server DTrace probes enables you to determine the DTrace probes to use for monitoring the aspects of Enterprise Server performance that are of interest to you.

You can obtain information about Enterprise Server DTrace probes by using the following software:

- DTrace
- Sun GlassFish Enterprise Manager Monitoring Scripting Client

Using DTrace to Obtain Information About Enterprise Server DTrace Probes

You can obtain information about Enterprise Server DTrace probes by using the `dtrace` command with appropriate options to list the probes that are of interest to you.

EXAMPLE 1 Listing All Enterprise Server DTrace Probes

This examples uses the `-l` option and the `-P` option of the `dtrace` command to list all Enterprise Server DTrace probes.

In this example, the process ID of the Enterprise Server process is 755.

For better readability, the spacing between columns is reduced and some probes that would listed by this example are not shown.

```
# /usr/sbin/dtrace -l -P glassfish\*
   ID   PROVIDER        MODULE            FUNCTION NAME
40518 glassfish755        kernel         thread-pool threadReleasedEvent
40519 glassfish755        kernel         thread-pool maxNumberOfThreadsReachedEvent
40520 glassfish755        kernel         thread-pool threadAllocatedEvent
40521 glassfish755        kernel         thread-pool setCoreThreadsEvent
40522 glassfish755        kernel         thread-pool threadReturnedToPoolEvent
...
40637 glassfish755           jca     connection-pool incrementNumConnFreeEvent
40638 glassfish755           orb   inboundconnection inboundConnectionClosed
40639 glassfish755           orb   inboundconnection inboundConnectionOpened
40640 glassfish755           orb  outboundconnection outboundConnectionClosed
40641 glassfish755           orb  outboundconnection outboundConnectionOpened
40786 glassfish755      security                 ejb policyDestructionEvent
40787 glassfish755      security                 ejb securityManagerCreationEvent
40788 glassfish755      security                 ejb securityManagerDestructionEvent
40789 glassfish755        jersey       server-hidden requestEnd
40790 glassfish755        jersey       server-hidden ruleAccept
40791 glassfish755        jersey       server-hidden requestStart
```

Using Monitoring Scripting Client to Obtain Information About Enterprise Server DTrace Probes

If you have installed Monitoring Scripting Client, you can obtain detailed information about each Enterprise Server DTrace probe. This information includes a description of the probe, an explanation of what causes the probe to be fired, and a description of the probe's parameters. This information is *not* available through standard DTrace tools.

For more information, see the following sections in *Sun GlassFish Enterprise Manager Monitoring Scripting Client 3.0 Installation and Quick Start Guide*:

- “Downloading and Installing Monitoring Scripting Client”
- “Obtaining Information About Events That Provide Monitoring Data”

Sample D Scripts for Monitoring Enterprise Server

The sample D scripts in this section show how to use Enterprise Server DTrace probes to monitor Enterprise Server.

EXAMPLE 2 Tracing Network Connection Activity

This example traces network connection activity, such as the opening and closing of network connections, and the addition and removal of tasks from the task queue. Each time an operation on a network connection is performed, the probe for that operation and the probe's parameters are displayed on standard output.

Line breaks are added to the `printf()` statements in this example to enhance readability. In an actual D script, the added line breaks would cause the script to fail to compile.

```
#!/usr/sbin/dtrace -Zs

dtrace:::BEGIN {
    printf("%-Y     Starting DTrace...\n", walltimestamp);
    i=0;
}

glassfish*:kernel:connection-queue:connectionAcceptedEvent {
    listenerName=copyinstr(arg0);
    connection=arg1;
    address=copyinstr(arg2);
        printf("connectionAcceptedEvent(listenerName=%s,connection=%d,address=%s)",
        listenerName,connection,address);
}
glassfish*:kernel:connection-queue:connectionClosedEvent {
    listenerName=copyinstr(arg0);
        connection=arg1;
        printf("connectionClosedEvent(listenerName=%s,connection=%d)",listenerName,
        connection);
}
glassfish*:kernel:connection-queue:connectionConnectedEvent {
    listenerName=copyinstr(arg0);
        connection=arg1;
    address=copyinstr(arg2);
        printf("connectionConnectedEvent(listenerName=%s,connection=%d,address=%s)",
        listenerName,connection,address);
}
```

EXAMPLE 2 Tracing Network Connection Activity *(Continued)*

```
glassfish*:kernel:connection-queue:onTaskQueuedEvent {
        listenerName=copyinstr(arg0);
        task=copyinstr(arg1);
        printf("onTaskQueuedEvent(listenerName=%s,task=%s)",listenerName,task);
}

glassfish*:kernel:connection-queue:onTaskDequeuedEvent {
    listenerName=copyinstr(arg0);
        task=copyinstr(arg1);
        printf("onTaskDequeuedEvent(listenerName=%s,task=%s)",listenerName,task);
}

glassfish*:kernel:connection-queue:onTaskQueueOverflowEvent {
    listenerName=copyinstr(arg0);
    printf("onTaskQueueOverflowEvent(listenerName=%s)",listenerName);
}

glassfish*:kernel:connection-queue:setMaxTaskQueueSizeEvent {
        listenerName=copyinstr(arg0);
        size=arg1;
        printf("setMaxTaskQueueSizeEvent(listenerName=%s,size%d)",listenerName,size);
}

dtrace:::END {
    printf("%-Y     Finished DTrace of glassfish*...\n", walltimestamp);
}
```

This script can be run with a command similar to the following:

```
# /usr/sbin/dtrace -s http-conn.d
```

Information similar to the following is displayed.

```
dtrace: script 'http-conn.d' matched 9 probes
CPU     ID                     FUNCTION:NAME
  0      1                            :BEGIN 2009 Dec  2 18:19:27     Starting DTrace...

  0  40548 connection-queue:connectionAcceptedEvent connectionAcceptedEvent(
listenerName=admin-listener,connection=5676699,address=/129.146.11.144:50600)
  0  40549 connection-queue:onTaskQueuedEvent onTaskQueuedEvent(
listenerName=admin-listener,task=13872330)
  0  40550 connection-queue:onTaskDequeuedEvent onTaskDequeuedEvent(
listenerName=admin-listener,task=13872330)
  0  40549 connection-queue:onTaskQueuedEvent onTaskQueuedEvent(
listenerName=admin-listener,task=17098630)
  0  40550 connection-queue:onTaskDequeuedEvent onTaskDequeuedEvent(
```

EXAMPLE 2 Tracing Network Connection Activity *(Continued)*

```
listenerName=admin-listener,task=17098630)
  0  40549 connection-queue:onTaskQueuedEvent onTaskQueuedEvent(
listenerName=admin-listener,task=16525461)
  0  40550 connection-queue:onTaskDequeuedEvent onTaskDequeuedEvent(
listenerName=admin-listener,task=16525461)
  0  40548 connection-queue:connectionAcceptedEvent connectionAcceptedEvent(
listenerName=admin-listener,connection=13390280,address=/129.146.11.144:50601)
```

The script runs until a user types Ctrl-C to stop the script.

EXAMPLE 3 Tracing HTTP Requests to the Web Container

This example traces HTTP requests to the web container and calculates the time that is required to handle each request. Each time a request is handled, details of the request and the time in microseconds to handle the request are displayed on standard output. When the script is stopped, it displays a summary of the requests that were handled while the script was running.

```
#!/usr/sbin/dtrace -Zs

dtrace:::BEGIN {   printf ("%-20s \n", "Monitoring Requests");   total = 0; }

glassfish$1:web:http-service:requestStartEvent {
   self->ts = timestamp;
   trace(probename);
   @counts[probename]=count();
}

glassfish$1:web:http-service:requestEndEvent
/self->ts/
{
   trace(probename);
   printf ("%s %s %d\n", copyinstr(arg0), copyinstr(arg1), arg2);
   @counts[probename]=count();
   @time[execname] = quantize(timestamp - self->ts);
   printf ("time in microseconds: %d\n", (timestamp - self->ts));
   self->ts = 0;
   total++; }
dtrace:::END {   printf ("Total requests = %d\n", total); }
```

This script requires the process ID of the Enterprise Server process. The process ID can be obtained with the following command:

```
# jps | grep ASMain
1086 ASMain
```

In this example, the process ID of the Enterprise Server process is 1086.

EXAMPLE 3 Tracing HTTP Requests to the Web Container *(Continued)*

This script can be run with a command similar to the following:

```
# /usr/sbin/dtrace -s trace-web-requests.d 1086
```

Information similar to the following is displayed. For better readability, the spacing between columns is reduced.

```
dtrace: script 'trace-web-requests.d' matched 4 probes
CPU     ID                    FUNCTION:NAME
  0      1                          :BEGIN Monitoring Requests

  0  41085   http-service:requestStartEvent   requestStartEvent
  0  41084     http-service:requestEndEvent   requestEndEvent hello server 3418876592
time in microseconds: 70935041

  0  41085   http-service:requestStartEvent   requestStartEvent
  0  41084     http-service:requestEndEvent   requestEndEvent hello server 3419925040
time in microseconds: 2174591

  0  41085   http-service:requestStartEvent   requestStartEvent
  0  41084     http-service:requestEndEvent   requestEndEvent hello server 3418876592
time in microseconds: 13554111

  0  41085   http-service:requestStartEvent   requestStartEvent
  0  41084     http-service:requestEndEvent   requestEndEvent hello server 3419925040
time in microseconds: 2499509
```

The script runs until a user types Ctrl-C to stop the script. When the script stops, information similar to the following is displayed.

```
^C
  0      2                            :END Total requests = 4


  requestEndEvent                                                   4
  requestStartEvent                                                 4
  java
           value  ------------- Distribution ------------- count
         1048576 |                                         0
         2097152 |@@@@@@@@@@@@@@@@@@@@                     2
         4194304 |                                         0
         8388608 |@@@@@@@@@@                               1
        16777216 |                                         0
        33554432 |                                         0
        67108864 |@@@@@@@@@@                               1
       134217728 |                                         0
```