Maintenance Guide

/ ForgeRock Access Management 7.1.4

Latest update: 7.1.4

ForgeRock AS.
201 Mission St., Suite 2900
San Francisco, CA 94105, USA
+1 415-599-1100 (US)
www.forgerock.com

Abstract

Guide to performing maintenance tasks in ForgeRock® Access Management (AM). ForgeRock Access Management provides intelligent authentication, authorization, federation, and single sign-on functionality.

Table of Contents

Overview

This guide covers how to perform maintenance tasks in ForgeRock Access Management such as backing up and restoring, monitoring, and others.

This guide is written for anyone that sets up and maintains Access Management services for their organizations. This guide covers tasks and configurations you might repeat throughout the life cycle of a deployment in your organization.

Quick Start

Back up and Restore Configurations Learn how to back up and restore AM.	Monitor Instances Monitor AM through any of the included interfaces, such ForgeRock® Common REST, Prometheus, Graphite, and others.
Tune Instances Learn best practices about tuning your AM environment.	Enable Debug Logging Discover how to enable debug logging to capture additional information that is useful when troubleshooting AM.

About ForgeRock Identity Platform™ Software

ForgeRock Identity Platform™ serves as the basis for our simple and comprehensive Identity and Access Management solution. We help our customers deepen their relationships with their customers, and improve the productivity and connectivity of their employees and partners. For more information about ForgeRock and about the platform, see https://www.forgerock.com.

Chapter 1 Backing Up Configurations

During normal production operations, you rely on directory replication to maintain multiple, current copies of AM's configuration. To recover from the loss of a server or from a serious administrative error, back up directory data and configuration files.

To backup your external configuration directory server, see the chapter *Backup and Restore* in the *DS Maintenance Guide*.

To Back Up Instance Configuration Data

This procedure backs up the configuration files stored with the server. This backup is to be restored when rebuilding a failed server.

Consider the following when using this procedure:

- Refer to the documentation for your external directory server or work with your directory server administrator to back up and restore configuration data stored in the directory service. For more information about DS, see the chapter *Backup and Restore* in the *DS Maintenance Guide*.
- Do not restore configuration data from a backup of a different major version of AM. The structure of the configuration data can change from release to release.

Follow these steps for each AM server that you want to back up:

1. Stop AM or the container in which it runs.
2. Back up AM server files.

   This example uses the default configuration location, and excludes logs. $HOME is the home directory of the user who runs the web container where AM is deployed, and AM is deployed in Apache Tomcat under `openam`:

   ```
   $ cd $HOME
   $ zip --recurse-paths \
     AM-config-dir-backup-`date -u +%F-%H-%M`.zip \
     "openam" ".openamcfg/*" \
     --exclude "openam/var/debug/*" "openam/var/audit/*" \
       "openam/var/stats*" "openam/opends/*"
   ...
   $ ls AM-config-dir-backup-2020-08-01-12-00.zip
   AM-config-dir-backup-2020-08-01-12-00.zip
   ```

3. Start AM or the container in which it runs.

Chapter 2
Monitoring Instances

In certain deployments, such as Kubernetes, it can be useful to poll endpoints to check if an instance is running, and ready to handle requests. You can determine the status of an AM instance by using the following endpoints:

`/json/health/live`

Use the `live` endpoint to determine if AM instances are up and running.

If the instance is running the endpoint returns an HTTP status code of `200`. If not, it returns a `503` response.

For example, use the following **curl** command to determine when an AM instance is alive:

```
$ curl --include --retry 10 --retry-connrefused 'http://am.example.com:8080/am/json/health/live'
Warning: Transient problem: connection refused Will retry in 1 seconds. 10
Warning: retries left.
Warning: Transient problem: connection refused Will retry in 2 seconds. 9
Warning: retries left.
Warning: Transient problem: connection refused Will retry in 4 seconds. 8
Warning: retries left.
HTTP/1.1 200
X-Frame-Options: SAMEORIGIN
X-Content-Type-Options: nosniff
Cache-Control: private
Content-API-Version: resource=1.0
Content-Length: 0
Date: Mon, 09 Nov 2020 12:22:38 GMT
```

`/json/health/ready`

Use the `ready` endpoint to determine if AM instances are ready to process requests.

If the instance is ready to process requests, the endpoint returns an HTTP status code of `200`. If not, it returns a `503` response.

One of the readiness checks the endpoint performs is that the CTS queue is not overloaded. The endpoint returns an HTTP 503 error if the following are *both true*:

1. The CTS queue is more full than the configured threshold property (`org.forgerock.services.cts.queue.readiness.threshold`).

2. The CTS data store is available.

A queue that is more full than the threshold, even though there is a CTS store available to process it, could indicate that the health of the AM instance is below expectations. That is why the endpoint returns HTTP 503.

The default threshold is 90% full, but you can configure the threshold percentage by using the `org.forgerock.services.cts.queue.readiness.threshold` advanced property. For more information, see "Advanced Properties" in the *Reference*.

For example, use the following **curl** command to determine when an AM instance is ready to process requests:

```
$ curl --include --retry 10 --retry-connrefused 'http://am.example.com:8080/am/json/health/ready'
Warning: Transient problem: connection refused Will retry in 1 seconds. 10
Warning: retries left.
Warning: Transient problem: connection refused Will retry in 2 seconds. 9
Warning: retries left.
Warning: Transient problem: connection refused Will retry in 4 seconds. 8
Warning: retries left.
Warning: Transient problem: connection refused Will retry in 8 seconds. 7
Warning: retries left.
HTTP/1.1 200
X-Frame-Options: SAMEORIGIN
X-Content-Type-Options: nosniff
Cache-Control: private
Content-API-Version: resource=1.0
Content-Length: 0
Date: Mon, 09 Nov 2020 12:45:03 GMT
```

+ *Deprecated isAlive JSP Page*

AM provides a deprecated `isAlive.jsp` page, to check whether AM is up. Point your application to the file under the deployment URL, such as `https://openam.example.com:8443/openam/isAlive.jsp`.

If you get a success code (with `Server is ALIVE:` in the body of the page returned), then the instance is in operation.

The `isAlive.jsp` page is deprecated and will be removed in a future version. Update your environment to use the `live` and `ready` endpoints instead.

Tip

The endpoints and `isAlive.jsp` page do not require any authentication. You may want to restrict access to them by the public, and instead only allow access from internal infrastructure.

For more advanced monitoring services that will let you monitor metrics, such as authentication and authorization outcomes, token-related operations, CTS queues, or JVM usage, see:

- "JMX Monitoring"
- "Prometheus Monitoring"

- "Common REST Monitoring"
- "Graphite Monitoring"
- "MBean Monitoring (Legacy)"
- "SNMP Monitoring (Legacy)"

JMX Monitoring

You can configure AM to let you listen for Java Management eXtension (JMX) clients, by default on port 9999. Either use the AM console page under Configure > Global Services > Monitoring and make sure both Monitoring Status and Monitoring RMI interface status are both set to Enabled, or use the **ssoadm** command:

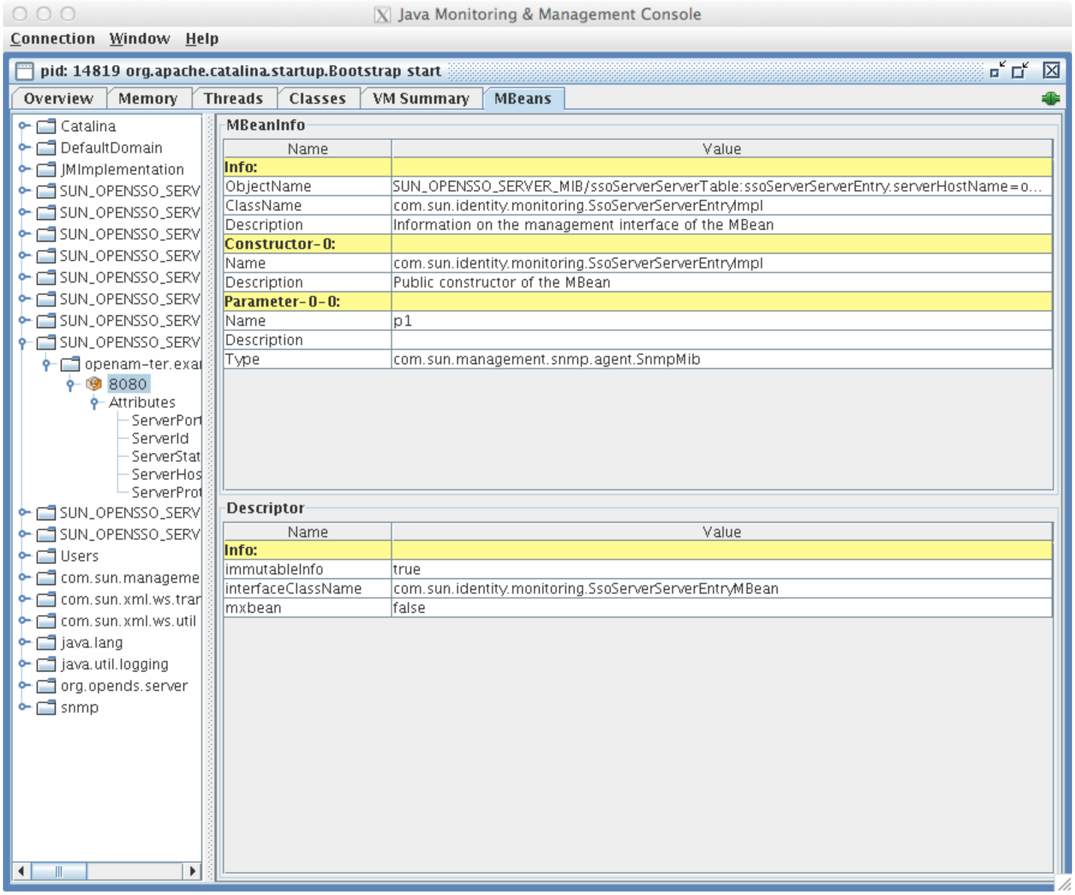
```
$ ssoadm \
set-attr-defs \
--servicename iPlanetAMMonitoringService \
--schematype Global \
--adminid uid=amAdmin,ou=People,dc=openam,dc=forgerock,dc=org \
--password-file /tmp/pwd.txt \
--attributevalues iplanet-am-monitoring-enabled=true \
iplanet-am-monitoring-rmi-enabled=true
```

A number of tools support JMX, including **jvisualvm** and **jconsole**. When you use **jconsole** to browse AM MBeans for example, the default URL for the AM running on the local system is `service:jmx:rmi:///jndi/rmi://localhost:9999/server`.

```
$ jconsole service:jmx:rmi:///jndi/rmi://localhost:9999/server &
```

You can also browse MBeans by connecting to your web application container, and browsing to the AM MBeans. By default, JMX monitoring for your container is likely to be accessible only locally, using the process ID.

JConsole Browsing MBeans



MBeanInfo

Name	Value
Info:	
ObjectName	SUN_OPENSSO_SERVER_MIB/ssoServerServerTable:ssoServerServerEntry.serverHostName=o…
ClassName	com.sun.identity.monitoring.SsoServerServerEntryImpl
Description	Information on the management interface of the MBean
Constructor-0:	
Name	com.sun.identity.monitoring.SsoServerServerEntryImpl
Description	Public constructor of the MBean
Parameter-0-0:	
Name	p1
Description	
Type	com.sun.management.snmp.agent.SnmpMib

Descriptor

Name	Value
Info:	
immutableInfo	true
interfaceClassName	com.sun.identity.monitoring.SsoServerServerEntryMBean
mxbean	false

Also see Monitoring and Management Using JMX for instructions on how to connect remotely, how to use SSL, and so forth.

Important

JMX has a limitation in that some Operations and CTS tables cannot be properly serialized from AM to JMX. As a result, only a portion of AM's monitoring information is available through JMX.

ForgeRock recommends using Prometheus, Graphite, or Common REST monitoring.

For monitoring metrics reference, see:

- "Monitoring Metrics"

Prometheus Monitoring

Prometheus is third-party software used for gathering and processing monitoring data. AM exposes an endpoint which Prometheus uses to gather metrics from the AM instance. For more information about installing and running Prometheus, see the Prometheus documentation.

When enabled, AM makes the Prometheus-formatted metrics available at the `/json/metrics/prometheus` endpoint.

Configure Prometheus to monitor the AM endpoint, using the `prometheus.yml` configuration file. For more information on configuring Prometheus, see the Prometheus configuration documentation.

> **Tip**
>
> Prometheus provides monitoring and processing for the information provided by AM, but further analysis and visualization may be desired. In this case, you can use tools such as Grafana to create customized charts and graphs based on the information collected by Prometheus. For more information on installing and running Grafana, see the Grafana website.

For monitoring metrics reference, see:

- "Monitoring Metrics"

To Enable the Prometheus Monitoring Interface

Before enabling Prometheus access to monitoring metrics, make sure that you have enabled monitoring. To enable monitoring, navigate to Configure > Global Services > Monitoring. Set the Monitoring Status to enabled, and then click Save Changes.

1. Navigate to Configure > Global Services > Monitoring.

2. Select the Secondary Configurations tab, and click prometheus.

3. Set prometheus to Enabled.

4. In the Authentication Type drop-down menu, select one of the following options:

    - **None**. Prometheus does not need to authenticate when accessing the endpoint.

    - **HTTP Basic**. Prometheus must authenticate using a username and a password when accessing the endpoint.

5. (Optional) If Prometheus must authenticate when accessing the endpoint, specify the Username and Password that it will use.

6. Click Save Changes.

Common REST Monitoring

Common REST is the ForgeRock® Common REST framework. AM exposes an endpoint which lets REST clients gather information about your AM installation, in JSON format.

When enabled, AM makes the Common REST-formatted metrics available at the `/json/metrics/api` endpoint.

For monitoring metrics reference, see:

- "Monitoring Metrics"

To Enable the Common REST Monitoring Interface

Before enabling Common REST access to monitoring metrics, make sure that you have enabled monitoring. To enable monitoring, navigate to Configure > Global Services > Monitoring. Set the Monitoring Status to enabled, and then click Save Changes.

1. Navigate to Configure > Global Services > Monitoring.

2. Select the Secondary Configurations tab, and click crest.

3. Set CREST to Enabled.

4. Click Save Changes.

Graphite Monitoring

Graphite is third-party software used for storing monitoring data, and rendering graphs of the data. For more information about installing and running Graphite, see the Graphite documentation.

For monitoring metrics reference, see:

- "Monitoring Metrics"

To Enable the Graphite Monitoring Interface

Before enabling Graphite access to monitoring metrics, make sure that you have enabled monitoring. To enable monitoring, navigate to Configure > Global Services > Monitoring. Set the Monitoring Status to enabled, and then click Save Changes.

1. Navigate to Configure > Global Services > Monitoring.

2. Select the Secondary Configurations tab, and click Add a Secondary Configuration.

3. Select Graphite Reporter.

4. Specify the Name and Hostname of the Graphite instance to push the metrics data to.

5. Click Create.

MBean Monitoring (Legacy)

Note

This functionality is labeled as *legacy*.

You can configure AM to let you access a web based view of AM MBeans on port 8082 where the core server runs, such as `http://openam.example.com:8082/`. Either use the console (Configure > Global Services > Monitoring), or use the **ssoadm** command:

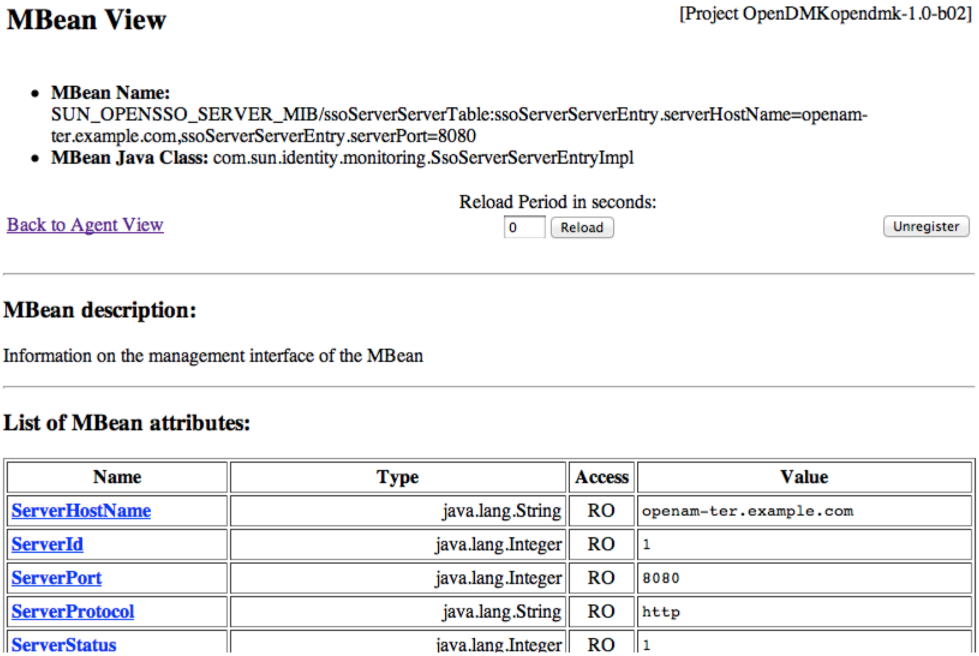
```
$ ssoadm \
set-attr-defs \
--servicename iPlanetAMMonitoringService \
--schematype Global \
--adminid uid=amAdmin,ou=People,dc=openam,dc=forgerock,dc=org \
--password-file /tmp/pwd.txt \
--attributevalues iplanet-am-monitoring-http-enabled=true
```

The default authentication file lets you authenticate over HTTP as user `demo`, password `Ch4ng31t`. The user name and password are kept in the file specified, with the password encrypted:

```
$ cat openam/security/openam_mon_auth
demo AQICMBCKlwx6G3vzK3TYYRbtTpNYAagVIPNP
```

You can encrypt a new password using the **ampassword** command. After changing the authentication file, you must restart AM for the changes to take effect.

MBeans in a Browser



MBean View [Project OpenDMKopendmk-1.0-b02]

- **MBean Name:** SUN_OPENSSO_SERVER_MIB/ssoServerServerTable:ssoServerServerEntry.serverHostName=openam-ter.example.com,ssoServerServerEntry.serverPort=8080
- **MBean Java Class:** com.sun.identity.monitoring.SsoServerServerEntryImpl

MBean description:

Information on the management interface of the MBean

List of MBean attributes:

Name	Type	Access	Value
ServerHostName	java.lang.String	RO	openam-ter.example.com
ServerId	java.lang.Integer	RO	1
ServerPort	java.lang.Integer	RO	8080
ServerProtocol	java.lang.String	RO	http
ServerStatus	java.lang.Integer	RO	1

SNMP Monitoring (Legacy)

> **Note**
>
> This functionality is labeled as *legacy*.

SNMP depends on labels known as Object Identifiers (OIDs). These are uniquely defined labels, organized in tree format. For AM, they are configured in a `.mib` file named `FORGEROCK-OPENAM-CTS.mib`, found inside the `/path/to/tomcat/webapps/openam/WEB-INF/lib/openam-mib-schema-7.1.4.jar` file of the AM deployment.

For detailed information on configured OIDs, see "SNMP CTS Object Identifiers".

With the OIDs in hand, you can set up an SNMP server to collect the data. You would also need SNMP utility commands with associated OIDs to measure the current state of a component.

To Enable the SNMP Monitoring Interface

1. Stop the AM instance or the container where it runs.

2. Download the AM 7.1.4 ZIP file from the ForgeRock BackStage download site.
3. Extract the contents of the ZIP file.
4. Navigate to the `/snmp` folder, and run the `opendmk.jar` installer file. For example:

   ```
   $ java -jar opendmk.jar
   ```
5. Accept the License Agreement.
6. Select the install directory you want to install to. For example: `/tmp/opendmk`.
7. Copy the `jdmkrt.jar` file from the `/lib` folder of the extracted archive to the AM `/WEB-INF/lib` folder. For example:

   ```
   $ cp /tmp/opendmk/OpenDMK-bin/lib/jdmkrt.jar /path/to/openam.war/WEB-INF/lib
   ```
8. Restart the AM instance or the container in which it runs.
9. Navigate to Configure > Global Services > Monitoring.
10. Set the Monitoring Status to enabled.
11. Set the Monitoring SNMP interface status property to Enabled. By default, AM will be set to let you listen on port 8085 for SNMP monitoring.
12. Click Save Changes.
13. Restart the AM instance for the change to take effect.

Once enabled, SNMP monitoring works over UDP by default. You may want to install one of many available network monitoring tools. For the purpose of this section, basic SNMP service and monitoring tools have been installed on a Unix-like system.

First, to verify the operation of SNMP on a GNU/Linux system, run the following command over port 8085 using SNMP version 2c:

```
 # snmpstatus -c public -v 2c localhost
```

The output should normally specify communications over UDP. If you get a `timeout` message, the SNMP service may not be running.

You can get the value for a specific OID. For example, the following command would retrieve the cumulative count for CTS create operations, over port 8085:

```
# snmpget -c public -v 2c :8085 enterprises.36733.1.2.3.3.1.1.1
```

If your version of the tool does not support the `enterprises` OID string, use `1.3.6.1.4.1` instead, as in `1.3.6.1.4.1.36733.1.2.3.3.1.1.1`.

For one view of the tree of OIDs, you can use the **snmpwalk** command. For example, the following command lists all OIDs related to CTS:

```
 # snmpwalk -c public -v 2c :8085 enterprises.36733.1.2.3
```

A number of CTS OIDs are listed with a `Counter64` value. As defined in *RFC 2578*, an OID so configured has a maximum value of `2^64 - 1`.

SNMP Monitoring for Sessions

You can monitor CTS-based session statistics over SNMP. AM records statistics for up to a configurable number of recent sessions. (You can configure the number in the AM console under Configuration > System > Monitoring. For details, see the system configuration reference section, "Monitoring" in the *Reference*.)

SNMP monitoring is not available for client-based sessions.

SNMP uses OIDs defined in a `.mib` file that specifies the statistics AM keeps for policy evaluation operations, the `FORGEROCK-OPENAM-SESSION.mib` file. This file is found inside the `/path/to/tomcat/webapps/openam/WEB-INF/lib/openam-mib-schema-7.1.4.jar` file of the AM deployment.

When monitoring is active, AM records statistics about both the numbers of internal, remote, and CTS sessions, and also the times taken to process sessions.

The statistics are all read-only. The base OID for session statistics is `enterprises.36733.1.2.1`. Times are expressed in nanoseconds rather than milliseconds, as many operations take less than one millisecond. The following table describes the values that you can read:

OIDs Used in SNMP Monitoring For Sessions

OID	Description	Syntax
`enterprises.36733.1.2.1.1.1`	Total number of current internal sessions	`Counter64`
`enterprises.36733.1.2.1.1.2`	Average time it takes to refresh an internal session	`Counter64`
`enterprises.36733.1.2.1.1.3`	Average time it takes to logout an internal session	`Counter64`
`enterprises.36733.1.2.1.1.4`	Average time it takes to destroy an internal session	`Counter64`
`enterprises.36733.1.2.1.1.5`	Average time it takes to set a property on an internal session	`Counter64`
`enterprises.36733.1.2.1.2.1`	Total number of current remote sessions	`Counter64`
`enterprises.36733.1.2.1.2.2`	Average time it takes to refresh a remote session	`Counter64`
`enterprises.36733.1.2.1.2.3`	Average time it takes to logout a remote session	`Counter64`
`enterprises.36733.1.2.1.2.4`	Average time it takes to destroy a remote session	`Counter64`
`enterprises.36733.1.2.1.2.5`	Average time it takes to set a property on a remote session	`Counter64`

OID	Description	Syntax
`enterprises.36733.1.2.1.3.1`	Total number of sessions currently in the Core Token Service (CTS)	`Counter64`
`enterprises.36733.1.2.1.3.2`	Average time it takes to refresh a CTS session	`Counter64`
`enterprises.36733.1.2.1.3.3`	Average time it takes to logout a CTS session	`Counter64`
`enterprises.36733.1.2.1.3.4`	Average time it takes to destroy a CTS session	`Counter64`
`enterprises.36733.1.2.1.3.5`	Average time it takes to set a property on a CTS session	`Counter64`

SNMP Monitoring for Policy Evaluation

You can monitor policy evaluation performance over SNMP. AM records statistics for up to a number of recent policy evaluation requests. (You can configure the number in the AM console under Configuration > System > Monitoring.) For details, see the reference section "Monitoring".

SNMP uses OIDs defined in the `.mib` file, `FORGEROCK-OPENAM-POLICY.mib`, found inside the `/path/to/tomcat/webapps/openam/WEB-INF/lib/openam-mib-schema-7.1.4.jar` file of the AM deployment. This file specifies the statistics AM keeps for policy evaluation operations.

When monitoring is active, AM records statistics about both the numbers and rates of policy evaluations performed, and also the time taken to process policy evaluations.

The statistics are all read-only. The base OID for policy evaluation statistics is `enterprises.36733.1.2.2.1`. The following table describes the values that you can read:

OIDs Used in SNMP Monitoring For Policy Evaluation

OID	Description	Syntax
`enterprises.36733.1.2.2.1.1.1`	Cumulative number of policy evaluations for specific resources (self)	`Counter64`
`enterprises.36733.1.2.2.1.1.2`	Average rate of policy evaluations for specific resources (self)	`Counter64`
`enterprises.36733.1.2.2.1.1.3`	Minimum rate of policy evaluations for specific resources (self)	`Counter64`
`enterprises.36733.1.2.2.1.1.4`	Maximum rate of policy evaluations for specific resources (self)	`Counter64`
`enterprises.36733.1.2.2.1.2.1`	Cumulative number of policy evaluations for a tree of resources (subtree)	`Counter64`

OID	Description	Syntax
`enterprises.36733.1.2.2.1.2.2`	Average rate of policy evaluations for a tree of resources (subtree)	`Counter64`
`enterprises.36733.1.2.2.1.2.3`	Minimum rate of policy evaluations for a tree of resources (subtree)	`Counter64`
`enterprises.36733.1.2.2.1.2.4`	Maximum rate of policy evaluations for a tree of resources (subtree)	`Counter64`
`enterprises.36733.1.2.2.1.1.2`	Average length of time to evaluate a policy for a specific resource (self)	`Counter64`
`enterprises.36733.1.2.2.2.1.2`	Slowest evaluation time for a specific resource (self)	`SnmpAdminString`
`enterprises.36733.1.2.2.1.2.2.1`	Average length of time to evaluate a policy for a tree of resources (subtree)	`Counter64`
`enterprises.36733.1.2.2.1.2.2.2`	Slowest evaluation time for a tree of resources (subtree)	`SnmpAdminString`
`enterprises.36733.1.2.2.1.3.1`	Slowest individual policy evaluation time overall	`SnmpAdminString`

Chapter 3
Changing Host Names

Changing host names associated to AM involves the following high-level steps:

- Adding the new host name to the Realm/DNS Aliases list.
- Exporting, editing, then importing the configuration.

  This step relies on the **ssoadm** command, which you install separately from AM as described in "*Setting Up Administration Tools*" in the *Installation Guide*.
- Stopping AM and editing configuration files.
- Removing the old host name from the Realm/DNS Aliases list.

Before you start, make sure you have a current backup of your current installation. See "*Backing Up Configurations*" for instructions.

To Add the New Host Name As an Alias

- In the AM console, go to Realms > *Realm Name*. Then click Properties, add the new host name to the Realm/DNS Aliases list, and then save your work.

To Export, Edit, and Import the Service Configuration

1. Export the service configuration:

   ```
   $ ssoadm \
    export-svc-cfg \
    --adminid uid=amAdmin,ou=People,dc=openam,dc=forgerock,dc=org \
    --encryptsecret myEncryptSecretString1234 \
    --password-file /tmp/pwd.txt \
    --outfile config.xml
   Service Configuration was exported.
   ```

   AM uses the value entered in `--encryptsecret` to encrypt passwords stored in the backup file. It can be any value, and is required when restoring a configuration.

2. Edit the service configuration file:

   - Change the fully qualified domain name, such as `openam.example.com`, throughout the file.
   - If you are changing the context path, such as `/openam`, then make the following changes:

    - Change the value of `com.iplanet.am.services.deploymentDescriptor`.
    - Change *contextPath* in the value of the `propertiesViewBeanURL="contextPath/auth/ACServiceInstanceList"`.
    - Change *contextPath* in the value of `propertiesViewBeanURL="contextPath/auth/ACModuleList"`.
    - Change the context path in a <Value> element that is a child of an <AttributeValuePair> element.
    - Change the context path where it occurs throughout the file in the full URL to AM, such as `http:&#47;&#47;openam.example.com:8080&#47;contextPath`.

  - If you are changing the port number, then change the value of `com.iplanet.am.server.port`.

    Also change the port number in `host:port` combinations throughout the file.

  - If you are changing the domain name, then change the cookie domain, such as `<Value>.example.com</Value>` throughout the file.

3. Import the updated service configuration:

```
$ ssoadm \
 import-svc-cfg \
 --adminid uid=amAdmin,ou=People,dc=openam,dc=forgerock,dc=org \
 --encryptsecret myEncryptSecretString1234 \
 --password-file /tmp/pwd.txt \
 --xmlfile config.xml
Directory Service contains existing data. Do you want to delete it? [y|N] y
Please wait while we import the service configuration...
Service Configuration was imported.
```

To Edit Configuration Files For the New Host Name

1. Stop AM or the web container where it runs.

2. Edit the boot properties file, such as `/home/user/openam/boot.json`, changing the fully qualified domain name (FQDN), port, and context path for AM as necessary.

3. If you are changing the context path, then move the folder containing AM configuration, such as `/home/user/openam/`, to match the new context path, such as `/home/user/openam2/`.

4. If you are changing the location or context path, change the name of the file in the `/home/user/.openamcfg` folder, such as `AMConfig_path_to_tomcat_webapps_openam_`, to match the new location and context path.

   Also edit the path name in the file to match the change you made when moving the folder.

5. Restart AM or the web container where it runs.

To Remove the Old Host Name As an Alias

1. In the AM console, go to realms > *Realm Name*, and click Properties.
2. Remove the old host name from the Realm/DNS Aliases list, and then save your work.

Chapter 4
Tuning Instances

This section covers key AM tunings to ensure smoothly performing access and federation management services, and to maximize throughput while minimizing response times.

> **Note**
>
> The recommendations provided here are guidelines for your testing rather than hard and fast rules for every situation. Said another way, the fact that a given setting is configurable implies that no one setting is right in all circumstances.
>
> The extent to which performance tuning advice applies depends to a large extent on your requirements, on your workload, and on what resources you have available. Test suggestions before rolling them out into production.

The suggestions in this section pertain to AM deployments with the following characteristics:

- The deployment has a dedicated DS server for the Core Token Service. The host running this directory server is a high-end server with a large amount of memory and multiple CPUs.
- The AM server is configured to use CTS-based sessions.

The following table summarizes the high-level tasks required to tune an AM instance:

Task	Resources
Tune General AM Settings	"Tuning Server Settings"
Tune Connectivity to LDAP Data Stores	"Tuning LDAP Connectivity"
Tune the JVM where AM Runs	"Tuning JVM Settings"
Tune the Configuration and User Caches	"Tuning Caching"

Tuning Server Settings

AM has a number of settings that can be tuned to increase performance.

Logging Settings

The following general points apply:

- Set debug logging level to `error`.
- Set container-level logging to a low level, such as `error` or `severe`.

Notification Settings

AM has two thread pools used to send notifications to clients. The Service Management Service (SMS) thread pool can be tuned in the AM console, under Configure > Server Defaults > SDK > Data Store:

SMS Notification Setting

Property	Default Value	Suggestions
Notification Pool Size	1	Specifies the size of the thread pool used to send notifications. A value of `1` causes notifications to be processed sequentially, avoiding any potential out-of-order conditions. In production, where configuration is unlikely to change often, keeping the default of `1` is recommended. (`com.sun.identity.sm.notification.threadpool.size`)

The session service has its own thread pool to send notifications to listeners about changes to CTS-based sessions. This is configured under Configure > Server Defaults > Session > Notification:

Session Service Notification Settings

Property	Default Value	Suggestions
Notification Pool Size	10	This is the size of the thread pool used to send notifications. In production this should be around 25-30. (`com.iplanet.am.notification.threadpool.size`)
Notification Thread Pool Threshold	5000	This is the maximum number of notifications in the queue waiting to be sent. The default value should be fine in the majority of installations. (`com.iplanet.am.notification.threadpool.threshold`)

Session Settings

The Session Service has additional properties to tune, which are configured under Configure > Server Defaults > Session > Session Limits. The following suggestion applies to deployments using CTS-based sessions:

Session Settings

Property	Default Value	Suggestion
Maximum Session Cache Size	5000	Maximum number of AM sessions to cache on the server.

Property	Default Value	Suggestion
		In production, this value can safely be set into the 100,000s. The maximum session cache size is really controlled by the maximum size of the JVM heap which must be tuned appropriately to match the desired session cache size. (org.forgerock.openam.session.service.access.persistence.caching.maxsize)

Policy Evaluation Settings

The AM policy engine places policies for evaluation on a queue in batches. Use ssoadm to optimize performance evaluation throughput by configuring the number of threads available for this queue.

This example increases the number of threads from the default value of 10 to 20:

```
./ssoadm set-entitlement-conf -u amadmin  -f pwd.txt -a evalThreadSize=20
```

Tuning LDAP Connectivity

AM instances use pools of connections when communicating to LDAP data stores. You can tune these connection pools to improve performance, and help with load balancing in the case of failover.

AM provides a global timeout setting for connections in a pool, and each store has properties for the maximum pool size, and in some cases, the minimum pool size.

AM will attempt to use as few connections to LDAP data stores as possible, down to the minimum pool value, if specified. Under heavy load, AM creates additional connections to the configured data stores, up to the maximum pool value. These connections are made to any of the available LDAP data stores that are configured for the relevant purpose.

When the load begins to drop, some of those connections become idle. If a connection is idle for longer than the configured connection idle time, AM closes the connection, until any specified minimum pool size is reached.

By closing idle connections and recreating them when needed, AM balances connections across all available LDAP servers, rather than keeping the entire pool connected to a single server.

Tuning the connection pool settings can increase performance, or make AM more responsive to LDAP data store outages.

To Configure Connection Pool Timeouts

1. To configure the timeout used for connections to LDAP stores:

   a. Open the bootstrapConfig.properties file in the AM classpath; for example, in /path/to/tomcat/webapps/openam/WEB-INF/classes/.

   b. Add, or update the following property, and set the idle timeout, in seconds:

      ```
      com.sun.am.ldap.connection.idle.seconds=300
      ```

2. You also need to configure the setting in the Advanced section of the server defaults, as follows:

   a. In the AM console, navigate to Configure > Server Defaults > Advanced.

   b. Add, or edit the following property, and set the idle timeout, in seconds:

      ```
      com.sun.am.ldap.connection.idle.seconds=300
      ```

3. Restart AM or the container in which it runs for these changes to take effect.

   After configuring the timeout for the stores, set the pool sizes assigned to the different stores:

   - "Tuning Configuration Store LDAP Connections"
   - "Tuning CTS Store LDAP Connections"
   - "Tuning Identity Store LDAP Connections"
   - "Tuning External Policy and Applications Store LDAP Connections"
   - "Tuning UMA Store LDAP Connections"
   - "Tuning Authentication Node/Module LDAP Connections"

Tuning Configuration Store LDAP Connections

To change LDAP configuration store settings, navigate to Deployment > Servers > *Server Name* > Directory Configuration.

LDAP Configuration Store Settings

Label	Default	Notes
Minimum Connection Pool	1	Property: `minConnectionPool`
Maximum Connection Pool	10	The default value of 10 is suitable for most cases; tuning this setting doesn't affect operational performance, only system startup. Property: `maxConnectionPool`

Tuning CTS Store LDAP Connections

You can increase the number of connections used for connecting to CTS to increase throughput.

One connection is reserved for cleanup of expired CTS tokens. The remaining connections are allocated for CTS operations such that the number is equal to a power of two. Because of this, you should set the maximum number of connections to `2^n+1`, as in 9, 17, 33, 65, and so forth.

The default maximum number of connections to the CTS is 10. To alter the default, navigate to Deployment > Servers > *Server Name* > CTS > CTS Token Store, and alter the Max Connections property.

You may need to click the Inherit Value property to unlock the value for editing.

Tip

You can also edit the Max Connections default globally by navigating to Configure > Server Defaults > CTS, click the CTS Token Store tab, and then alter the Max Connection property.

If you need to change the default CTS connection timeout, set the `org.forgerock.services.datalayer.connection.timeout.cts.async` property under Deployment > Servers > *Server Name* > Advanced.

Most CTS requests to the directory server are handled quickly, so the default timeout of 10 seconds is suitable in most cases.

You must restart AM or the container in which it runs for these changes to take effect.

Tuning External Policy and Applications Store LDAP Connections

To change external policy and application data store settings, navigate to Configure > Global Services > External Data Stores > Secondary Configurations > *Store Name*.

Note

Policy and application data is stored in the configuration data store if not configured separately. To manage the configuration store connection pool, see "Tuning Configuration Store LDAP Connections".

LDAP Policy and Application Store Settings

Label	Default	Notes
Minimum Connection Pool Size	1	Must be less than the maximum size to allow reaping to function. Property: `minimumConnectionPool`
Maximum Connection Pool Size	10	Property: `maximumConnectionPool`

Tuning Identity Store LDAP Connections

To change LDAP data store settings, navigate to Realms > *Realm Name* > Identity Stores > *Identity Store Name* in the AM console. Each store has its own connection pool—so each store needs its own tuning:

LDAP Identity Store Settings

Label	Default	Notes
LDAP Connection Pool Minimum Size	1	A good tuning value for this property is 10. Property: `sun-idrepo-ldapv3-config-connection_pool_min_size`
LDAP Connection Pool Maximum Size	10	The maximum LDAP connection pool size; a high tuning value for this property is 65, though you might well be able to reduce this for your deployment. Ensure your LDAP server can cope with the maximum number of clients across all the AM servers. Property: `sun-idrepo-ldapv3-config-connection_pool_max_size`

Tuning UMA Store LDAP Connections

To change the various UMA-related data store settings, navigate to Deployment > Servers > *Server Name*.

To increase the number of connections used for the various UMA-related data stores, navigate to Deployment > Servers > *Server Name* > UMA > *UMA Store Type*, and alter the Max Connections property.

You may need to click the Inherit Value property to unlock the value for editing.

Tip

You can also edit the Max Connections defaults globally by navigating to Configure > Server Defaults > UMA, click the relevant UMA store tab, and then alter the Max Connection property.

LDAP UMA Store Settings

Label	Default	Notes
UMA Resource Store > Max Connections	10	Property: `org.forgerock.services.resourcesets.store.max.connections`
UMA Audit Store > Max Connections	10	Property: `org.forgerock.services.umaaudit.store.max.connections`
Pending Requests Store > Max Connections	10	Property: `org.forgerock.services.pendingrequests.store.max.connections`
UMA Resource Labels Store > Max Connections	2	Property: `org.forgerock.services.uma.labels.store.max.connections`

Tuning Authentication Node/Module LDAP Connections

To change connection pool settings for the "LDAP Decision Node" in the *Authentication and Single Sign-On Guide* and LDAP Authentication Module in the *Authentication and Single Sign-On Guide*, in the AM console, go to Configure > Authentication > Core Attributes > Global Attributes.

LDAP Authentication Node/Module Settings

Label	Default	Notes
Default LDAP Connection Pool Size	1:10	The minimum and maximum LDAP connection pool used by the LDAP authentication node/module, separated by a colon (:) character. Use `10:65` for production AM instances. Property: `iplanet-am-auth-ldap-connection-pool-default-size`

Tuning JVM Settings

This section gives some initial guidance on configuring the JVM for running AM when the deployment has a dedicated CTS token store, and AM is configured to use CTS-based sessions.

These settings provide a strong foundation to the JVM before a more detailed garbage collection tuning exercise, or as best practice configuration for production:

Heap Size Settings

JVM Parameters	Suggested Value	Description
`-Xms` & `-Xmx`	At least 1 GB (2 GB with embedded DS), in production environments at least 2 GB to 3 GB. This setting depends on the available physical memory, and on whether a 32- or 64-bit JVM is used.	-
`-XX:MetaspaceSize` & `-XX:MaxMetaspaceSize`	Set both to 256 MB	Controls the size of the metaspace in the JVM
`-Dsun.net.client.defaultReadTimeout`	60000	Controls the read timeout in the Java HTTP client implementation This applies only to the Sun/ Oracle HotSpot JVM.
`-Dsun.net.client.defaultConnectTimeout`	High setting: 30000	Controls the connect timeout in the Java HTTP client implementation

JVM Parameters	Suggested Value	Description
	(30 seconds)	When you have hundreds of incoming requests per second, reduce this value to avoid a huge connection queue. This applies only to the Sun/ Oracle HotSpot JVM.

Security Settings

JVM Parameters	Suggested Value	Description
`-Dhttps.protocols`	`TLSv1.2`	Controls the protocols used for outbound HTTPS connections from AM. Specify one or more of the following values, separated by commas: • TLSv1.2 • TLSv1.3 This setting applies only to Sun/Oracle Java environments.
`-Dorg.forgerock.openam.ldap.secure.protocol.version`	`TLSv1.2`	Controls the protocol AM uses to connect to various external resources. Specify one or more of the following values, separated by commas: • TLSv1.2 • TLSv1.3 This setting overrides the default server value. For details, refer to Advanced Properties in the *Reference*

Garbage Collection Settings

JVM Parameters	Suggested Value	Description
`-verbose:gc`	-	Verbose garbage collection reporting.
`-Xlog:gc*`	`-Xlog:gc=info:file=$CATALINA_HOME/logs/gc-info.log`	Logs detailed information about garbage collection.

JVM Parameters	Suggested Value	Description
		When using the `-Xlog:gc` option, you can also specify the level, and output file.
`-XX:+HeapDumpOnOutOfMemoryError`	-	Out of Memory errors generate a heap dump automatically.
`-XX:HeapDumpPath`	`$CATALINA_HOME/logs/heapdump.hprof`	Location of the heap dump.
`-XX:+PrintClassHistogram`	-	Prints a heap histogram when the JVM receives a SIGTERM signal.

Tuning Caching

AM caches data to avoid having to query user and configuration data stores each time it needs the information. By default, AM makes use of LDAP persistent search to receive notification of changes to cached data. For this reason, caching works best when data are stored in a directory server that supports LDAP persistent search.

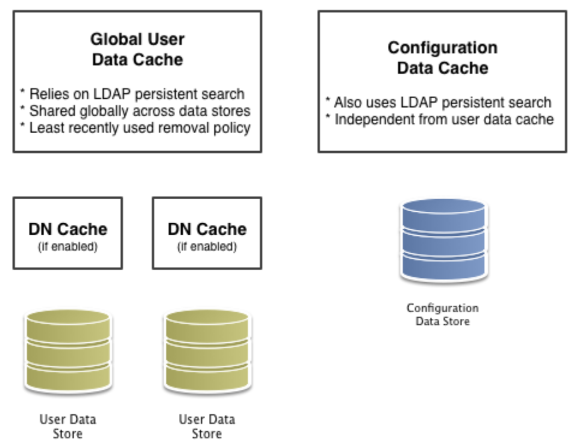
AM has two kinds of configurable cache on the server side; one for configuration data and one for user data. You can generally use the default settings for configuration data cache. This section covers the configuration choices available for caching user data.

AM implements the global user data cache for its user data stores.

The user data store also supports a *DN cache*, used to cache DN lookups that tend to occur in bursts during authentication. The DN cache can become out of date when a user is moved or renamed in the underlying LDAP store, events that are not always reflected in a persistent search result. You can enable the DN cache when the underlying LDAP store supports persistent search and `modDN` operations (that is, move or rename DN).

The following diagram depicts the two kinds of cache, and also the two types of caching available for user data:

Caches


Global User
Data Cache
* Relies on LDAP persistent search
* Shared globally across data stores
* Least recently used removal policy
Configuration
Data Cache
* Also uses LDAP persistent search
* Independent from user data cache
DN Cache
(if enabled)
DN Cache
(if enabled)
Configuration
Data Store
User Data
Store
User Data
Store

The rest of this section concerns mainly settings for global user data cache and for SDK clients. For a look at data store cache settings, see "LDAP Identity Store Settings".

Overall Server Cache Settings

By default, AM has caching enabled for both configuration data and user data. This setting is governed by the server property `com.iplanet.am.sdk.caching.enabled`, which is `true` by default. If you set this advanced property to `false`, you can enable caching independently for configuration data and for user data.

To Turn Off Global User Data Caching

> **Important**
>
> Disabling caching can have a **severe negative impact** on performance. This is because when caching is disabled, AM must query a data store each time it needs data.

If, however, you have at least one identity store that does not support LDAP persistent search, then you must disable the *global* cache for user data. Otherwise, user data caches cannot stay in sync with changes to user data entries:

1. In the AM console, go to Deployment > Servers > *Server Name* > Advanced.
2. Set the value of the `com.iplanet.am.sdk.caching.enabled` property to `false` to disable caching overall.
3. Set the value of the `com.sun.identity.sm.cache.enabled` property to `true` to enable configuration data caching.

All supported configuration data stores support LDAP persistent search, so it is safe to enable configuration data caching.

You must explicitly set this property to `true`, because setting the value of the property `com.iplanet.am.sdk.caching.enabled` to `false` in the previous step disables both user and configuration data caching.

4. Save your work.

5. AM starts persistent searches on user data stores when possible [1] in order to monitor changes. With user data store caching disabled, AM still starts the persistent searches, even though it no longer uses the results.

   Therefore, if you disable user data store caching, you should also disable persistent searches on identity stores in your deployment to improve performance. To disable persistent search on an identity store, go to Realms > *Realm Name* > Identity Stores > *Identity Store Name* > Persistent Search Controls and remove the value of the Persistent Search Base DN configuration property (leave it blank).

To Change the Maximum Size of Global User Data Cache

With a large user data store and active user base, the number of user entries in cache can grow large.

1. In the AM console, go to Configure > Server Defaults > SDK.

2. Change the value of SDK Caching Maximum Size.

   There is no corresponding setting for configuration data, as the number of configuration entries in a large deployment is not likely to grow nearly as large as the number of user entries.

Cache Settings

The table below provides a quick reference, primarily for user data cache settings.

Notice that many properties for configuration data cache have `sm` (for Service Management) in their names, whereas those for user data have `idm` (for Identity Management) in their names:

Cache Properties

Property	Description	Default	Applies To
`com.iplanet.am.sdk.cache.maxSize`	Maximum number of user entries cached.	10000	Server and SDK

[1] AM starts persistent searches on user data stores on directory servers that support the `psearch` control.

Property	Description	Default	Applies To
com.iplanet.am.sdk.caching.enabled	Whether to enable caching for both configuration data and also for user data. If true, this setting overrides com.sun.identity.idm.cache.enabled and com.sun.identity.sm.cache.enabled. If false, you can enable caching independently for configuration data and for user data using the aforementioned properties.	true	Server & SDK
com.iplanet.am.sdk.remote.pollingTime	How often in minutes the SDK client, such as a web or a Java agent should poll AM for modified user data entries. The SDK also uses this value to determine the age of the oldest changes requested. The oldest changes requested are 2 minutes older than this setting. In other words, by default the SDK polls for entries changed in the last 3 minutes. Set this to 0 or a negative integer to disable polling.	1 (minute)	SDK
com.sun.am.event.notification.expire.time	How long AM stores a given change to a cached entry, so that clients polling for changes do not miss the change.	30 (minutes)	Server only
com.sun.identity.idm.cache.enabled	If com.iplanet.am.sdk.caching.enabled is true, this property is ignored. Otherwise, set this to true to enable caching of user data.	false	Server & SDK
com.sun.identity.idm.cache.entry.default.expire.time	How many minutes to store a user data entry in the global user data cache.	30 (minutes)	Server & SDK
com.sun.identity.idm.cache.entry.expire.enabled	Whether user data entries in the global user data cache should expire over time.	false	Server & SDK
com.sun.identity.idm.remote.notification.enabled	Whether the SDK client, such as a web or a Java agent should register a notification listener for user data changes with the AM server. The SDK client uses the URL specified by com.sun.identity.client.notification.url to register the listener so that AM knows where to send notifications.	true	SDK

Property	Description	Default	Applies To
	If notifications cannot be enabled for some reason, then the SDK client falls back to polling for changes.		
`com.sun.identity.sm.cache.enabled`	If `com.iplanet.am.sdk.caching.enabled` is `true`, this property is ignored. Otherwise, set this to `true` to enable caching of configuration data. It is recommended that you always set this to `true`.	`false`	Server & SDK
`sun-idrepo-ldapv3-dncache-enabled`	Set this to `true` to enable DN caching of user data.	`false`	Server & SDK
`sun-idrepo-ldapv3-dncache-size`	Sets the cache size.	`1500`	Server & SDK

Chapter 5
Debug Logging

AM services capture a variety of information in debug logs. Unlike audit log records, debug log records are unstructured. Debug logs contain a variety of types of information that is useful when troubleshooting AM, including stack traces.

AM uses Logback as the handler for debug logging, making it easily customizable. For example, the level of debug log record output is configurable, as is the storage location and format.

AM lets you enable the debug log level for specific classes in the AM code base. This can be useful when you must turn on debug logging in a production system where you want to avoid excessive logging, but must gather messages when you reproduce a problem.

You can choose the level of logging from the following options:

`Off`

  No debug messages are logged.

`Error`

  Debug messages signifying that an error has occurred are logged.

`Warning`

  Debug messages signifying potentially harmful situations are logged.

`Information`

  Debug messages that contain coarse-grained information about the status of AM are logged.

`Debug`

  Debug messages that contain fine-grained information useful for troubleshooting AM are logged.

  This is the default level.

`Trace`

  All debug messages are logged.

Create *loggers* to specify the debug level for a class, and choose where the output is recorded. The logger used by a feature in AM is hierarchical, based on the class that is creating the debug messages. The most specific logger is used, which is the logger whose path most closely matches the class that is creating the log messages.

For example, if you knew there was an issue in an authentication module, you might enable trace-level debug logging in `org.forgerock.openam.authentication.modules`. If you are not sure where the problem lies, you may choose a broader option, for example `org.forgerock.openam.authentication`.

The least-specific, catch-all logger is named `ROOT`.

AM also logs information related to client interactions using the `org.apache.http.wire` and `org.apache.http.headers` appenders. The information they collect is useful, for example, when you are developing authentications scripts or when your environment requires STS transformations.

By default, these appenders are always set to the `Warning` level unless logging is disabled. For more information, see the `org.forgerock.allow.http.client.debug` advanced server property.

You can configure debug logging temporarily by using the AM console, or you can create a file in the AM classpath with persistent debug configuration. See the following procedures:

- "To Temporarily Enable Debug Logging with Logback.jsp"
- "To Configure Basic Debug Logging"

Temporary Debug Logging with Logback.jsp

Configuring `Logback.jsp` lets you temporarily capture debug messages until the next time AM or the container in which it runs is restarted.

To Temporarily Enable Debug Logging with Logback.jsp

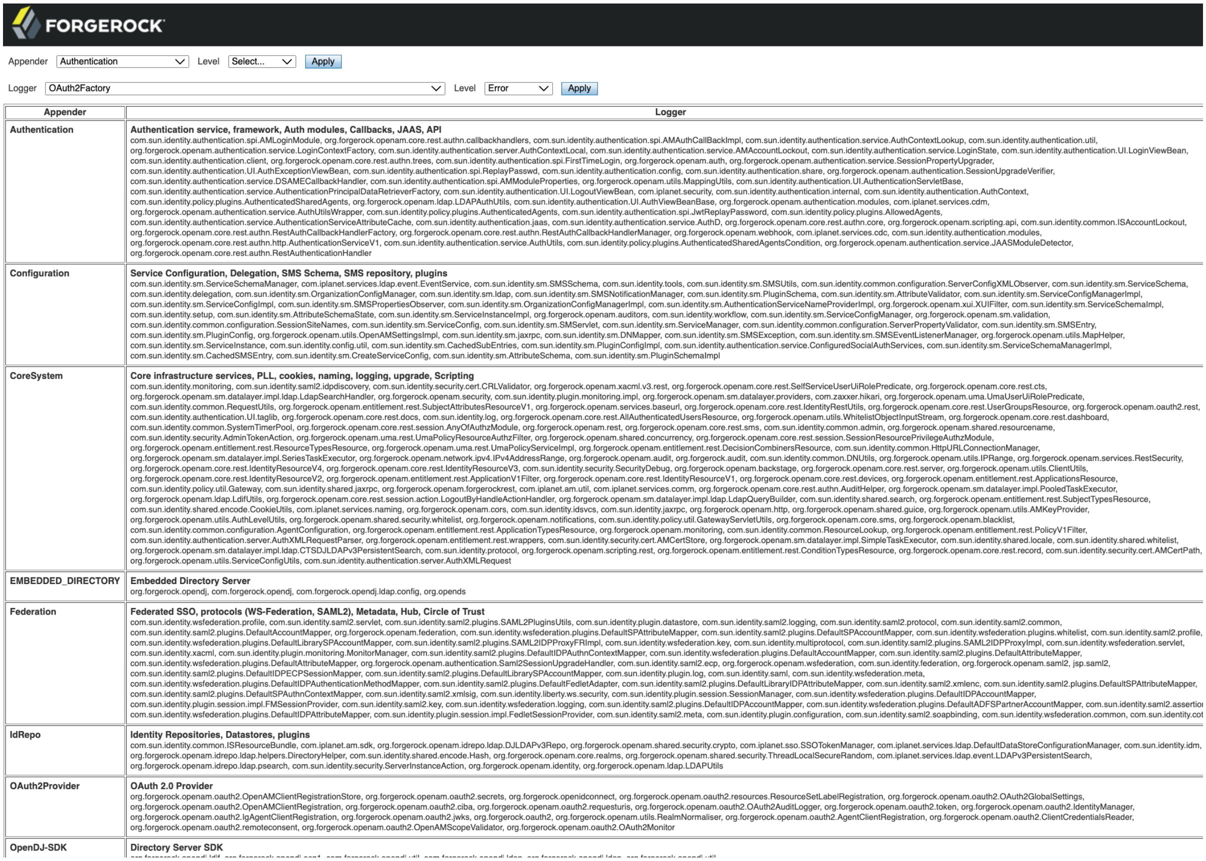
Follow these steps to view and set logging levels in `Logback.jsp`.

1. In the AM console, go to `Logback.jsp` in the root context of the AM installation, for example `https://openam.example.com:8443/openam/Logback.jsp`.

   No links to this page are provided in the AM console.

   > **Important**
   >
   > Only the `amAdmin` administrator account can access the `Logback.jsp` page and alter the debug settings; delegated administrators do not have access.

   The page displays all the appenders and their associated debug loggers:



Appender [Authentication] Level [Select...] Apply

Logger [OAuth2Factory] Level [Error] Apply

Appender	Logger
Authentication	**Authentication service, framework, Auth modules, Callbacks, JAAS, API** com.sun.identity.authentication.spi.AMLoginModule, org.forgerock.openam.core.rest.authn.callbackhandlers, com.sun.identity.authentication.spi.AMAuthCallBackImpl, com.sun.identity.authentication.service.AuthContextLookup, com.sun.identity.authentication.util, org.forgerock.openam.authentication.service.LoginContextFactory, com.sun.identity.authentication.server.AuthContextLocal, com.sun.identity.authentication.service.AMAccountLockout, com.sun.identity.authentication.service.LoginState, com.sun.identity.authentication.UI.LoginViewBean, com.sun.identity.authentication.client, org.forgerock.openam.core.rest.authn.trees, com.sun.identity.authentication.spi.FirstTimeLogin, org.forgerock.openam.auth, org.forgerock.openam.authentication.service.SessionPropertyUpgrader, com.sun.identity.authentication.UI.AuthExceptionViewBean, com.sun.identity.authentication.spi.ReplayPasswd, com.sun.identity.authentication.config, com.sun.identity.authentication.share, org.forgerock.openam.authentication.SessionUpgradeVerifier, com.sun.identity.authentication.service.DSAMECallbackHandler, com.sun.identity.authentication.spi.AMModuleProperties, org.forgerock.openam.utils.MappingUtils, com.sun.identity.authentication.UI.AuthenticationServletBase, com.sun.identity.authentication.service.AuthenticationPrincipalDataRetrieverFactory, com.sun.identity.authentication.UI.LogoutViewBean, com.iplanet.security, com.sun.identity.authentication.internal, com.sun.identity.authentication.AuthContext, com.sun.identity.policy.plugins.AuthenticatedSharedAgents, org.forgerock.openam.ldap.LDAPAuthUtils, com.sun.identity.authentication.UI.AuthViewBeanBase, org.forgerock.openam.authentication.modules, com.iplanet.services.cdm, org.forgerock.openam.authentication.service.AuthUtilsWrapper, com.sun.identity.policy.plugins.AuthenticatedAgents, com.sun.identity.authentication.spi.JwtReplayPassword, com.sun.identity.policy.plugins.AllowedAgents, com.sun.identity.authentication.service.AuthenticationServiceAttributeCache, com.sun.identity.authentication.jaas, com.sun.identity.authentication.service.AuthD, org.forgerock.openam.core.rest.authn.core, org.forgerock.openam.scripting.api, com.sun.identity.common.ISAccountLockout, org.forgerock.openam.core.rest.authn.RestAuthCallbackHandlerFactory, org.forgerock.openam.core.rest.authn.RestAuthCallbackHandlerManager, org.forgerock.openam.webhook, com.iplanet.services.cdc, com.sun.identity.authentication.modules, org.forgerock.openam.core.rest.authn.http.AuthenticationServiceV1, com.sun.identity.authentication.service.AuthUtils, com.sun.identity.policy.plugins.AuthenticatedSharedAgentsCondition, org.forgerock.openam.authentication.service.JAASModuleDetector, org.forgerock.openam.core.rest.authn.RestAuthenticationHandler
Configuration	**Service Configuration, Delegation, SMS Schema, SMS repository, plugins** com.sun.identity.sm.ServiceSchemaManager, com.iplanet.services.ldap.event.EventService, com.sun.identity.sm.SMSSchema, com.sun.identity.tools, com.sun.identity.sm.SMSUtils, com.sun.identity.common.configuration.ServerConfigXMLObserver, com.sun.identity.sm.ServiceSchema, com.sun.identity.delegation, com.sun.identity.sm.OrganizationConfigManager, com.sun.identity.sm.ldap, com.sun.identity.sm.SMSNotificationManager, com.sun.identity.sm.PluginSchema, com.sun.identity.sm.AttributeValidator, com.sun.identity.sm.ServiceConfigManagerImpl, com.sun.identity.sm.ServiceConfigImpl, com.sun.identity.sm.SMSPropertiesObserver, com.sun.identity.sm.OrganizationConfigManagerImpl, com.sun.identity.sm.AuthenticationServiceNameProviderImpl, org.forgerock.openam.xui.XUIFilter, com.sun.identity.sm.ServiceSchemaImpl, com.sun.identity.setup, com.sun.identity.sm.AttributeSchemaState, com.sun.identity.sm.ServiceInstanceImpl, org.forgerock.openam.auditors, com.sun.identity.workflow, com.sun.identity.sm.ServiceConfigManager, org.forgerock.openam.sm.validation, com.sun.identity.common.configuration.SessionSiteNames, com.sun.identity.sm.ServiceConfig, com.sun.identity.sm.SMServlet, com.sun.identity.sm.ServiceManager, com.sun.identity.common.configuration.ServerPropertyValidator, com.sun.identity.sm.SMSEntry, com.sun.identity.sm.PluginConfig, org.forgerock.openam.utils.OpenAMSettingsImpl, com.sun.identity.sm.jaxrpc, com.sun.identity.sm.DNMapper, com.sun.identity.sm.SMSException, com.sun.identity.sm.SMSEventListenerManager, org.forgerock.openam.utils.MapHelper, com.sun.identity.sm.ServiceInstance, com.sun.identity.config.util, com.sun.identity.sm.CachedSubEntries, com.sun.identity.sm.PluginConfigImpl, com.sun.identity.authentication.service.ConfiguredSocialAuthServices, com.sun.identity.sm.ServiceSchemaManagerImpl, com.sun.identity.sm.CachedSMSEntry, com.sun.identity.sm.CreateServiceConfig, com.sun.identity.sm.AttributeSchema, com.sun.identity.sm.PluginSchemaImpl
CoreSystem	**Core infrastructure services, PLL, cookies, naming, logging, upgrade, Scripting** com.sun.identity.monitoring, com.sun.identity.saml2.idpdiscovery, com.sun.identity.security.cert.CRLValidator, org.forgerock.openam.xacml.v3.rest, org.forgerock.openam.core.rest.SelfServiceUserUiRolePredicate, org.forgerock.openam.core.rest.cts, org.forgerock.openam.sm.datalayer.impl.ldap.LdapSearchHandler, org.forgerock.openam.security, com.sun.identity.plugin.monitoring.impl, org.forgerock.openam.sm.datalayer.providers, com.zaxxer.hikari, org.forgerock.openam.uma.UmaUserUiRolePredicate, com.sun.identity.common.RequestUtils, org.forgerock.openam.entitlement.rest.SubjectAttributesResourceV1, org.forgerock.openam.services.baseurl, org.forgerock.openam.core.rest.IdentityRestUtils, org.forgerock.openam.core.rest.UserGroupsResource, org.forgerock.openam.oauth2.rest, com.sun.identity.authentication.UI.taglib, org.forgerock.openam.core.rest.docs, com.sun.identity.log, org.forgerock.openam.core.rest.AllAuthenticatedUsersResource, org.forgerock.openam.utils.WhitelistObjectInputStream, org.forgerock.openam.core.rest.dashboard, com.sun.identity.common.SystemTimerPool, org.forgerock.openam.core.rest.session.AnyOfAuthzModule, org.forgerock.openam.rest, org.forgerock.openam.core.rest.sms, com.sun.identity.common.admin, org.forgerock.openam.shared.resourcename, com.sun.identity.security.AdminTokenAction, org.forgerock.openam.uma.rest.UmaPolicyResourceAuthzFilter, org.forgerock.openam.shared.concurrency, org.forgerock.openam.core.rest.session.SessionResourcePrivilegeAuthzModule, org.forgerock.openam.entitlement.rest.ResourceTypesResource, org.forgerock.openam.uma.rest.UmaPolicyServiceImpl, org.forgerock.openam.entitlement.rest.DecisionCombinersResource, com.sun.identity.common.HttpURLConnectionManager, org.forgerock.openam.sm.datalayer.impl.SeriesTaskExecutor, org.forgerock.openam.network.ipv4.IPv4AddressRange, org.forgerock.openam.audit, org.forgerock.audit, com.sun.identity.common.DNUtils, org.forgerock.openam.utils.IPRange, org.forgerock.openam.services.RestSecurity, org.forgerock.openam.core.rest.IdentityResourceV4, org.forgerock.openam.core.rest.IdentityResourceV3, com.sun.identity.security.SecurityDebug, org.forgerock.openam.backstage, org.forgerock.openam.core.rest.server, org.forgerock.openam.utils.ClientUtils, org.forgerock.openam.core.rest.IdentityResourceV2, org.forgerock.openam.entitlement.rest.ApplicationV1Filter, org.forgerock.openam.core.rest.IdentityResourceV1, org.forgerock.openam.core.rest.devices, org.forgerock.openam.entitlement.rest.ApplicationsResource, com.sun.identity.policy.util.Gateway, com.sun.identity.shared.jaxrpc, org.forgerock.openam.forgerockrest, com.iplanet.am.util, com.iplanet.services.comm, org.forgerock.openam.core.rest.authn.AuditHelper, org.forgerock.openam.sm.datalayer.impl.PooledTaskExecutor, org.forgerock.openam.ldap.LdifUtils, org.forgerock.openam.core.rest.session.action.LogoutByHandleActionHandler, org.forgerock.openam.sm.datalayer.impl.ldap.LdapQueryBuilder, com.sun.identity.shared.search, org.forgerock.openam.entitlement.rest.SubjectTypesResource, com.sun.identity.shared.encode.CookieUtils, com.iplanet.services.naming, org.forgerock.openam.cors, com.sun.identity.idsvcs, com.sun.identity.jaxrpc, org.forgerock.openam.http, org.forgerock.openam.shared.guice, org.forgerock.openam.utils.AMKeyProvider, org.forgerock.openam.utils.AuthLevelUtils, org.forgerock.openam.shared.security.whitelist, org.forgerock.openam.notifications, com.sun.identity.policy.util.GatewayServletUtils, org.forgerock.openam.core.sms, org.forgerock.openam.blacklist, com.sun.identity.common.configuration.AgentConfiguration, org.forgerock.openam.entitlement.rest.ApplicationTypesResource, org.forgerock.openam.monitoring, com.sun.identity.common.ResourceLookup, org.forgerock.openam.entitlement.rest.PolicyV1Filter, com.sun.identity.authentication.server.AuthXMLRequestParser, org.forgerock.openam.entitlement.rest.wrappers, com.sun.identity.security.cert.AMCertStore, org.forgerock.openam.sm.datalayer.impl.SimpleTaskExecutor, com.sun.identity.shared.locale, com.sun.identity.shared.whitelist, org.forgerock.openam.sm.datalayer.impl.ldap.CTSDJLDAPv3PersistentSearch, com.sun.identity.protocol, org.forgerock.openam.scripting.rest, org.forgerock.openam.entitlement.rest.ConditionTypesResource, org.forgerock.openam.core.rest.record, com.sun.identity.security.cert.AMCertPath, org.forgerock.openam.utils.ServiceConfigUtils, com.sun.identity.authentication.server.AuthXMLRequest
EMBEDDED_DIRECTORY	**Embedded Directory Server** org.forgerock.opendj, com.forgerock.opendj, com.forgerock.opendj.ldap.config, org.opends
Federation	**Federated SSO, protocols (WS-Federation, SAML2), Metadata, Hub, Circle of Trust** com.sun.identity.wsfederation.profile, com.sun.identity.saml2.servlet, com.sun.identity.saml2.plugins.SAML2PluginsUtils, com.sun.identity.plugin.datastore, com.sun.identity.saml2.logging, com.sun.identity.saml2.protocol, com.sun.identity.saml2.common, com.sun.identity.saml2.plugins.DefaultAccountMapper, org.forgerock.openam.federation, com.sun.identity.wsfederation.plugins.DefaultSPAttributeMapper, com.sun.identity.saml2.plugins.DefaultSPAccountMapper, com.sun.identity.wsfederation.plugins.whitelist, com.sun.identity.saml2.profile, com.sun.identity.wsfederation.plugins.DefaultLibrarySPAccountMapper, com.sun.identity.saml2.plugins.SAML2IDPProxyFRImpl, com.sun.identity.wsfederation.key, com.sun.identity.multiprotocol, com.sun.identity.saml2.plugins.SAML2IDPProxyImpl, com.sun.identity.wsfederation.servlet, com.sun.identity.xacml, com.sun.identity.plugin.monitoring.MonitorManager, com.sun.identity.saml2.plugins.DefaultIDPAuthnContextMapper, com.sun.identity.wsfederation.plugins.DefaultAccountMapper, com.sun.identity.saml2.plugins.DefaultAttributeMapper, com.sun.identity.wsfederation.plugins.DefaultAttributeMapper, org.forgerock.openam.authentication.Saml2SessionUpgradeHandler, com.sun.identity.saml2.ecp, org.forgerock.openam.wsfederation, com.sun.identity.federation, org.forgerock.openam.saml2, jsp.saml2, com.sun.identity.saml2.plugins.DefaultIDPECPSessionMapper, com.sun.identity.saml2.plugins.DefaultLibrarySPAccountMapper, com.sun.identity.plugin.log, com.sun.identity.saml, com.sun.identity.wsfederation.meta, com.sun.identity.wsfederation.plugins.DefaultIDPAuthenticationMethodMapper, com.sun.identity.saml2.plugins.DefaultFedletAdapter, com.sun.identity.saml2.plugins.DefaultLibraryIDPAttributeMapper, com.sun.identity.saml2.xmlenc, com.sun.identity.saml2.plugins.DefaultSPAttributeMapper, com.sun.identity.saml2.plugins.DefaultSPAuthnContextMapper, com.sun.identity.saml2.xmlsig, com.sun.identity.liberty.ws.security, com.sun.identity.plugin.session.SessionManager, com.sun.identity.wsfederation.plugins.DefaultIDPAccountMapper, com.sun.identity.plugin.session.impl.FMSessionProvider, com.sun.identity.saml2.key, com.sun.identity.wsfederation.logging, com.sun.identity.saml2.plugins.DefaultIDPAccountMapper, com.sun.identity.wsfederation.plugins.DefaultADFSPartnerAccountMapper, com.sun.identity.saml2.assertio com.sun.identity.wsfederation.plugins.DefaultIDPAttributeMapper, com.sun.identity.plugin.session.impl.FedletSessionProvider, com.sun.identity.saml2.meta, com.sun.identity.plugin.configuration, com.sun.identity.saml2.soapbinding, com.sun.identity.wsfederation.common, com.sun.identity.co
IdRepo	**Identity Repositories, Datastores, plugins** com.sun.identity.common.ISResourceBundle, com.iplanet.am.sdk, org.forgerock.openam.idrepo.ldap.DJLDAPv3Repo, org.forgerock.openam.shared.security.crypto, com.iplanet.sso.SSOTokenManager, com.iplanet.services.ldap.DefaultDataStoreConfigurationManager, com.sun.identity.idm, org.forgerock.openam.idrepo.ldap.helpers.DirectoryHelper, com.sun.identity.shared.encode.Hash, org.forgerock.openam.core.realms, org.forgerock.openam.shared.security.ThreadLocalSecureRandom, com.iplanet.services.ldap.event.LDAPv3PersistentSearch, org.forgerock.openam.idrepo.ldap.psearch, com.sun.identity.security.ServerInstanceAction, org.forgerock.openam.identity, org.forgerock.openam.ldap.LDAPUtils
OAuth2Provider	**OAuth 2.0 Provider** org.forgerock.openam.oauth2.OpenAMClientRegistrationStore, org.forgerock.openam.oauth2.secrets, org.forgerock.openidconnect, org.forgerock.openam.oauth2.resources.ResourceSetLabelRegistration, org.forgerock.openam.oauth2.OAuth2GlobalSettings, org.forgerock.openam.oauth2.OpenAMClientRegistration, org.forgerock.openam.oauth2.ciba, org.forgerock.openam.oauth2.requesturis, org.forgerock.openam.oauth2.OAuth2AuditLogger, org.forgerock.openam.oauth2.token, org.forgerock.openam.oauth2.IdentityManager, org.forgerock.openam.oauth2.IgAgentClientRegistration, org.forgerock.openam.oauth2.jwks, org.forgerock.oauth2, org.forgerock.openam.utils.RealmNormaliser, org.forgerock.openam.oauth2.AgentClientRegistration, org.forgerock.openam.oauth2.ClientCredentialsReader, org.forgerock.openam.oauth2.remoteconsent, org.forgerock.openam.oauth2.OpenAMScopeValidator, org.forgerock.openam.oauth2.OAuth2Monitor
OpenDJ-SDK	**Directory Server SDK** org.forgerock.opendj.ldif, org.forgerock.opendj.asn1, com.forgerock.opendj.util, com.forgerock.opendj.ldap, org.forgerock.opendj.ldap, org.forgerock.opendj.util

2. To set the logging level for all loggers that output to a particular appender:

   a. Select the name of the appender from the Appender drop-down list.

   b. Select the debug level from the Level drop-down list.

   c. Click Apply.

3. To set the logging level for a class or package:

   a. Select the name of the individual logger from the Logger drop-down list, or select the global `ROOT` logger to set the level for all loggers.

      The current debug level is shown in the Level field.

> **Note**
>
> Any scripts that create debug messages have their own logger, which is only created after the script has executed at least once.
>
> The name of the logger has the format: `scripts.script_type.script_UUID`.
>
> For example, `scripts.POLICY_CONDITION.9de3eb62-f131-4fac-a294-7bd170fd4acb`.

b. Select a new debug level from the Level drop-down list.

c. Click Apply.

When you apply any changes to the logger settings, a `Logger settings updated` message is shown at the top of the `Logback.jsp` page.

> **Important**
>
> Changes made in `Logback.jsp` apply immediately, but are not permanently stored. Restarting AM or the container in which it runs will reset the levels to defaults.
>
> You can configure the default settings that will be applied when AM starts up. See "Altering the Startup Debug Settings".

4. As soon as you have reproduced the problem you are investigating, return to the `Logback.jsp` page and revert the logger levels to the previous settings, to avoid filling up disk space.

Persistent Debug Logging with Logback.xml

Debug logging can be enabled and persisted in AM by configuring a `logback.xml` file. The file describes the classes to capture the debug messages for, and the destination, or *appender* where the output is stored. For more information about configuring Logback, see Logback configuration in the *Logback Documentation*.

This section contains the following procedures:

- "To Configure Basic Debug Logging"
- "To Output to stdout"
- "To Output to Multiple Locations"
- "To Format Log Files"
- "To Rotate Debug Logs with Logback.xml"

To Configure Basic Debug Logging

Perform the following steps to configure a basic, persistent debug logging setup in AM using a `logback.xml` file:

1. Create a `logback.xml` file in the AM classpath, for example in `/path/to/tomcat/webapps/openam/WEB-INF/classes/`.

2. In the `logback.xml` file, add an empty, top-level element named `configuration`.

   For example:

   ```
   <configuration>
   </configuration>
   ```

   This element contains the configuration of the loggers and appenders, covered in later steps.

   a. (Optional) To instruct AM to periodically check the `logback.xml` file for changes, and apply them to the running instance, add both a `scan` and a `scanPeriod` attribute to the `<configuration>` element. For example:

      ```
      <configuration scan="true" scanPeriod="30 seconds">
      </configuration>
      ```

      > **Tip**
      >
      > If AM is not configured to scan the `logback.xml` file for changes, you will need to reboot the instance in order to pick up any changes.
      >
      > You can set the `scanPeriod` attribute to a longer time period, for example one hour, so that rebooting a running system is not required when you need to alter the debugging level.

      For more information, see Automatically reloading configuration file upon modification in the *Logback Documentation*.

   b. (Optional) To troubleshoot issues when configuring debug logging using the `logback.xml` file, add a `debug` attribute, set to `true`, to the `<configuration>` element. For example:

      ```
      <configuration debug="true">
      </configuration>
      ```

      AM will record debug logging status information to the default log file for the container in which it is running. For example, in Tomcat, status messages about the configuration of logback will be recorded in the `Catalina.out` file.

      For more information, see Status data in the *Logback Documentation*.

3. Inside the `<configuration>` element, add the definition of one or more appenders. The following example appender logs messages to a file named `debug.out` in the default AM debug directory:

```
<configuration>
  <appender name="DEBUG.OUT" class="ch.qos.logback.core.FileAppender">
    <file>openam/var/debug/debug.out</file>
    <encoder>
      <pattern>%lo{5}: %d{ISO8601}: Thread[%t]: TransactionId[%X{transactionId}]%n%level: %m%n%ex</
pattern>
    </encoder>
  </appender>
</configuration>
```

The pattern in the above example creates debug log entries that are identical to the output produced by previous versions of AM, including the transaction ID to aid with tracking events as they occur throughout the system.

4. Inside the `<configuration>` element, add the definition of one or more loggers.

   Loggers specify which classes to capture debug messages from, including any sub-classes. They also specify the level of debug information to capture, and which of the specified appenders is used to store the output.

   The following example logger applies the `Debug` level to the `scripts.POLICY_CONDITION` class and its sub-classes. The output is recorded in the file specified in the `debug.out` appender, created in a previous step:

```
<configuration>
  <appender name="DEBUG.OUT" class="ch.qos.logback.core.FileAppender">
    <file>openam/var/debug/debug.out</file>
    <encoder>
      <pattern>%lo{5}: %d{ISO8601}: Thread[%t]: TransactionId[%X{transactionId}]%n%level: %m%n%ex</
pattern>
    </encoder>
  </appender>
  <logger name="scripts.POLICY_CONDITION" level="Debug" >
    <appender-ref ref="DEBUG.OUT" />
  </logger>
</configuration>
```

5. (Optional) Inside the `<configuration>` element, add a `root` catch-all logger, to specify the global level of debug logging to all classes that do not match any of the loggers created in the previous step:

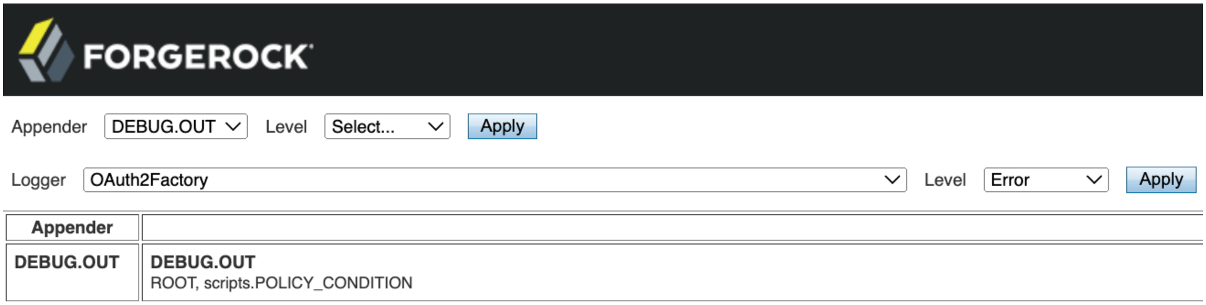
```
<configuration>
  <appender name="DEBUG.OUT" class="ch.qos.logback.core.FileAppender">
    <file>openam/var/debug/debug.out</file>
    <encoder>
      <pattern>%lo{5}: %d{ISO8601}: Thread[%t]: TransactionId[%X{transactionId}]%n%level: %m%n%ex</
pattern>
    </encoder>
  </appender>
  <logger name="scripts.POLICY_CONDITION" level="Debug" >
    <appender-ref ref="DEBUG.OUT" />
  </logger>
  <root level="Error">
    <appender-ref ref="DEBUG.OUT" />
  </root>
</configuration>
```

6. Save your changes.

   The changes will be applied the next time you reboot AM or the container in which it runs.

   > **Note**
   >
   > If you are editing an existing `logback.xml` that AM has already loaded, and contains the `scan="true"` attribute, you do not need to reboot. Instead wait for the amount of time specified in the `scanPeriod` attribute and the new configuration will be loaded into AM.

7. (Optional) To verify that the configuration from the `logback.xml` file has loaded, navigate to the `Logback.jsp` file, for example at `https://openam.example.com:8443/openam/Logback.jsp`, which reflects the configuration found:



Appender DEBUG.OUT Level Select... Apply

Logger OAuth2Factory Level Error Apply

Appender	
DEBUG.OUT	DEBUG.OUT ROOT, scripts.POLICY_CONDITION

   Note that any changes made in the `Logback.jsp` are temporary, and are not persisted to the `logback.xml` file.

To Output to stdout

Configure `logback.xml` to send logging to standard output. For example, for Apache Tomcat deployments, console output is typically redirected to the Tomcat logging file, `catalina.out`.

This example configuration captures all debug-level logging using the default `<root>` element, and redirects it to the STDOUT appender:

```
<configuration>
  <appender name="STDOUT" class="ch.qos.logback.core.ConsoleAppender">
    <encoder>
      <pattern>%d{HH:mm:ss.SSS} [%thread] %-5level %logger{36} - %msg%n</pattern>
    </encoder>
  </appender>
  <root level="Debug">
    <appender-ref ref="STDOUT" />
  </root>
</configuration>
```

1. To configure this example, create the following elements:

   - An `<appender>` that uses the `ch.qos.logback.core.ConsoleAppender` class.

   - A `<logger>`, or a `<root>` element as shown here, referencing the STDOUT appender.

2. Save your changes as described in "To Configure Basic Debug Logging".

3. Check that debug logging is now output to stdout. For example:

   ```
   tail -f $TOMCAT_HOME/logs/catalina.out
   ```

To Output to Multiple Locations

You can direct debug logging to more than one output location by defining multiple appenders and loggers. Note that you can define only one root element.

This example defines loggers for the `scripts.POLICY_CONDITION` and `scripts.OAUTH2_VALIDATE_SCOPE` classes and subclasses that output debug logging to file, using the DEBUG.OUT appender.

All debug-level logging is also directed to standard output using the STDOUT appender.

```
<configuration>
  <appender name="DEBUG.OUT" class="ch.qos.logback.core.FileAppender">
    <file>openam/var/debug/debug.out</file>
    <encoder>
      <pattern>%lo{5}: %d{ISO8601}: Thread[%t]: TransactionId[%X{transactionId}]%n%level: %m%n%ex</
pattern>
    </encoder>
  </appender>
  <appender name="STDOUT" class="ch.qos.logback.core.ConsoleAppender">
    <encoder>
      <pattern>%d{HH:mm:ss.SSS} [%thread] %-5level %logger{36} - %msg%n</pattern>
    </encoder>
  </appender>
  <logger name="scripts.OAUTH2_VALIDATE_SCOPE" level="Debug" >
      <appender-ref ref="DEBUG.OUT" />
    </logger>
  <logger name="scripts.POLICY_CONDITION" level="Debug" >
    <appender-ref ref="DEBUG.OUT" />
  </logger>
  <root level="Debug">
    <appender-ref ref="STDOUT" />
  </root>
</configuration>
```

1. To configure this example, create the following elements:

   - An `<appender>` that uses the `ch.qos.logback.core.FileAppender` class.
   - An `<appender>` that uses the `ch.qos.logback.core.ConsoleAppender` class.
   - A `<logger>` for each of the script types, referencing the DEBUG.OUT appender.
   - A `<logger>`, or a `<root>` element as shown here, referencing the STDOUT appender.

2. Save your changes as described in "To Configure Basic Debug Logging".

To Format Log Files

The `org.forgerock.openam.logback.JsonLayout` class extends Logback JSON layout functionality by adding the transaction ID to the JSON output.

This example shows how you can include the JsonLayout class to format your log files:

```
<configuration>
  <appender name="DEBUG.OUT" class="ch.qos.logback.core.FileAppender">
    <file>openam/var/debug/debug.out</file>
    <encoder>
      <pattern>%lo{5}: %d{ISO8601}: Thread[%t]: TransactionId[%X{transactionId}]%n%level: %m%n%ex</
pattern>
    </encoder>
  </appender>
  <appender name="STDOUT" class="ch.qos.logback.core.ConsoleAppender">
    <encoder>
      <pattern>%d{HH:mm:ss.SSS} [%thread] %-5level %logger{36} - %msg%n</pattern>
    </encoder>
  </appender>
  <logger name="scripts.OAUTH2_VALIDATE_SCOPE" level="Debug" >
      <appender-ref ref="DEBUG.OUT" />
    </logger>
  <logger name="scripts.POLICY_CONDITION" level="Debug" >
    <appender-ref ref="DEBUG.OUT" />
  </logger>
  <root level="Debug">
    <appender-ref ref="STDOUT" />
  </root>
</configuration>
```

1. To configure this example, create the following elements:

   - An `<appender>` that uses the `ch.qos.logback.core.rolling.RollingFileAppender` class.
   - An `<encoder>` that uses the `ch.qos.logback.core.encoder.LayoutWrappingEncoder` class.
   - A `<layout>` element that uses the `org.forgerock.openam.logback.JsonLayout` class.
   - A `<jsonFormatter>` element that uses the `ch.qos.logback.contrib.jackson.JacksonJsonFormatter` class.

2. Save and verify your changes as described in "To Configure Basic Debug Logging".

   The use of the JsonLayout class results in the addition of a `transactionId` at the top level of the log entry. For example:

   ```
   {
   "timestamp" : "2022-07-28 15:39:44.562",
   "level" : "DEBUG",
   "thread" : "http-nio-8080-exec-6",
   "mdc" : {
     "transactionId" : "eb0664cc-4615-461e-973a-64a1fc4f659a-34695"
   },
   "logger" : "org.forgerock.openam.rest.restAuthenticationFilter",
   "message" : "OpenAM SSO Token Session Module has successfully authenticated the client",
   "context" : "default",
   "transactionId" : "eb0664cc-4615-461e-973a-64a1fc4f659a-34695"
   }
   ```

To Rotate Debug Logs with Logback.xml

Logback provides built-in support for a number of log file rotation schemes, for example both time and size based rotation. If you have configured AM with a `logback.xml` file, you can configure log file rotation in the appenders, by performing the following steps:

1. Edit the `logback.xml` file you created in the AM classpath, for example in `/path/to/tomcat/webapps/openam/WEB-INF/classes/`.

   If you need to create the file, see "To Configure Basic Debug Logging".

2. In the `<configuration>` element, create an appender that uses the `ch.qos.logback.core.rolling.RollingFileAppender` class, for example:

   ```
   <appender name="DAILYLOG" class="ch.qos.logback.core.rolling.RollingFileAppender">
     <encoder>
       <pattern>%lo{5}: %d{ISO8601}: Thread[%t]: TransactionId[%X{transactionId}]%n%level: %m%n%ex</
   pattern>
     </encoder>
   </appender>
   ```

   Within the appender, specify whether to rotate based on time, and optionally also size, as follows:

   a. (Optional) To rotate the log files based only on time, add a `<rollingPolicy>` element to the appender, which uses the `ch.qos.logback.core.rolling.TimeBasedRollingPolicy` class.

      Include a `<fileNamePattern>` element that defines when the log files should roll over, and the naming convention.

      For example, the following appender rolls the log file over at midnight each day, and includes the date in the filename:

      ```
      <appender name="DAILYLOG" class="ch.qos.logback.core.rolling.RollingFileAppender">
        <rollingPolicy class="ch.qos.logback.core.rolling.TimeBasedRollingPolicy">
          <fileNamePattern>openam/var/debug/dailyLog.%d{yyyy-MM-dd}.log</fileNamePattern>
        </rollingPolicy>
        <encoder>
          <pattern>%lo{5}: %d{ISO8601}: Thread[%t]: TransactionId[%X{transactionId}]%n%level: %m%n%ex</
      pattern>
        </encoder>
      </appender>
      ```

   b. (Optional) To rotate the log files based on both time and size, add a `<rollingPolicy>` element to the appender, which uses the `ch.qos.logback.core.rolling.SizeAndTimeBasedRollingPolicy` class.

      Include a `<fileNamePattern>` element that defines when the log files should roll over, and where the counter for rolling over based on size occurs, specified by including `%i`. You must also include a `<maxFileSize>` element to define the maximum size of the log files.

      For example, the following appender rolls the log file over at midnight each day, but earlier if the file reaches 2 gigabytes in size, and includes the date in the filename:

```
<appender name="DAILYLOG2GB" class="ch.qos.logback.core.rolling.RollingFileAppender">
  <rollingPolicy class="ch.qos.logback.core.rolling.SizeAndTimeBasedRollingPolicy">
    <fileNamePattern>openam/var/debug/dailyLog2GB.%d{yyyy-MM-dd}-%i.log</fileNamePattern>
    <maxFileSize>2GB</maxFileSize>
  </rollingPolicy>
  <encoder>
    <pattern>%lo{5}: %d{ISO8601}: Thread[%t]: TransactionId[%X{transactionId}]%n%level: %m%n%ex</
pattern>
  </encoder>
</appender>
```

3. Save your changes.

   The changes will be applied the next time you reboot AM or the container in which it runs.

   > **Note**
   >
   > If you are editing an existing `logback.xml` that AM has already loaded, and contains the `scan="true"` attribute, you do not need to reboot. Instead wait for the amount of time specified in the `scanPeriod` attribute and the new configuration will be loaded into AM.

   Debug log files will roll over each night, and also if they reach the 2GB size limit. The file names will contain the date, and a counter to signify the order in which they were written.

Altering the Startup Debug Settings

You can configure the settings that will be applied when AM starts up and there is no `logback.xml` file present.

The settings specified as defaults will be reflected in the `Logback.jsp` file, for example at `https://openam.example.com:8443/openam/Logback.jsp`. However, they will not override the configuration contained with a custom `logback.xml` file.

This section contains the following procedures:

- "To Set the Default Debug Level"
- "To Set the Default Debug Directory"
- "To Combine Log Messages in a Single File"

To Set the Default Debug Level

Perform the following steps to set the default debug level used by all loggers when AM starts up:

1. In the AM console, go to Deployment > Servers > *Server Name* > General > Debugging.
2. Select an option from the Debug Level field.

The default level for debug logging is `Error`. This level is appropriate for normal production operations, in which case no debug log messages are expected.

Setting the debug log level to `Warning` increases the volume of messages. Setting the debug log level to `Message` dumps detailed trace messages.

Unless told to do so by qualified support personnel, do not use `Warning` or `Message` levels as a default in production. Instead, set the levels on a per-class basis. See "*Debug Logging*".

3. Save your changes.

   Changes are applied immediately.

To Set the Default Debug Directory

Perform the following steps to set the default directory used to store debug log files:

1. In the AM console, go to Deployment > Servers > *Server Name* > General > Debugging.
2. Enter a directory in which to store log files in the Debug Directory field.

   The default level for debug logging is `Error`. This level is appropriate for normal production operations, in which case no debug log messages are expected.

   The default value is `%BASE_DIR%/%SERVER_URI%/debug`. The `%BASE_DIR%` value equates to the folder containing the AM local configuration, for example `/path/to/openam`. The `%SERVER_URI%` value equates to the deployment URI used when deploying AM, for example `/openam`.

   Therefore, the default location for the debug logs resembles the following: `/path/to/openam/var/debug/`.

   > **Important**
   >
   > Ensure that the specified folder can be written to by the account that is running AM or the container in which it runs.

3. Save your changes.

   The changes will be applied the next time you reboot AM or the container in which it runs.

To Combine Log Messages in a Single File

Perform the following steps to log all debug messages to a single `debug.out` file:

1. In the AM console, go to Deployment > Servers > *Server Name* > General > Debugging.
2. Set the Merge Debug Files property to On.

3. Save your changes.

   Changes are applied immediately.

   All debug log messages will be written to a single debug file named `debug.out`. The file will be located in the directory specified in the Debug Directory property. See "To Set the Default Debug Directory".

Chapter 6

Recording Troubleshooting Information

The AM recording facility lets you initiate events to monitor AM while saving output that is useful when performing troubleshooting.

AM recording events save four types of information:

- AM debug logs
- Thread dumps, which show you the status of every active thread, with output similar to a JStack stack trace
- Important runtime properties
- The AM configuration

You initiate a recording event by invoking the **ssoadm start-recording** with a JSON file, or through a REST call with a JSON payload. The file or payload controls the amount of information AM records, the duration of the recording, and the location of recording output files.

+ *Recording Control File/Payload Reference*

Record Control File Configuration Properties

`issueID`

Type: Number

Required. The issue identifier—a positive integer stored internally as a Java `long` data type. A case number is a good choice for the `issueID` value.

The `issueID` is a component of the path at which recorded information is stored. See "Retrieving Recording Information" for more information.

`referenceID`

Type: String

Required. A second identifier for the recording event. Use this property to segregate multiple recording events for the same issue.

The `referenceID` is a component of the path at which recorded information is stored. See "Retrieving Recording Information" for more information.

Note that spaces are not allowed in the `referenceID` value.

`Description`

Type: String

Required. A textual description of the recording event.

`zipEnable`

Type: Boolean

Required. Whether to compress the output directory into a zip file when recording has stopped.

`configExport`

Type: Object

Required. An object containing the following properties:

`enable`

Type: Boolean

Required. Whether to export the AM configuration upon completion of the recording event. Exporting the AM configuration is a best practice, because it is extremely useful to have access to the configuration when troubleshooting.

`password`

Type: String

Required if `enable` is `true`. A key required to import the exported configuration. The key is used the same way that the **ssoadm export-svc-cfg** command uses the `-e` argument.

`sharePassword`

Type: Boolean

Required if `enable` is `true`. Whether to show the `password` value in the **ssoadm start-recording**, **ssoadm get-recording-status**, and **ssoadm stop-recording** output, and in the `info.json` file, which is output during recording events, and which contains runtime properties.

`debugLogs`

Type: Object

Required. An object containing the following properties:

`debugLevel`

Type: String

Required. The debug level to set for the recording event. Set the value of `debugLevel` to `MESSAGE` to get the most troubleshooting information from your recording period. Other acceptable but less commonly used values are `ERROR` and `WARNING`.

`autoStop`

Type: Object

Optional. Contains another object used to specify an event that automatically ends a recording period. For time-based termination, specify a `time` object; for termination based on uncompressed file size, specify a `fileSize` object. If you specify both `time` and `fileSize` objects, the event that occurs first causes recording to stop.

Specifying `fileSize` and `time` objects is a best practice, because it ensures that the recorded output does not occupy a larger than expected amount of space on your file system, and that recording events end in a timely fashion.

`time`

Type: Object

Optional; must be specified in the `autoStop` object if `fileSize` is not specified. Configures a recording period to terminate recording after this amount of time.

`timeUnit`

Type: String

Required. Acceptable values are `MILLISECONDS`, `SECONDS`, `MINUTES`, `HOURS`, and `DAYS`.

`value`

Type: Numeric

Required. Values in `MILLISECONDS` are rounded down to the second. The minimum acceptable value for `autoStop` is one second.

`fileSize`

Type: Object

Optional; must be specified in the `autoStop` object if `time` is not specified. Configures a recording period to terminate after the aggregate size of uncompressed debug logs has reached this size.

`sizeUnit`

Type: String

Required. Acceptable values are `B`, `KB`, `MB`, and `GB`.

`value`

Type: Numeric

Required.

`threadDump`

Type: Object

Required. An object containing the following properties:

`enable`

Type: Boolean

Required. Whether to dump threads during the recording event. Thread dumps are especially useful when troubleshooting performance issues and issues with unresponsive servers.

`delay`

Type: Object

Required if `enable` is `true`. Contains another object used to specify an interval at which thread dumps are taken. The initial thread dump is taken at the start of the recording event; subsequent thread dumps are taken at multiples of the `delay` interval.

`timeUnit`

Type: String

Required. Acceptable values are `MILLISECONDS`, `SECONDS`, `MINUTES`, `HOURS`, and `DAYS`.

`value`

Type: Numeric

Required. The minimum acceptable value is one second. Time units that are smaller than seconds, such as `MILLISECONDS`, are rounded to the closest second.

+ *Recording Control File/Payload Example*

```
{
  "issueID": 103572,
  "referenceID": "policyEvalFails",
  "description": "Troubleshooting artifacts in support of case 103572",
  "zipEnable": true,
  "configExport": {
    "enable": true,
    "password": "5x2RR70",
    "sharePassword": false
  },
  "debugLogs": {
    "debugLevel": "MESSAGE",
    "autoStop": {
      "time": {
        "timeUnit": "SECONDS",
        "value": 15
      },
      "fileSize": {
        "sizeUnit": "GB",
        "value": 1
      }
    }
  },
  "threadDump": {
    "enable": true,
    "delay": {
      "timeUnit": "SECONDS",
      "value": 5
    }
  }
}
```

The recording control file properties in the preceding example affect the recording output as follows:

Recording Control File Example Properties and Their Effect on Recording Behavior

Recording Control File Property	Value	Effect
`issueID`, `referenceID`	`103572`, `policyEvalFails`	Recording output is stored at the path `*debugFileLocation*/record/103572/policyEvalFails_*timestamp*.zip`. For more information about the location of recording output, see "Retrieving Recording Information".
`Description`	`Troubleshooting artifacts in support of case 103572`	No effect.
`zipEnable`	`true`	Recording output is compressed into a ZIP file.
`configExport` / `enable`	`true`	The AM configuration is exported at the start of the recording event.

Recording Control File Property	Value	Effect
`configExport` / `password`	`5x2RR70`	Knowledge of this password will be required to access the AM configuration that was saved during recording.
`configExport` / `sharePassword`	`false`	The password is not displayed in output messages displayed during the recording event or in the `info.json` file.
`debugLogs` / `debugLevel`	`MESSAGE`	Recording enables message-level debug logs during the recording event.
`debugLogs` / `autoStop` / `time`	`SECONDS`, `15`	Because both the `time` and `fileSize` properties are set, recording stops after 15 seconds, or after the size of the debug logs exceeds 1 GB, whichever occurs first.
`debugLogs` / `autoStop` / `fileSize`	`GB`, `1`	Because both the `time` and `fileSize` properties are set, recording stops after 15 seconds, or after the size of the debug logs exceeds 1 GB, whichever occurs first.
`threadDump` / `enable`	`true`	Thread dumps are taken throughout the recording event.
`threadDump` / `delay`	`SECONDS`, `5`	The first thread dump is taken when the recording event starts. Additional thread dumps are taken every five seconds hence.

The following table shows different tasks related to recording troubleshooting information:

Task or Requirement	Resources
Start and Stop Recording Information Use the **ssoadm** command or REST calls to start and stop recording information. You can also check if there are active recording events using REST ("Getting the Status of a Recording Event (REST)").	• "Starting and Stopping Recording (ssoadm)" • "Starting and Stopping Recording (REST)"
Retrieve Information AM stores the troubleshooting information you gathered, so it is ready to be sent to ForgeRock Support representatives.	• "Retrieving Recording Information"

Starting and Stopping Recording (ssoadm)

Start AM recording with the **ssoadm start-recording** command. For example:

```
$ ssoadm \
start-recording \
--servername https://openam.example.com:8443/openam \
--adminid uid=amAdmin,ou=People,dc=openam,dc=forgerock,dc=org \
--password-file /tmp/pwd.txt \
--jsonfile recording.json
{
  "recording": true,
  "record": {
    "issueID": 103572,
    "referenceID": "policyEvalFails",
    "description": "Record everything",
    "zipEnable": false,
    "threadDump": {
      "enable": true,
      "delay": {
        "timeUnit": "SECONDS",
        "value": 5
      }
    },
    "configExport": {
      "enable": true,
      "password": "admin password",
      "sharePassword": true
    },
    "debugLogs": {
      "debugLevel": "message",
      "autoStop": {
        "time": {
          "timeUnit": "MILLISECONDS",
          "value": 15000
        },
        "fileSize": {
          "sizeUnit": "KB",
          "value": 1048576
        }
      }
    },
    "status": "RUNNING",
    "folder": "/home/openam/debug/record/103572/policyEvalFails/"
  }
}
```

Note

The **ssoadm** command output in the preceding example is shown in indented format for ease of reading. The actual output is *not* indented.

In the preceding **ssoadm start-recording** command example, the `recording.json` file specifies the information to be recorded and under what conditions recording automatically terminates.

An active recording event stops when:

- You explicitly tell AM to stop recording by executing the **ssoadm stop-recording** command. See the "ssoadm" in the *Reference* for details about this command.
- Another **ssoadm start-recording** command is sent to AM that specifies an issue ID that differs from the active recording event's issue ID. In this case, the initial recording session terminates and the

new recording event starts. Note that you can determine whether an AM recording event is active by using the **ssoadm get-recording-status** command.

- A timer configured in the recording control file determines that the maximum amount of time for the recording event has been reached.
- A file size monitor configured in the recording control file determines that the maximum amount of information in debug logs has been reached.

Starting and Stopping Recording (REST)

To start a recording event, perform an HTTP POST using the `/json/records` endpoint, specifying the `_action=start` parameter in the URL. Specify a JSON payload identical in format to the input file for the **ssoadm start-recording** command.

You must authenticate to AM as an administrative user to obtain an SSO token prior to calling the `/json/records` REST endpoint. You then pass the SSO token in the `iPlanetDirectoryPro` header as proof of authentication.

```
$ curl \
--request POST \
--header "Content-Type: application/json" \
--header "iPlanetDirectoryPro: AQIC5..." \
--header "Accept-API-Version: resource=1.0" \
--data ' {
  "issueID": 103572,
  "referenceID": "policyEvalFails",
  "description": "Troubleshooting artifacts in support of case 103572",
  "zipEnable": true,
  "configExport": {
   "enable": true,
   "password": "5x2RR70",
   "sharePassword": false
  },
  "debugLogs": {
   "debugLevel": "MESSAGE",
   "autoStop": {
    "time":  {
     "timeUnit": "SECONDS",
     "value": 15
    },
    "fileSize": {
     "sizeUnit": "GB",
     "value": 1
    }
   }
  },
  "threadDump" : {
   "enable": true,
   "delay" :  {
    "timeUnit": "SECONDS",
    "value": 5
   }
  }
```

```
 }' \
https://openam.example.com:8443/openam/json/records?_action=start
{
    "recording":true,
    "record":{
        "issueID":103572,
        "referenceID":"policyEvalFails",
        "description":"Troubleshooting artifacts in support of case 103572",
        "zipEnable":true,
        "threadDump":{
            "enable":true,
            "delay":{
                "timeUnit":"SECONDS",
                "value":5
            }
        },
        "configExport":{
            "enable":true,
            "password":"xxxxxx",
            "sharePassword":false
        },
        "debugLogs":{
            "debugLevel":"message",
            "autoStop":{
                "time":{
                    "timeUnit":"MILLISECONDS",
                    "value":15000
                },
                "fileSize":{
                    "sizeUnit":"KB",
                    "value":1048576
                }
            }
        },
        "status":"RUNNING",
        "folder":"/opt/demo/openam/config/openam/debug/record/103572/policyEvalFails/"
    }
}
```

The **curl** command output is indented for ease of reading. The actual output is *not* indented, and the actions available from the `/json/records` endpoint do not support the `_prettyPrint` parameter.

To **stop** a recording event, perform an HTTP POST using the `/json/records` endpoint, specifying the `_action=stop` parameter in the URL:

```
$ curl \
--request POST \
--header "iPlanetDirectoryPro: AQIC5..." \
--header "Accept-API-Version: resource=1.0" \
https://openam.example.com:8443/openam/json/records?_action=stop
```

If there is no active recording event, AM returns a 400 error code.

If there is an active recording event, output similar to the following appears:

```
{
  "recording": false,
  "record": {
    "issueID": 103572,
    "referenceID": "policyEvalFails",
    "description": "Troubleshooting artifacts in support of case 103572",
    "zipEnable": true,
    "threadDump": {
      "enable": true,
      "delay": {
        "timeUnit": "SECONDS",
        "value": 5
      }
    },
    "configExport": {
      "enable": true,
      "password": "xxxxxx",
      "sharePassword": false
    },
    "debugLogs": {
      "debugLevel": "message",
      "autoStop": {
        "time": {
          "timeUnit": "MILLISECONDS",
          "value": 15000
        },
        "fileSize": {
          "sizeUnit": "KB",
          "value": 1048576
        }
      }
    },
    "status": "STOPPED",
    "folder": "/opt/demo/openam/config/openam/debug/record/103572/policyEvalFails/"
  }
}
```

Getting the Status of a Recording Event (REST)

To get the status of a recording event, perform an HTTP POST using the `/json/records` endpoint, specifying the `_action=status` parameter in the URL:

```
$ curl \
--request POST \
--header "iPlanetDirectoryPro: AQIC5..." \
--header "Accept-API-Version: resource=1.0" \
https://openam.example.com:8443/openam/json/records?_action=status
```

If there is no active recording event, the following output appears:

```
{
    "recording":false
}
```

If there is an active recording event, output similar to the following appears:

```
{
    "recording":true,
    "record":{
        "issueID":103572,
        "referenceID":"policyEvalFails",
        "description":"Troubleshooting artifacts in support of case 103572",
        "zipEnable":true,
        "threadDump":{
            "enable":true,
            "delay":{
                "timeUnit":"SECONDS",
                "value":5
            }
        },
        "configExport":{
            "enable":true,
            "password":"xxxxxx",
            "sharePassword":false
        },
        "debugLogs":{
            "debugLevel":"message",
            "autoStop":{
                "time":{
                    "timeUnit":"MILLISECONDS",
                    "value":15000
                },
                "fileSize":{
                    "sizeUnit":"KB",
                    "value":1048576
                }
            }
        },
        "status":"RUNNING",
        "folder":"/opt/demo/openam/config/openam/debug/record/103572/policyEvalFails/"
    }
}
```

Retrieving Recording Information

Information recorded by AM is stored at the path `debugFileLocation/record/issueID/referenceID`. For example, if the debug file location is `/home/openam/debug`, the issue ID `103572`, and the reference ID `policyEvalFails`, the path containing recorded information is `/home/openam/debug/record/103572/policyEvalFails`.

When there are multiple recording events with the same `issueID` and `referenceID`, AM appends a timestamp to the `referenceID` of the earliest paths. For example, multiple recording events for issue ID `103572` and reference ID `policyEvalFails` might be stored at the following paths:

- Most recent recording: `debugFileLocation/record/103572/policyEvalFails`
- Next most recent recording: `debugFileLocation/record/103572/policyEvalFails_2015-10-24-11-48-51-902-PDT`
- Earliest recording: `debugFileLocation/record/103572/policyEvalFails_2015-08-10-15-15-10-140-PDT`

AM compresses the output from recording events when you set the `zipEnable` property to `true`. The output file can be found at the path `debugFileLocation/record/issueID/referenceID_timestamp.zip`.

For example, compressed output for a recording event for issue ID `103572` and reference ID `policyEvalFails` might be stored at the following path: `*debugFileLocation*/record/103572/policyEvalFails_2015-08-12-12-19-02-683-PDT.zip`.

Use the `referenceID` property value to segregate output from multiple problem recreations associated with the same case. For example, while troubleshooting case 103572, you notice that you only have a problem when evaluating policy for members of the Finance realm. You could trigger two recording events as follows:

Segregating Recording Output Using the referenceID Value

AM Behavior	`referenceID`Value	Recording Output Path
Policy evaluation behaves as expected for members of the Engineering realm.	`policyEvalSucceeds`	`*debugFileLocation*/record/103572/policyEvalSucceeds`
Policy evaluation unexpectedly fails for members of the Finance realm.	`policyEvalFails`	`*debugFileLocation*/record/103572/policyEvalFails`

Chapter 7

Reference

This reference section covers other information relating to maintaining an AM instance. For the global services reference, see Reference.

- "Monitoring"
- "Monitoring Metric Types"
- "Monitoring Metrics"
- "SNMP CTS Object Identifiers"

Monitoring

amster service name: `Monitoring`

Configuration

The following settings appear on the **Configuration** tab:

Monitoring Status

Enable / Disable the monitoring system

Default value: `false`

amster attribute: `enabled`

Monitoring HTTP Port

Port number for the HTTP monitoring interface

Default value: `8082`

amster attribute: `httpPort`

Monitoring HTTP interface status

Enable / Disable the HTTP access to the monitoring system

Default value: `false`

amster attribute: `httpEnabled`

Monitoring HTTP interface authentication file path

Path to the monitoring system authentication file

The `openam_mon_auth` file contains the username and password of the account used to protect the monitoring interfaces. The default username is `demo` with a password of `changeit`. Use the `ampassword` command to encrypt a new password.

Default value: `%BASE_DIR%/security/openam_mon_auth`

amster attribute: `authfilePath`

Monitoring RMI Port

Port number for the JMX monitoring interface

Default value: `9999`

amster attribute: `rmiPort`

Monitoring RMI interface status

Enable / Disable the JMX access to the monitoring system

Default value: `false`

amster attribute: `rmiEnabled`

Monitoring SNMP Port

Port number for the SNMP monitoring interface

Default value: `8085`

amster attribute: `snmpPort`

Monitoring SNMP interface status

Enable / Disable the SNMP access to the monitoring system

Default value: `false`

amster attribute: `snmpEnabled`

Policy evaluation monitoring history size

Size of the window of most recent policy evaluations to record to expose via monitoring system. Valid range is 100 - 1000000.

Default value: `10000`

amster attribute: `policyHistoryWindowSize`

Session monitoring history size

Size of the window of most recent session operations to record to expose via monitoring system. Valid range is 100 - 1000000.

Default value: `10000`

amster attribute: `sessionHistoryWindowSize`

Secondary Configurations

This service has the following Secondary Configurations.

crest

Enabled

Default value: `false`

amster attribute: `enabled`

graphite

Hostname

The hostname of the Graphite server to which metrics should be published.

amster attribute: `host`

Port

The port of the Graphite server to which metrics should be published.

Default value: `2004`

amster attribute: `port`

Frequency

The frequency (in seconds) at which metrics should be published.

Default value: `30`

amster attribute: `frequency`

prometheus

Enabled

Default value: `false`

amster attribute: `enabled`

Authentication Type

Default value: `BASIC`

amster attribute: `authenticationType`

Username

Default value: `prometheus`

amster attribute: `username`

Password

amster attribute: `password`

Monitoring Metric Types

This section describes the monitoring metric types that are available in AM. The available types are:

- Summary
- Timer
- Gauge
- DistinctCounter

Summary

Metric that samples observations, providing a count of observations, sum total of observed amounts, average rate of events, and moving average rates across sliding time windows.

- Fields

  When using the Common REST, JMX, or Graphite interfaces, the `Summary` metric type has the following fields:

Field	Description
`_id`	The metric ID.
`_type`	The metric type.

Field	Description
`count`	The number of events recorded for this metric.
`total`	The sum of the values of events recorded for this metric. **Note** As the increment is always 1, the total and the count will always be equal.
`m1_rate`	The one-minute average rate.
`m5_rate`	The five-minute average rate.
`m15_rate`	The fifteen-minute average rate.
`mean_rate`	The average rate.
`units`	A description of the units the metric is presented in.

The following is an example of the `authentication.success` metric from the Common REST endpoint:

```
{
  "_id" : "authentication.success",
  "_type" : "summary",
  "count" : 2,
  "total" : 2.0,
  "m1_rate" : 3.2668341885586836E-14,
  "m5_rate" : 7.794695663154025E-5,
  "m15_rate" : 0.01377545747021923,
  "mean_rate" : 8.238608027596704E-4,
  "units" : "events/second"
}
```

- Prometheus Fields

  The Prometheus endpoint does not provide rate-based statistics, as rates can be calculated from the time-series data.

  When using the Prometheus interface, the `Summary` metric type has the following fields:

Field	Description
`# TYPE`	The metric ID, and type. Formatted as a comment.
`_count`	The number of events recorded.
`_total`	The sum of the amounts of events recorded

The following is an example of the `am_authentication{outcome="success"}` metric from the Prometheus endpoint:

```
# TYPE am_authentication summary
am_authentication_count{outcome="success"} 2.0
am_authentication_total{outcome="success"} 2.0
```

Timer

Metric that combines both rate and duration information.

- Fields

  When using the Common REST, JMX, or Graphite interfaces, the `Timer` metric type has the following fields:

Field	Description
`_id`	The metric ID.
`_type`	The metric type.
`count`	The number of events recorded for this metric.
`total`	The sum of the durations recorded for this metric.
`min`	The minimum duration recorded for this metric.
`max`	The maximum duration recorded for this metric.
`mean`	The mean average duration recorded for this metric.
`stddev`	The standard deviation of durations recorded for this metric.
`duration_units`	The units used for measuring the durations in the metric.
`p50`	50% of the durations recorded are at or below this value.
`p75`	75% of the durations recorded are at or below this value.
`p95`	95% of the durations recorded are at or below this value.
`p98`	98% of the durations recorded are at or below this value.
`p99`	99% of the durations recorded are at or below this value.
`p999`	99.9% of the durations recorded are at or below this value.
`m1_rate`	The one-minute average rate.
`m5_rate`	The five-minute average rate.
`m15_rate`	The fifteen-minute average rate.
`mean_rate`	The average rate.
`rate_units`	The units used for measuring the rate of the metric.

Note

Duration-based values, such as `min`, `max`, and `p50`, are weighted towards newer data. By representing approximately the last five minutes of data, the timers make it easier to see recent changes in behavior, rather than a uniform average of recordings since the server was started.

The following is an example of the `cts.connection.success` metric from the Common REST endpoint:

```
{
  "_id" : "cts.connection.success",
  "_type" : "timer",
  "count" : 486,
  "total" : 80.0,
  "min" : 0.0,
  "max" : 1.0,
  "mean" : 0.1905615495053855,
  "stddev" : 0.39274399467782056,
  "duration_units" : "milliseconds",
  "p50" : 0.0,
  "p75" : 0.0,
  "p95" : 1.0,
  "p98" : 1.0,
  "p99" : 1.0,
  "p999" : 1.0,
  "m1_rate" : 0.1819109974890356,
  "m5_rate" : 0.05433445522996721,
  "m15_rate" : 0.03155662103953588,
  "mean_rate" : 0.020858521722211427,
  "rate_units" : "calls/second"
}
```

- Prometheus Fields

  The Prometheus endpoint does not provide rate-based statistics, as rates can be calculated from the time-series data.

  When using the Prometheus interface, the `Timer` metric type has the following fields:

Field	Description
`# TYPE`	The metric ID, and type. Note that the `Timer` metric type is reported as a `Summary` type. Formatted as a comment.
`_count`	The number of events recorded.
`_total`	The sum of the durations recorded.
`{quantile="0.5"}`	50% of the durations are at or below this value.
`{quantile="0.75"}`	75% of the durations are at or below this value.
`{quantile="0.95"}`	95% of the durations are at or below this value.
`{quantile="0.98"}`	98% of the durations are at or below this value.
`{quantile="0.99"}`	99% of the durations are at or below this value.
`{quantile="0.999"}`	99.9% of the durations are at or below this value.

Note

Duration-based quantile values are weighted towards newer data. By representing approximately the last five minutes of data, the timers make it easier to see recent changes in behavior, rather than a uniform average of recordings since the server was started.

The following is an example of the `am_authentication{outcome="success"}` metric from the Prometheus endpoint:

```
# TYPE am_cts_connection_seconds summary
am_cts_connection_seconds{outcome="success",quantile="0.5",} 0.0
am_cts_connection_seconds{outcome="success",quantile="0.75",} 0.0
am_cts_connection_seconds{outcome="success",quantile="0.95",} 0.001
am_cts_connection_seconds{outcome="success",quantile="0.98",} 0.001
am_cts_connection_seconds{outcome="success",quantile="0.99",} 0.001
am_cts_connection_seconds{outcome="success",quantile="0.999",} 0.001
am_cts_connection_count{outcome="success",} 492.0
am_cts_connection_seconds_total{outcome="success",} 0.081
```

Gauge

Metric for a numerical value that can increase or decrease. The value for a gauge is calculated when requested, and represents the state of the metric at that specific time.

- Fields

  When using the Common REST, JMX, or Graphite interfaces, the `Timer` metric type has the following fields:

Field	Description
`_id`	The metric ID.
`_type`	The metric type.
`value`	The current value of the metric.

The following is an example of the `jvm.used-memory` metric from the Common REST endpoint:

```
{
  "_id" : "jvm.used-memory",
  "_type" : "gauge",
  "value" : 2.13385216E9
}
```

- Prometheus Fields

  When using the Prometheus interface, the `Timer` metric type has the following fields:

Field	Description
`# TYPE`	The metric ID, and type. Formatted as a comment.
`{Metric ID}`	The current value. Large values may be represented in scientific E-notation.

The following is an example of the `am_jvm_used_memory_bytes` metric from the Prometheus endpoint:

```
# TYPE am_jvm_used_memory_bytes gauge
am_jvm_used_memory_bytes 2.13385216E9
```

DistinctCounter

Metric providing an estimate of the number of *unique* values recorded.

For example, this could be used to estimate the number of unique users who have authenticated, or unique client IP addresses.

> **Note**
>
> The `DistinctCounter` metric is calculated per instance of AM, and cannot be aggregated across multiple instances to get a site-wide view.

- Fields

  When using the Common REST, JMX, or Graphite interfaces, the `DistinctCounter` metric type has the following fields:

Field	Description
`_id`	The metric ID.
`_type`	The metric type. Note that the `distinctCounter` type is reported as a `gauge` type. The output formats are identical.
`value`	The calculated estimate of the number of unique values recorded in the metric.

  The following is an example of the `authentication.unique-uuid.success` metric from the Common REST endpoint:

```
{
  "_id" : "authentication.unique-uuid.success",
  "_type" : "gauge",
  "value" : 3.0
}
```

- Prometheus Fields

  When using the Prometheus interface, the `distinctCounter` metric type has the following fields:

Field	Description
`# TYPE`	The metric ID, and type. Note that the `distinctCounter` type is reported as a `gauge` type. The output formats are identical. Formatted as a comment.
`{Metric ID}`	The calculated estimate of the number of unique values recorded in the metric.

The following is an example of the `am_authentication_unique_uuid{outcome="success"}` metric from the Prometheus endpoint:

```
# TYPE am_authentication_unique_uuid gauge
am_authentication_unique_uuid{outcome="success"} 3.0
```

Monitoring Metrics

AM exposes the monitoring metrics described in this section.

Authentication Metrics

AM exposes the following authentication-related monitoring metrics:

`authentication.module.<auth-module-name>.<outcome>`

Rate of successful/unsuccessful authentication module outcomes. (Summary)

Prometheus syntax:

```
am_authentication_module{module=<auth-module-name>,outcome=<outcome>}
```

Labels:

`<auth-module-name>`

Classname of the authentication module, for example:

`Application`

`DataStore`

`<outcome>`

`success`

`failure`

`timeout`

`authentication.unique-uuid.success`

Count of unique identities which have successfully logged in. (DistinctCounter)

Prometheus syntax:

```
am_authentication_unique_uuid{outcome=success}
```

authentication.*<outcome>*

Rate of successful/unsuccessful/timed-out authentication flows. (Summary)

Prometheus syntax:

```
am_authentication{outcome=<outcome>}
```

Labels:

<outcome>

`success`

`failure`

`timeout`

Authorization Metrics

AM exposes the following authorization-related monitoring metrics:

authorization.policy-set.*<policy-set-name>*.evaluate.action.*<policy-action-name>*.*<outcome>*

Rate of policy evaluation allowed/denied actions being returned under a given policy set. (Summary)

Prometheus syntax:

```
am_authorization_policy_set_evaluate_action{policy_set=<policy-set-name>, action-type=<policy-action-name>,outcome=<outcome>}
```

Labels:

<policy-set-name>

Name of the policy set, for example:

`iPlanetAMWebAgentService`

`oauth2Scopes`

<policy-action-name>

Name of the action as specified in the policy, for example:

`GET`

`POST`

`GRANT`

<outcome>

`allow`

`deny`

`authorization.policy-set.<policy-set-name>.evaluate.advice.<policy-advice-type-name>`

Rate of policy evaluation advice types being returned under a given policy set. (Summary)

Prometheus syntax:

```
am_authorization_policy_set_evaluate_advice{policy_set=<policy-set-name>,advice-type=<policy-advice-
type-name>}
```

Labels:

<policy-set-name>

Name of the policy set, for example:

`iPlanetAMWebAgentService`

`oauth2Scopes`

<policy-advice-type-name>

Name of the policy condition advice, for example:

`AuthSchemeConditionAdvice`

`AuthenticateToServiceConditionAdvice`

`AuthLevelConditionAdvice`

`AuthenticateToTreeConditionAdvice`

`AuthenticateToRealmConditionAdvice`

`TransactionConditionAdvice`

`authorization.policy-set.evaluate.subject-cache.size`

Number of cached subject membership relationships. (Gauge)

Prometheus syntax:

```
am_authorization_policy_set_evaluate_subject_cache_size
```

`authorization.policy-set.<policy-set-name>.evaluate.<outcome>`

Rate of successful/unsuccessful policy evaluation calls under a given policy set and time taken to perform this operation. (Timer)

Prometheus syntax:

```
am_authorization_policy_set_evaluate{policy_set=<policy-set-name>,outcome=<outcome>}
```

Labels:

`<policy-set-name>`

Name of the policy set, for example:

`iPlanetAMWebAgentService`

`oauth2Scopes`

`<outcome>`

`success`

`failure`

`timeout`

`authorization.policy-set.<policy-set-name>.policy.<operation>`

Number of policies created/updated/deleted under a given policy set since this AM instance was started. (Summary)

Prometheus syntax:

```
am_authorization_policy_set_policy{policy_set=<policy-set-name>,operation=<operation>}
```

Labels:

`<policy-set-name>`

Name of the policy set, for example:

`iPlanetAMWebAgentService`

`oauth2Scopes`

`<operation>`

`create`

`update`

`delete`

Blacklisting Metrics

AM exposes the following blacklisting-related monitoring metrics:

`<blacklist-type>.blacklist.bloomfilter.check.<outcome>`

Rate of bloom filter blacklist checks. (Summary)

Prometheus syntax:

```
am_blacklist_bloomfilter_check{blacklist_type=<blacklist-type>,outcome=<outcome>}
```

Labels:

`<blacklist-type>`

`session.client-based` (Prometheus: `session_client_based`)

`oauth2`

`<outcome>`

`negative`. The bloom filter reports that the checked token is not blacklisted.

`false-positive`. The bloom filter reports that the checked token may be blacklisted, but the token was not blacklisted.

`positive`. The bloom filter reports that the checked token may be blacklisted, and this was found to be true.

`<blacklist-type>.blacklist.cache.hit`

Rate of cache hits of the blacklist cache layer. (Summary)

Prometheus syntax:

```
am_blacklist_cache{blacklist_type=<blacklist-type>,outcome=hit}
```

Labels:

`<blacklist-type>`

`session.client-based` (Prometheus: `session_client_based`)

`oauth2`

`<blacklist-type>.blacklist.cache.miss`

Rate of cache misses of the blacklist cache layer. (Summary)

Prometheus syntax:

```
am_blacklist_cache{blacklist_type=<blacklist-type>,outcome=miss}
```

Labels:

`<blacklist-type>`

`session.client-based` (Prometheus: `session_client_based`)

`oauth2`

`<blacklist-type>.blacklist.check.<outcome>`

Rate of blacklist checks. (Summary)

Prometheus syntax:

```
am_blacklist_check{blacklist_type=<blacklist-type>,outcome=<outcome>}
```

Labels:

`<blacklist-type>`

`session.client-based` (Prometheus: `session_client_based`)

`oauth2`

`<outcome>`

`true`. The token is blacklisted.

`false`. The token is not blacklisted.

`<blacklist-type>.blacklist.cts.search.result`

Rate of blacklist entries returned by searches. (Summary)

Prometheus syntax:

```
am_blacklist_cts_search_result{blacklist_type=<blacklist-type>}
```

Labels:

`<blacklist-type>`

`session.client-based` (Prometheus: `session_client_based`)

`oauth2`

`<blacklist-type>.blacklist.cts.search.<outcome>`

Tracks time to search CTS for blacklist entries. (Timer)

Prometheus syntax:

```
am_blacklist_cts_search{blacklist_type=<blacklist-type>,outcome=<outcome>}
```

Labels:

`<blacklist-type>`

`session.client-based` (Prometheus: `session_client_based`)

`oauth2`

`<outcome>`

`success`

`failure`

CTS Metrics

AM exposes the following CTS-related monitoring metrics:

`cts.connection.<outcome>`

Rate of successful/unsuccessful CTS connections to DS and time taken to obtain the connection. (Timer)

Prometheus syntax:

```
am_cts_connection{outcome=<outcome>}
```

Labels:

`<outcome>`

`success`

`failure`

`cts.reaper.cache.size`

Number of entries in the token reaper cache. (Gauge)

Prometheus syntax:

```
am_cts_reaper_cache_size
```

`cts.reaper.cache.<token-type>.deletion.<outcome>`

Rate of successful/unsuccessful token deletions from cache by token type. (Summary)

Prometheus syntax:

```
am_cts_reaper_deletion{reaper_type=cache,token_type=<token-type>,outcome=<outcome>}
```

Labels:

`<token-type>`

`session`

`saml2`

`oauth2`

`rest`

`oauth2-csrf-protection` (Prometheus: `oauth2_csrf_protection`)

`resource-set` (Prometheus: `resource_set`)

`uma-permission-ticket` (Prometheus: `uma_permission_ticket`)

`uma-requesting-party` (Prometheus: `uma_requesting_party`)

`uma-audit-entry` (Prometheus: `uma_audit_entry`)

`session-blacklist` (Prometheus: `session_blacklist`)

`uma-pending-request` (Prometheus: `uma_pending_request`)

`sts`

`oauth2-blacklist` (Prometheus: `oauth2_blacklist`)

`oauth2-stateless` (Prometheus: `oauth2_stateless`)

`push-notification` (Prometheus: `push_notification`)

`cluster-notification` (Prometheus: `cluster_notification`)

`oauth2-stateless-grant` (Prometheus: `oauth2_stateless_grant`)

`transaction`

`authentication-whitelist` (Prometheus: `authentication_whitelist`)

`oauth2-grant-set` (Prometheus: `oauth2_grant_set`)

`<outcome>`

`success`

`failure`

`cts.reaper.search.<token-type>.deletion.<outcome>`

Rate of successful/unsuccessful token deletions from search by token type. (Summary)

Prometheus syntax:

```
am_cts_reaper_deletion{reaper_type=search,token_type=<token-type>,outcome=<outcome>}
```

Labels:

`<token-type>`

- `session`
- `saml2`
- `oauth2`
- `rest`
- `oauth2-csrf-protection` (Prometheus: `oauth2_csrf_protection`)
- `resource-set` (Prometheus: `resource_set`)
- `uma-permission-ticket` (Prometheus: `uma_permission_ticket`)
- `uma-requesting-party` (Prometheus: `uma_requesting_party`)
- `uma-audit-entry` (Prometheus: `uma_audit_entry`)
- `session-blacklist` (Prometheus: `session_blacklist`)
- `uma-pending-request` (Prometheus: `uma_pending_request`)
- `sts`
- `oauth2-blacklist` (Prometheus: `oauth2_blacklist`)
- `oauth2-stateless` (Prometheus: `oauth2_stateless`)
- `push-notification` (Prometheus: `push_notification`)
- `cluster-notification` (Prometheus: `cluster_notification`)
- `oauth2-stateless-grant` (Prometheus: `oauth2_stateless_grant`)
- `transaction`
- `authentication-whitelist` (Prometheus: `authentication_whitelist`)
- `oauth2-grant-set` (Prometheus: `oauth2_grant_set`)

`<outcome>`

- `success`
- `failure`

`cts.reaper.search.<outcome>`

Rate of successful/unsuccessful search and time taken to perform this operation. (Timer)

Prometheus syntax:

```
am_cts_reaper_search{outcome=<outcome>}
```

Labels:

`<outcome>`

`success`

`failure`

`cts.task.queue`

Queueing times for CTS operations. (Timer)

Prometheus syntax:

```
am_cts_task_queue
```

`cts.task.queue.size`

Number of operations waiting in a CTS queue. (Gauge)

Prometheus syntax:

```
am_cts_task_queue_size
```

`cts_task.<token-type>.<operation-type>.<outcome>`

Rate of successful/unsuccessful CTS operation types, by token type and time taken to perform them. (Timer)

Prometheus syntax:

```
am_cts_task{operation=<operation-type>,token-type=<token-type>,outcome=<outcome>}
```

Labels:

`<token-type>`

`session`

`saml2`

`oauth2`

`rest`

- `oauth2-csrf-protection` (Prometheus: `oauth2_csrf_protection`)
- `resource-set` (Prometheus: `resource_set`)
- `uma-permission-ticket` (Prometheus: `uma_permission_ticket`)
- `uma-requesting-party` (Prometheus: `uma_requesting_party`)
- `uma-audit-entry` (Prometheus: `uma_audit_entry`)
- `session-blacklist` (Prometheus: `session_blacklist`)
- `uma-pending-request` (Prometheus: `uma_pending_request`)
- `sts`
- `oauth2-blacklist` (Prometheus: `oauth2_blacklist`)
- `oauth2-stateless` (Prometheus: `oauth2_stateless`)
- `push-notification` (Prometheus: `push_notification`)
- `cluster-notification` (Prometheus: `cluster_notification`)
- `oauth2-stateless-grant` (Prometheus: `oauth2_stateless_grant`)
- `transaction`
- `authentication-whitelist` (Prometheus: `authentication_whitelist`)
- `oauth2-grant-set` (Prometheus: `oauth2_grant_set`)

`<operation-type>`

- `create`
- `read`
- `update`
- `delete`
- `patch`
- `query`
- `partial-query` (Prometheus: `partial_query`)

`<outcome>`

- `success`
- `failure`

JVM Metrics

AM exposes the JVM-related monitoring metrics covered in this section.

To get the metric name used by Prometheus, prepend `am_` to the names below, and replace period (**.**) and hyphen (**-**) characters with underscore (_) characters.

For example, the `jvm.available-cpus` metric is named `am_jvm_available_cpus` in Prometheus.

> **Note**
>
> These metrics may depend on the JVM version and configuration. In particular, garbage-collector-related metrics depend on the garbage collector that the server uses. The garbage-collector metric names are *unstable*, and can change even in a minor JVM release.

JVM Metrics by Name

Name	Description
`jvm.available-cpus`	Number of processors available to the Java virtual machine. (Gauge)
`jvm.class-loading.loaded`	Number of classes loaded since the Java virtual machine started. (Gauge)
`jvm.class-loading.unloaded`	Number of classes unloaded since the Java virtual machine started. (Gauge)
`jvm.garbage-collector.PS-MarkSweep.count`	Number of collections performed by the "parallel scavenge mark sweep" garbage collection algorithm. (Gauge)
`jvm.garbage-collector.PS-MarkSweep.time`	Approximate accumulated time taken by the "parallel scavenge mark sweep" garbage collection algorithm. (Gauge)
`jvm.garbage-collector.PS-Scavenge.count`	Number of collections performed by the "parallel scavenge" garbage collection algorithm. (Gauge)
`jvm.garbage-collector.PS-Scavenge.time`	Approximate accumulated time taken by the "parallel scavenge" garbage collection algorithm. (Gauge)
`jvm.memory-usage.heap.init`	Amount of heap memory that the Java virtual machine initially requested from the operating system. (Gauge)
`jvm.memory-usage.heap.max`	Maximum amount of heap memory that the Java virtual machine will attempt to use. (Gauge)
`jvm.memory-usage.heap.committed`	Amount of heap memory that is committed for the Java virtual machine to use. (Gauge)
`jvm.memory-usage.heap.used`	Amount of heap memory used by the Java virtual machine. (Gauge)
`jvm.memory-usage.total.init`	Amount of memory that the Java virtual machine initially requested from the operating system. (Gauge)

Name	Description
`jvm.memory-usage.total.max`	Maximum amount of memory that the Java virtual machine will attempt to use. (Gauge)
`jvm.memory-usage.non-heap.init`	Amount of non-heap memory that the Java virtual machine initially requested from the operating system. (Gauge)
`jvm.memory-usage.non-heap.max`	Maximum amount of non-heap memory that the Java virtual machine will attempt to use. (Gauge)
`jvm.memory-usage.non-heap.committed`	Amount of non-heap memory that is committed for the Java virtual machine to use. (Gauge)
`jvm.memory-usage.non-heap.used`	Amount of non-heap memory used by the Java virtual machine. (Gauge)
`jvm.memory-usage.pools.Code-Cache.init`	Amount of "code cache" memory that the Java virtual machine initially requested from the operating system. (Gauge)
`jvm.memory-usage.pools.Code-Cache.max`	Maximum amount of "code cache" memory that the Java virtual machine will attempt to use. (Gauge)
`jvm.memory-usage.pools.Code-Cache.committed`	Amount of "code cache" memory that is committed for the Java virtual machine to use. (Gauge)
`jvm.memory-usage.pools.Code-Cache.used`	Amount of "code cache" memory used by the Java virtual machine. (Gauge)
`jvm.memory-usage.pools.Compressed-Class-Space.init`	Amount of "compressed class space" memory that the Java virtual machine initially requested from the operating system. (Gauge)
`jvm.memory-usage.pools.Compressed-Class-Space.init`	Maximum amount of "compressed class space" memory that the Java virtual machine will attempt to use. (Gauge)
`jvm.memory-usage.pools.Compressed-Class-Space.committed`	Amount of "compressed class space" memory that is committed for the Java virtual machine to use. (Gauge)
`jvm.memory-usage.pools.Compressed-Class-Space.used`	Amount of "compressed class space" memory used by the Java virtual machine. (Gauge)
`jvm.memory-usage.pools.Metaspace.init`	Amount of "metaspace" memory that the Java virtual machine initially requested from the operating system. (Gauge)
`jvm.memory-usage.pools.Metaspace.max`	Maximum amount of "metaspace" memory that the Java virtual machine will attempt to use. (Gauge)
`jvm.memory-usage.pools.Metaspace.committed`	Amount of "metaspace" memory that is committed for the Java virtual machine to use. (Gauge)
`jvm.memory-usage.pools.Metaspace.used`	Amount of "metaspace" memory used by the Java virtual machine. (Gauge)

Name	Description
`jvm.memory-usage.pools.PS-Eden-Space.init`	Amount of "parallel scavenge eden space" memory that the Java virtual machine initially requested from the operating system. (Gauge)
`jvm.memory-usage.pools.PS-Eden-Space.max`	Maximum amount of "parallel scavenge eden space" memory that the Java virtual machine will attempt to use. (Gauge)
`jvm.memory-usage.pools.PS-Eden-Space.committed`	Amount of "parallel scavenge eden space" memory that is committed for the Java virtual machine to use. (Gauge)
`jvm.memory-usage.pools.PS-Eden-Space.used-after-gc`	Amount of "parallel scavenge eden space" memory after the last time garbage collection recycled unused objects in this memory pool. (Gauge)
`jvm.memory-usage.pools.PS-Eden-Space.used`	Amount of "parallel scavenge eden space" memory used by the Java virtual machine. (Gauge)
`jvm.memory-usage.pools.PS-Old-Gen.init`	Amount of "parallel scavenge old generation" memory that the Java virtual machine initially requested from the operating system. (Gauge)
`jvm.memory-usage.pools.PS-Old-Gen.max`	Maximum amount of "parallel scavenge old generation" memory that the Java virtual machine will attempt to use. (Gauge)
`jvm.memory-usage.pools.PS-Old-Gen.committed`	Amount of "parallel scavenge old generation" memory that is committed for the Java virtual machine to use. (Gauge)
`jvm.memory-usage.pools.PS-Old-Gen.used-after-gc`	Amount of "parallel scavenge old generation" memory after the last time garbage collection recycled unused objects in this memory pool. (Gauge)
`jvm.memory-usage.pools.PS-Old-Gen.used`	Amount of "parallel scavenge old generation" memory used by the Java virtual machine. (Gauge)
`jvm.memory-usage.pools.PS-Survivor-Space.init`	Amount of "parallel scavenge survivor space" memory that the Java virtual machine initially requested from the operating system. (Gauge)
`jvm.memory-usage.pools.PS-Survivor-Space.max`	Maximum amount of "parallel scavenge survivor space" memory that the Java virtual machine will attempt to use. (Gauge)
`jvm.memory-usage.pools.PS-Survivor-Space.committed`	Amount of "parallel scavenge survivor space" memory that is committed for the Java virtual machine to use. (Gauge)
`jvm.memory-usage.pools.PS-Survivor-Space.used-after-gc`	Amount of "parallel scavenge survivor space" memory after the last time garbage collection recycled unused objects in this memory pool. (Gauge)
`jvm.memory-usage.pools.PS-Survivor-Space.used`	Amount of "parallel scavenge survivor space" memory used by the Java virtual machine. (Gauge)

Name	Description
`jvm.memory-usage.total.committed`	Amount of memory that is committed for the Java virtual machine to use. (Gauge)
`jvm.memory-usage.total.used`	Amount of memory used by the Java virtual machine. (Gauge)
`jvm.thread-state.blocked.count`	Number of threads in the BLOCKED state. (Gauge)
`jvm.thread-state.count`	Number of live threads including both daemon and non-daemon threads. (Gauge)
`jvm.thread-state.daemon.count`	Number of live daemon threads. (Gauge)
`jvm.thread-state.new.count`	Number of threads in the NEW state. (Gauge)
`jvm.thread-state.runnable.count`	Number of threads in the RUNNABLE state. (Gauge)
`jvm.thread-state.terminated.count`	Number of threads in the TERMINATED state. (Gauge)
`jvm.thread-state.timed_waiting.count`	Number of threads in the TIMED_WAITING state. (Gauge)
`jvm.thread-state.waiting.count`	Number of threads in the WAITING state. (Gauge)

OAuth 2.0 Metrics

AM exposes the following CTS-related monitoring metrics:

`oauth2.grant.<grant-type>`

Rate of OAuth 2.0 grant completion by grant type. (Summary)

Prometheus syntax:

```
am_oauth2_grant{grant_type=<grant-type>}
```

Labels:

`<grant-type>`

`authorization-code` (Prometheus: `authorization_code`)

`client-credentials` (Prometheus: `client_credentials`)

`device-code` (Prometheus: `device_code`)

`implicit`

`refresh`

`resource-owner-password` (Prometheus: `resource_owner_password`) `oauth2.grant.revoke`

Rate of OAuth 2.0 grant revocation. (Summary)

Prometheus syntax:

```
am_oauth2_grant_revoke
```

`oauth2.token.<token-type>.issue`

Rate of OAuth 2.0 token issuance by token type. (Summary)

Prometheus syntax:

```
am_oauth2_token_issue{token_type=<token-type>}
```

Labels:

`<token-type>`

`access-token` (Prometheus: `access_token`)

`authorization-code` (Prometheus: `authorization_code`)

`device-code` (Prometheus: `device_code`)

`id-token`. OpenID Connect ID token. (Prometheus: `id_token`)

`ops`. OpenID Connect Ops token for session management.

`permission-ticket`. User-Managed Access permission ticket. (Prometheus: `permission_ticket`)

`refresh-token` (Prometheus: `refresh_token`)

`oauth2.token.access-token.revoke`

Rate of OAuth 2.0 access token revocation. (Summary)

Prometheus syntax:

```
am_oauth2_token_revoke{token_type=access_token}
```

`oauth2.token.read-as-jwt.<outcome>`

Rate of successfully/unsuccessfully reading OAuth 2.0 JSON Web Tokens (JWT). (Timer)

Prometheus syntax:

```
am_oauth2_token_read_as_jwt{outcome=<outcome>}
```

Labels:

`<outcome>`

`success`

`failure`

Session Metrics

AM exposes the following session-related monitoring metrics:

`session.authentication-in-memory.store.size`

Number of authentication sessions stored in the in-memory authentication session store. (Gauge)

Prometheus syntax:

```
am_session_authentication_in_memory_store_size
```

`session.cts-based.cache.eviction`

Rate of evictions from the session cache. (Summary)

Prometheus syntax:

```
am_session_cts_based_cache_eviction
```

`session.cts-based.cache.size`

Number of sessions in the session cache. (Gauge)

Prometheus syntax:

```
am_session_cts_based_cache_size
```

`session.cts-based.cache.hit`

Rate of cache hits for the session cache. (Summary)

Prometheus syntax:

```
am_session_cts_based_cache{outcome=hit}
```

`session.cts-based.cache.miss`

Rate of cache misses for the session cache. (Summary)

Prometheus syntax:

```
am_session_cts_based_cache{outcome=miss}
```

session.*<session-type>*.lifetime

Rate of session lifetimes. (Timer)

Prometheus syntax:

```
am_session_lifetime{session_type=<session-type>}
```

Labels:

<session-type>

`authentication-in-memory`. In-memory *authentication* sessions used to track authentication progress. (Prometheus: `authentication_in_memory`

`authentication-cts-based`. CTS-based *authentication* sessions used to track authentication progress. (Prometheus: `authentication_cts_based`

`authentication-client-based`. Client-based *authentication* sessions used to track authentication progress. (Prometheus: `authentication_client_based`

`cts-based`. CTS-based sessions issued after successful authentication. (Prometheus: `cts_based`

`client-based`. Client-based sessions, for example in a browser cookie, issued after successful authentication. (Prometheus: `client_based`

session.*<session-type>*.add-listener.*<outcome>*

Rate of successful/unsuccessful p-search listener adds and time taken to perform this operation. (Timer)

Prometheus syntax:

```
am_session{session_type=<session-type>,operation=add-listener, outcome=<outcome>}
```

Labels:

<session-type>

`authentication-in-memory`. In-memory *authentication* sessions used to track authentication progress. (Prometheus: `authentication_in_memory`

`authentication-cts-based`. CTS-based *authentication* sessions used to track authentication progress. (Prometheus: `authentication_cts_based`

`authentication-client-based`. Client-based *authentication* sessions used to track authentication progress. (Prometheus: `authentication_client_based`

`cts-based`. CTS-based sessions issued after successful authentication. (Prometheus: `cts_based`

`client-based`. Client-based sessions, for example in a browser cookie, issued after successful authentication. (Prometheus: `client_based`

`<outcome>`

`success`

`failure`

`session.<session-type>.add-pll-listener.<outcome>`

Rate of successful/unsuccessful PLL listener adds and time taken to perform this operation. (Timer)

Prometheus syntax:

```
am_session{session_type=<session-type>,operation=add-pll-listener, outcome=<outcome>}
```

Labels:

`<session-type>`

`authentication-in-memory`. In-memory *authentication* sessions used to track authentication progress. (Prometheus: `authentication_in_memory`

`authentication-cts-based`. CTS-based *authentication* sessions used to track authentication progress. (Prometheus: `authentication_cts_based`

`authentication-client-based`. Client-based *authentication* sessions used to track authentication progress. (Prometheus: `authentication_client_based`

`cts-based`. CTS-based sessions issued after successful authentication. (Prometheus: `cts_based`

`client-based`. Client-based sessions, for example in a browser cookie, issued after successful authentication. (Prometheus: `client_based`

`<outcome>`

`success`

`failure`

`session.<session-type>.check-exists.<outcome>`

Rate of successful/unsuccessful calls to check if a session exists and time taken to perform this operation. (Timer)

Prometheus syntax:

```
am_session{session_type=<session-type>,operation=check-exists, outcome=<outcome>}
```

Labels:

`<session-type>`

`authentication-in-memory`. In-memory *authentication* sessions used to track authentication progress. (Prometheus: `authentication_in_memory`

`authentication-cts-based`. CTS-based *authentication* sessions used to track authentication progress. (Prometheus: `authentication_cts_based`

`authentication-client-based`. Client-based *authentication* sessions used to track authentication progress. (Prometheus: `authentication_client_based`

`cts-based`. CTS-based sessions issued after successful authentication. (Prometheus: `cts_based`

`client-based`. Client-based sessions, for example in a browser cookie, issued after successful authentication. (Prometheus: `client_based`

`<outcome>`

`success`

`failure`

`session.<session-type>.create.<outcome>`

Rate of successful/unsuccessful session creation and time taken to perform this operation. (Timer)

Prometheus syntax:

```
am_session{session_type=<session-type>,operation=create, outcome=<outcome>}
```

Labels:

`<session-type>`

`authentication-in-memory`. In-memory *authentication* sessions used to track authentication progress. (Prometheus: `authentication_in_memory`

`authentication-cts-based`. CTS-based *authentication* sessions used to track authentication progress. (Prometheus: `authentication_cts_based`

`authentication-client-based`. Client-based *authentication* sessions used to track authentication progress. (Prometheus: `authentication_client_based`

`cts-based`. CTS-based sessions issued after successful authentication. (Prometheus: `cts_based`

`client-based`. Client-based sessions, for example in a browser cookie, issued after successful authentication. (Prometheus: `client_based`

`<outcome>`

`success`

`failure`

`session.<session-type>.destroy.<outcome>`

Rate of successful/unsuccessful session destroy and time taken to perform this operation. (Timer)

Prometheus syntax:

```
am_session{session_type=<session-type>,operation=destroy, outcome=<outcome>}
```

Labels:

`<session-type>`

`authentication-in-memory`. In-memory *authentication* sessions used to track authentication progress. (Prometheus: `authentication_in_memory`

`authentication-cts-based`. CTS-based *authentication* sessions used to track authentication progress. (Prometheus: `authentication_cts_based`

`authentication-client-based`. Client-based *authentication* sessions used to track authentication progress. (Prometheus: `authentication_client_based`

`cts-based`. CTS-based sessions issued after successful authentication. (Prometheus: `cts_based`

`client-based`. Client-based sessions, for example in a browser cookie, issued after successful authentication. (Prometheus: `client_based`

`<outcome>`

`success`

`failure`

`session.<session-type>.get-restricted-token-id.<outcome>`

Rate of successful/unsuccessful restricted token ID dereferencing and time taken to perform this operation. (Timer)

Prometheus syntax:

```
am_session{session_type=<session-type>,operation=get-restricted-token-id, outcome=<outcome>}
```

Labels:

`<session-type>`

`authentication-in-memory`. In-memory *authentication* sessions used to track authentication progress. (Prometheus: `authentication_in_memory`

`authentication-cts-based`. CTS-based *authentication* sessions used to track authentication progress. (Prometheus: `authentication_cts_based`

`authentication-client-based`. Client-based *authentication* sessions used to track authentication progress. (Prometheus: `authentication_client_based`

`cts-based`. CTS-based sessions issued after successful authentication. (Prometheus: `cts_based`

`client-based`. Client-based sessions, for example in a browser cookie, issued after successful authentication. (Prometheus: `client_based`

`<outcome>`

`success`

`failure`

`session.<session-type>.idle-timeout.<outcome>`

Rate of successful/unsuccessful session idle time out and time taken to perform this operation. (Timer)

Prometheus syntax:

```
am_session{session_type=<session-type>,operation=idle-timeout, outcome=<outcome>}
```

Labels:

`<session-type>`

`authentication-in-memory`. In-memory *authentication* sessions used to track authentication progress. (Prometheus: `authentication_in_memory`

`authentication-cts-based`. CTS-based *authentication* sessions used to track authentication progress. (Prometheus: `authentication_cts_based`

`authentication-client-based`. Client-based *authentication* sessions used to track authentication progress. (Prometheus: `authentication_client_based`

`cts-based`. CTS-based sessions issued after successful authentication. (Prometheus: `cts_based`

`client-based`. Client-based sessions, for example in a browser cookie, issued after successful authentication. (Prometheus: `client_based`

`<outcome>`

`success`

`failure`

`session.<session-type>.logout.<outcome>`

Rate of successful/unsuccessful session logout and time taken to perform this operation. (Timer)

Prometheus syntax:

```
am_session{session_type=<session-type>,operation=logout, outcome=<outcome>}
```

Labels:

`<session-type>`

`authentication-in-memory`. In-memory *authentication* sessions used to track authentication progress. (Prometheus: `authentication_in_memory`

`authentication-cts-based`. CTS-based *authentication* sessions used to track authentication progress. (Prometheus: `authentication_cts_based`

`authentication-client-based`. Client-based *authentication* sessions used to track authentication progress. (Prometheus: `authentication_client_based`

`cts-based`. CTS-based sessions issued after successful authentication. (Prometheus: `cts_based`

`client-based`. Client-based sessions, for example in a browser cookie, issued after successful authentication. (Prometheus: `client_based`

`<outcome>`

`success`

`failure`

`session.<session-type>.max-timeout.<outcome>`

Rate of successful/unsuccessful session end of life and time taken to perform this operation. (Timer)

Prometheus syntax:

```
am_session{session_type=<session-type>,operation=max-timeout, outcome=<outcome>}
```

Labels:

`<session-type>`

`authentication-in-memory`. In-memory *authentication* sessions used to track authentication progress. (Prometheus: `authentication_in_memory`

`authentication-cts-based`. CTS-based *authentication* sessions used to track authentication progress. (Prometheus: `authentication_cts_based` `authentication-client-based`. Client-based *authentication* sessions used to track authentication progress. (Prometheus: `authentication_client_based`

`cts-based`. CTS-based sessions issued after successful authentication. (Prometheus: `cts_based`

`client-based`. Client-based sessions, for example in a browser cookie, issued after successful authentication. (Prometheus: `client_based`

`<outcome>`

`success`

`failure`

`session.<session-type>.read-all.<outcome>`

Rate of successful/unsuccessful requests to read all sessions and time taken to perform this operation. (Timer)

Prometheus syntax:

```
am_session{session_type=<session-type>,operation=read-all, outcome=<outcome>}
```

Labels:

`<session-type>`

`authentication-in-memory`. In-memory *authentication* sessions used to track authentication progress. (Prometheus: `authentication_in_memory`

`authentication-cts-based`. CTS-based *authentication* sessions used to track authentication progress. (Prometheus: `authentication_cts_based`

`authentication-client-based`. Client-based *authentication* sessions used to track authentication progress. (Prometheus: `authentication_client_based`

`cts-based`. CTS-based sessions issued after successful authentication. (Prometheus: `cts_based`

`client-based`. Client-based sessions, for example in a browser cookie, issued after successful authentication. (Prometheus: `client_based`

`<outcome>`

`success`

`failure`

`session.<session-type>.read.<outcome>`

Rate of successful/unsuccessful session reads and time taken to perform this operation. (Timer)

Prometheus syntax:

```
am_session{session_type=<session-type>,operation=read, outcome=<outcome>}
```

Labels:

`<session-type>`

`authentication-in-memory`. In-memory *authentication* sessions used to track authentication progress. (Prometheus: `authentication_in_memory`

`authentication-cts-based`. CTS-based *authentication* sessions used to track authentication progress. (Prometheus: `authentication_cts_based`

`authentication-client-based`. Client-based *authentication* sessions used to track authentication progress. (Prometheus: `authentication_client_based`

`cts-based`. CTS-based sessions issued after successful authentication. (Prometheus: `cts_based`

`client-based`. Client-based sessions, for example in a browser cookie, issued after successful authentication. (Prometheus: `client_based`

`<outcome>`

`success`

`failure`

`session.<session-type>.refresh.<outcome>`

Rate of successful/unsuccessful session refresh and time taken to perform this operation. (Timer)

Prometheus syntax:

```
am_session{session_type=<session-type>,operation=refresh, outcome=<outcome>}
```

Labels:

`<session-type>`

`authentication-in-memory`. In-memory *authentication* sessions used to track authentication progress. (Prometheus: `authentication_in_memory`

`authentication-cts-based`. CTS-based *authentication* sessions used to track authentication progress. (Prometheus: `authentication_cts_based`

`authentication-client-based`. Client-based *authentication* sessions used to track authentication progress. (Prometheus: `authentication_client_based`

`cts-based`. CTS-based sessions issued after successful authentication. (Prometheus: `cts_based`

`client-based`. Client-based sessions, for example in a browser cookie, issued after successful authentication. (Prometheus: `client_based`

`<outcome>`

`success`

`failure`

`session.<session-type>.search.<outcome>`

Rate of successful/unsuccessful session searches and time taken to perform this operation. (Timer)

Prometheus syntax:

```
am_session{session_type=<session-type>,operation=search, outcome=<outcome>}
```

Labels:

`<session-type>`

`authentication-in-memory`. In-memory *authentication* sessions used to track authentication progress. (Prometheus: `authentication_in_memory`

`authentication-cts-based`. CTS-based *authentication* sessions used to track authentication progress. (Prometheus: `authentication_cts_based`

`authentication-client-based`. Client-based *authentication* sessions used to track authentication progress. (Prometheus: `authentication_client_based`

`cts-based`. CTS-based sessions issued after successful authentication. (Prometheus: `cts_based`

`client-based`. Client-based sessions, for example in a browser cookie, issued after successful authentication. (Prometheus: `client_based`

`<outcome>`

`success`

`failure`

`session.<session-type>.set-external-property.<outcome>`

Rate of successful/unsuccessful setting a property on a session and time taken to perform this operation. (Timer)

Prometheus syntax:

```
am_session{session_type=<session-type>,operation=set-external-property, outcome=<outcome>}
```

Labels:

<session-type>

`authentication-in-memory`. In-memory *authentication* sessions used to track authentication progress. (Prometheus: `authentication_in_memory`

`authentication-cts-based`. CTS-based *authentication* sessions used to track authentication progress. (Prometheus: `authentication_cts_based`

`authentication-client-based`. Client-based *authentication* sessions used to track authentication progress. (Prometheus: `authentication_client_based`

`cts-based`. CTS-based sessions issued after successful authentication. (Prometheus: `cts_based`

`client-based`. Client-based sessions, for example in a browser cookie, issued after successful authentication. (Prometheus: `client_based`

<outcome>

`success`

`failure`

`session.<session-type>.set-property.<outcome>`

Rate of successful/unsuccessful session property setting and time taken to perform this operation. (Timer)

Prometheus syntax:

```
am_session{session_type=<session-type>,operation=set-property, outcome=<outcome>}
```

Labels:

<session-type>

`authentication-in-memory`. In-memory *authentication* sessions used to track authentication progress. (Prometheus: `authentication_in_memory`

`authentication-cts-based`. CTS-based *authentication* sessions used to track authentication progress. (Prometheus: `authentication_cts_based`

`authentication-client-based`. Client-based *authentication* sessions used to track authentication progress. (Prometheus: `authentication_client_based`

`cts-based`. CTS-based sessions issued after successful authentication. (Prometheus: `cts_based`

`client-based`. Client-based sessions, for example in a browser cookie, issued after successful authentication. (Prometheus: `client_based`

<outcome>

success

failure

session.*<session-type>*.validate.*<outcome>*

Rate of successful/unsuccessful session validation and time taken to perform this operation. (Timer)

Prometheus syntax:

```
am_session{session_type=<session-type>,operation=validate, outcome=<outcome>}
```

Labels:

<session-type>

authentication-in-memory. In-memory *authentication* sessions used to track authentication progress. (Prometheus: authentication_in_memory

authentication-cts-based. CTS-based *authentication* sessions used to track authentication progress. (Prometheus: authentication_cts_based

authentication-client-based. Client-based *authentication* sessions used to track authentication progress. (Prometheus: authentication_client_based

cts-based. CTS-based sessions issued after successful authentication. (Prometheus: cts_based

client-based. Client-based sessions, for example in a browser cookie, issued after successful authentication. (Prometheus: client_based

<outcome>

success

failure

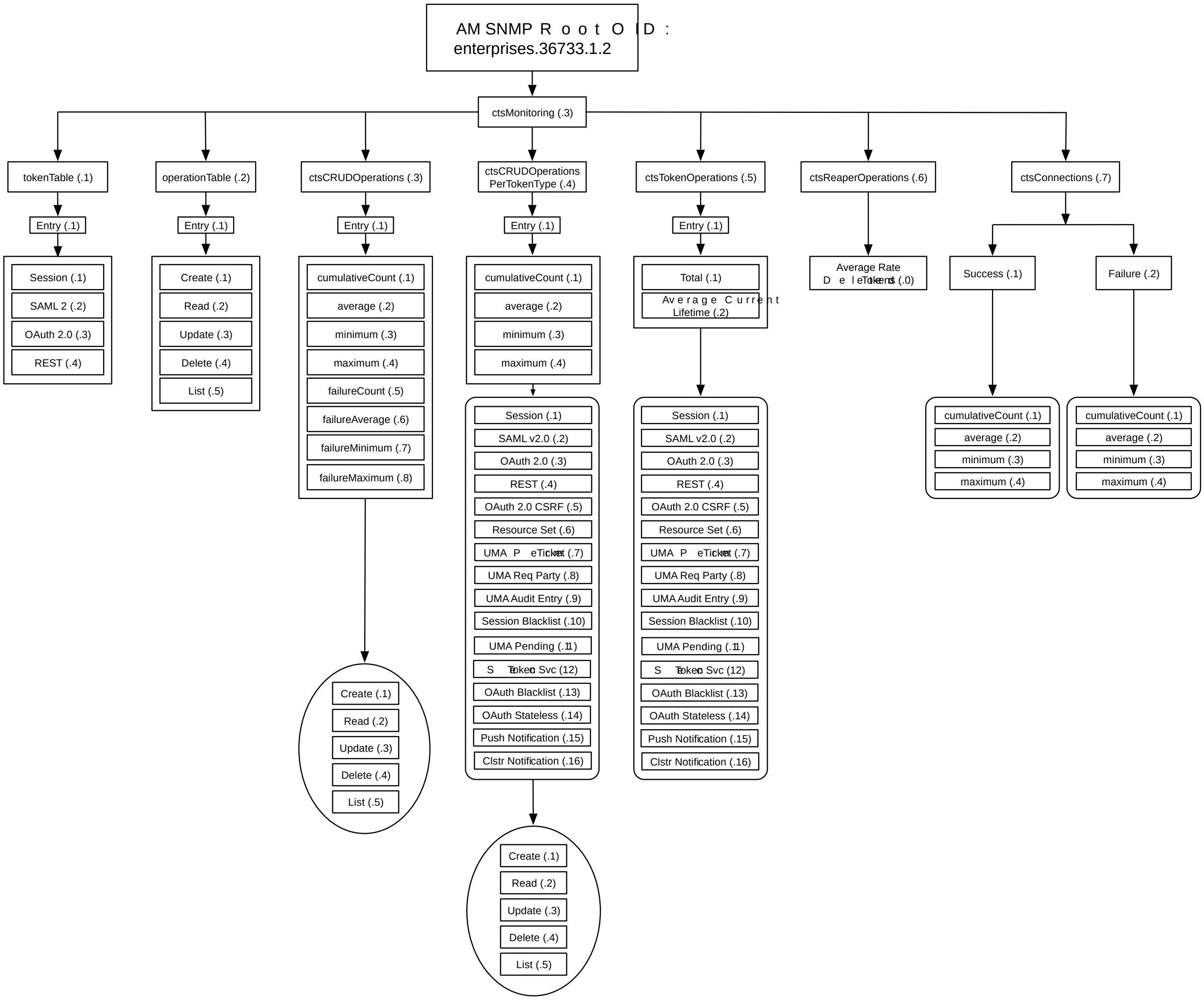
SNMP CTS Object Identifiers

The OIDs related to SNMP monitoring of CTS follow guidance described in RFC 1271.

The OIDs listed in this section include the prefix assigned to ForgeRock, enterprises.36733. They also include the entries associated with AM (1), SNMP (2), and CTS monitoring (3): 1.2.3.

Therefore, the root OID for all CTS monitored components is enterprises.36733.1.2.3. All individual monitored CTS components are suffixes that are consistent with the image shown here.

Diagram of CTS OIDs



CTS Token Type OIDs

The table below shows how OIDs are split into different token types. Do not forget the prefix. For example, the complete OID for monitoring SAML v2.0 tokens is `enterprises.36733.1.2.3.1.1.2`

The options for the token table are shown in the following table. For example, the token table OID for SAML v2.0 is based on the entries associated with ForgeRock, `enterprises.36733`, AM `1`, SNMP `2`, CTS Monitoring `3`, token table `1`, entry `1`, and SAML v2.0 `2`, which is `enterprises.36733.1.2.3.1.1.2`.

CTS Monitoring OID Categories

OID, by Token Type	Description
`enterprises.36733.1.2.3.1.1.1`	Session
`enterprises.36733.1.2.3.1.1.2`	SAML v2.0
`enterprises.36733.1.2.3.1.1.3`	OAuth 2.0
`enterprises.36733.1.2.3.1.1.4`	REST
`enterprises.36733.1.2.3.1.1.5`	OAuth 2.0 CSRF Protection
`enterprises.36733.1.2.3.1.1.6`	UMA Resource
`enterprises.36733.1.2.3.1.1.7`	UMA Permission Ticket
`enterprises.36733.1.2.3.1.1.8`	UMA Requesting Party
`enterprises.36733.1.2.3.1.1.9`	UMA Audit Entry
`enterprises.36733.1.2.3.1.1.10`	Session Blacklist
`enterprises.36733.1.2.3.1.1.11`	UMA Pending Request
`enterprises.36733.1.2.3.1.1.12`	Security Token Service
`enterprises.36733.1.2.3.1.1.13`	OAuth 2.0 Blacklist
`enterprises.36733.1.2.3.1.1.14`	OAuth 2.0 Client-Based
`enterprises.36733.1.2.3.1.1.15`	Push Notification
`enterprises.36733.1.2.3.1.1.16`	Cluster-wide Notification

CTS Monitoring Operation Types

OIDs related to CTS monitoring operations are based on basic CRUD operations (plus list).

The options for the operation table are shown in the following table.

CTS Monitoring Operation Types

OID, by Operation	Description
`enterprises.36733.1.2.3.2.1.1`	Create
`enterprises.36733.1.2.3.2.1.2`	Read
`enterprises.36733.1.2.3.2.1.3`	Update
`enterprises.36733.1.2.3.2.1.4`	Delete
`enterprises.36733.1.2.3.2.1.5`	List

CTS Monitoring Entry Data Types

CTS monitoring entries use the following data types:

`Counter64`

A 64-bit, unsigned integer type.

`Counter64` is a standard data type returned by SNMP OIDs. For more information, see Structure of Management Information Version 2.

`Float2dp`

A floating point number with the value `d-2` in the `DISPLAY-HINT` clause. SNMP clients that handle the `DISPLAY-HINT` clause will correctly display the value as a floating point number with two decimal places. Other types of clients that do not handle the `DISPLAY-HINT` clause will incorrectly display the value as an integer that is one hundred times larger than the correct value.

`Float2dp` is a custom data type returned by some ForgeRock CTS OIDs.

CTS CRUD Operation Entries

The OIDs in this table relate to all CRUD (and list) operations.

The options for the CRUD operations table are shown in the following tables. Each value is associated with CRUD and list operations.

CTS CRUD Operation Entries

OID, by Operation Entry	Data Type	Description
`enterprises.36733.1.2.3.3.1.1`	`Counter64`	Cumulative count
`enterprises.36733.1.2.3.3.1.2`	`Float2dp`	Average (in period)
`enterprises.36733.1.2.3.3.1.3`	`Counter64`	Minimum (in period)
`enterprises.36733.1.2.3.3.1.4`	`Counter64`	Maximum (in period)
`enterprises.36733.1.2.3.3.1.5`	`Counter64`	Cumulative failure count
`enterprises.36733.1.2.3.3.1.6`	`Float2dp`	Average failures (in period)
`enterprises.36733.1.2.3.3.1.7`	`Counter64`	Minimum failures (in period)
`enterprises.36733.1.2.3.3.1.8`	`Counter64`	Maximum failures (in period)

Each of the options in this table can be divided into CRUD and list related operations. The suffix OID for such operations is as follows:

- 1: Create
- 2: Read
- 3: Update

- 4: Delete
- 5: List

For example, since the OID for cumulative count is `enterprises.36733.1.2.3.3.1.1`, the OID for the cumulative count of delete operations is `enterprises.36733.1.2.3.3.1.1.4`

CTS CRUD Operation Table Cumulative Operations

Cumulative Count Operations OID	Data Type	Description
`enterprises.36733.1.2.3.3.1.1.1`	`Counter64`	Cumulative count of CREATE operations
`enterprises.36733.1.2.3.3.1.1.2`	`Counter64`	Cumulative count of READ operations
`enterprises.36733.1.2.3.3.1.1.3`	`Counter64`	Cumulative count of UPDATE operations
`enterprises.36733.1.2.3.3.1.1.4`	`Counter64`	Cumulative count of DELETE operations
`enterprises.36733.1.2.3.3.1.1.5`	`Counter64`	Cumulative count of LIST operations

CTS CRUD Operation Table Average Operations (In Period)

Average Number Operations OID	Data Type	Description
`enterprises.36733.1.2.3.3.1.2.1`	`Float2dp`	Average number of CREATE operations (in period)
`enterprises.36733.1.2.3.3.1.2.2`	`Float2dp`	Average number of READ operations (in period)
`enterprises.36733.1.2.3.3.1.2.3`	`Float2dp`	Average number of UPDATE operations (in period)
`enterprises.36733.1.2.3.3.1.2.4`	`Float2dp`	Average number of DELETE operations (in period)
`enterprises.36733.1.2.3.3.1.2.5`	`Float2dp`	Average number of LIST operations (in period)

CTS CRUD Operation Table Minimum Operations (In Period)

Minimum Number Operations OID	Data Type	Description
`enterprises.36733.1.2.3.3.1.3.1`	`Counter64`	Minimum number of CREATE operations (in period)
`enterprises.36733.1.2.3.3.1.3.2`	`Counter64`	Minimum number of READ operations (in period)
`enterprises.36733.1.2.3.3.1.3.3`	`Counter64`	Minimum number of UPDATE operations (in period)
`enterprises.36733.1.2.3.3.1.3.4`	`Counter64`	Minimum number of DELETE operations (in period)

Minimum Number Operations OID	Data Type	Description
`enterprises.36733.1.2.3.3.1.3.5`	`Counter64`	Minimum number of LIST operations (in period)

CTS CRUD Operation Table Maximum Operations (In Period)

Maximum Number Operations OID	Data Type	Description
`enterprises.36733.1.2.3.3.1.4.1`	`Counter64`	Maximum number of CREATE operations (in period)
`enterprises.36733.1.2.3.3.1.4.2`	`Counter64`	Maximum number of READ operations (in period)
`enterprises.36733.1.2.3.3.1.4.3`	`Counter64`	Maximum number of UPDATE operations (in period)
`enterprises.36733.1.2.3.3.1.4.4`	`Counter64`	Maximum number of DELETE operations (in period)
`enterprises.36733.1.2.3.3.1.4.5`	`Counter64`	Maximum number of LIST operations (in period)

CTS CRUD Operation Table Cumulative Failure Operations

Cumulative Failure Operations OID	Data Type	Description
`enterprises.36733.1.2.3.3.1.5.1`	`Counter64`	Cumulative Failure of CREATE operations (in period)
`enterprises.36733.1.2.3.3.1.5.2`	`Counter64`	Cumulative Failure of READ operations (in period)
`enterprises.36733.1.2.3.3.1.5.3`	`Counter64`	Cumulative Failure of UPDATE operations (in period)
`enterprises.36733.1.2.3.3.1.5.4`	`Counter64`	Cumulative Failure of DELETE operations (in period)
`enterprises.36733.1.2.3.3.1.5.5`	`Counter64`	Cumulative Failure of LIST operations (in period)

CTS CRUD Operation Table Average Failure Operations in Period

Average Number, Failure Operations OID	Data Type	Description
`enterprises.36733.1.2.3.3.1.6.1`	`Float2dp`	Average number of CREATE operations failures (in period)
`enterprises.36733.1.2.3.3.1.6.2`	`Float2dp`	Average number of READ operations failures (in period)
`enterprises.36733.1.2.3.3.1.6.3`	`Float2dp`	Average number of UPDATE operations failures (in period)

Average Number, Failure Operations OID	Data Type	Description
enterprises.36733.1.2.3.3.1.6.4	Float2dp	Average number of DELETE operations failures (in period)
enterprises.36733.1.2.3.3.1.6.5	Float2dp	Average number of LIST operations failures (in period)

CTS CRUD Operation Table Minimum Operations Failures in Period

Minimum Number, Operations Failures OID	Data Type	Description
enterprises.36733.1.2.3.3.1.7.1	Counter64	Minimum number of CREATE operations failures (in period)
enterprises.36733.1.2.3.3.1.7.2	Counter64	Minimum number of READ operations failures (in period)
enterprises.36733.1.2.3.3.1.7.3	Counter64	Minimum number of UPDATE operations failures (in period)
enterprises.36733.1.2.3.3.1.7.4	Counter64	Minimum number of DELETE operations failures (in period)
enterprises.36733.1.2.3.3.1.7.5	Counter64	Minimum number of LIST operations failures (in period)

CTS CRUD Operation Table Maximum Operations Failures in Period

Maximum Number, Operations Failures OID	Data Type	Description
enterprises.36733.1.2.3.3.1.8.1	Counter64	Maximum number of CREATE operations failures (in period)
enterprises.36733.1.2.3.3.1.8.2	Counter64	Maximum number of READ operations failures (in period)
enterprises.36733.1.2.3.3.1.8.3	Counter64	Maximum number of UPDATE operations failures (in period)
enterprises.36733.1.2.3.3.1.8.4	Counter64	Maximum number of DELETE operations failures (in period)
enterprises.36733.1.2.3.3.1.8.5	Counter64	Maximum number of LIST operations failures (in period)

CTS CRUD Operations Per Token Type

OIDs that start with `enterprises.36733.1.2.3.4.1` are labels for CTS CRUD operations per token type.

Tokens of each type can be created, read, updated, deleted, and listed. Each of these types can be measured cumulatively. They can also be measured over a period of time (default=10 seconds), as an average, minimum, and maximum.

OID suffixes for CRUD operations are defined according to the following rules.

The first part of the OID is `enterprises.36733.1.2.3.4.1`.

The next OID suffix specifies a metric:

CTS CRUD Operation Metrics

OID Suffix	Data Type	Metric
1	`Counter64`	Cumulative count
2	`Float2dp`	Average (in period)
3	`Counter64`	Minimum (in period)
4	`Counter64`	Maximum (in period)

The next OID suffix specifies a token type:

CTS CRUD Operation Token Types

OID Suffix	Token Type
1	Session
2	SAML v2.0
3	OAuth 2
4	REST
5	OAuth 2.0 CSRF Protection
6	UMA Resource
7	UMA Permission Ticket
8	UMA Requesting Party
9	UMA Audit Entry
10	Session Blacklist
11	UMA Pending Request
12	Security Token Service
13	OAuth 2.0 Blacklist
14	OAuth 2.0 Client-Based
15	Push Notification
16	Cluster-wide Notification

The final OID suffix specifies an operation:

CTS CRUD Operations

OID Suffix	Operation
1	Create
2	Read
3	Update
4	Delete
5	List

The following examples illustrate OID construction for CTS CRUD operations per token type.

OID Examples for CTS CRUD Operations Per Token Type

OID	Data Type	Description
`enterprises.36733.1.2.3.4.1.1.1.3`	`Counter64`	Cumulative count of updated Session tokens
`enterprises.36733.1.2.3.4.1.4.3.4`	`Counter64`	Maximum deleted OAuth 2.0 tokens (in period)
`enterprises.36733.1.2.3.4.1.2.10.5`	`Float2dp`	Average listed Session Blacklist tokens (in period)

CTS Token Operation Status

The CTS token OIDs defined in this section specify the total number of tokens of each type and their average current lifetimes.

The options for token operations are shown in the following tables. Total and average current lifetimes are associated with each CTS token type.

CTS Total Tokens, by Type

Total Tokens, by Type	Data Type	Description
`enterprises.36733.1.2.3.5.1.1.1`	`Counter64`	Total number of Session tokens
`enterprises.36733.1.2.3.5.1.1.2`	`Counter64`	Total number of SAML v2.0 tokens
`enterprises.36733.1.2.3.5.1.1.3`	`Counter64`	Total number of OAuth 2.0 tokens
`enterprises.36733.1.2.3.5.1.1.4`	`Counter64`	Total number of REST tokens
`enterprises.36733.1.2.3.5.1.1.5`	`Counter64`	Total number of OAuth 2.0 CSRF Protection tokens
`enterprises.36733.1.2.3.5.1.1.6`	`Counter64`	Total number of UMA Resource tokens
`enterprises.36733.1.2.3.5.1.1.7`	`Counter64`	Total number of UMA Permission Ticket tokens

Total Tokens, by Type	Data Type	Description
enterprises.36733.1.2.3.5.1.1.8	Counter64	Total number of UMA Requesting Party tokens
enterprises.36733.1.2.3.5.1.1.9	Counter64	Total number of UMA Audit Entry tokens
enterprises.36733.1.2.3.5.1.1.10	Counter64	Total number of Session Blacklist tokens
enterprises.36733.1.2.3.5.1.1.11	Counter64	Total number of UMA Pending Request tokens
enterprises.36733.1.2.3.5.1.1.12	Counter64	Total number of Security Token Service tokens
enterprises.36733.1.2.3.5.1.1.13	Counter64	Total number of OAuth 2.0 Blacklist tokens
enterprises.36733.1.2.3.5.1.1.14	Counter64	Total number of OAuth 2.0 client-based tokens
enterprises.36733.1.2.3.5.1.1.15	Counter64	Total number of Push Notification tokens
enterprises.36733.1.2.3.5.1.1.16	Counter64	Total number of Cluster-wide Notification tokens

CTS Token Average Lifetime, by Type

Average Token Lifetime, by Type	Data Type	Description
enterprises.36733.1.2.3.5.1.2.1	Counter64	Average lifetime of Session tokens in seconds
enterprises.36733.1.2.3.5.1.2.2	Counter64	Average lifetime of SAML v2.0 tokens in seconds
enterprises.36733.1.2.3.5.1.2.3	Counter64	Average lifetime of OAuth 2.0 tokens in seconds
enterprises.36733.1.2.3.5.1.2.4	Counter64	Average lifetime of REST tokens in seconds
enterprises.36733.1.2.3.5.1.2.5	Counter64	Average lifetime of OAuth 2.0 CSRF Protection tokens in seconds
enterprises.36733.1.2.3.5.1.2.6	Counter64	Average lifetime of UMA Resource tokens in seconds
enterprises.36733.1.2.3.5.1.2.7	Counter64	Average lifetime of UMA Permission Ticket tokens in seconds
enterprises.36733.1.2.3.5.1.2.8	Counter64	Average lifetime of UMA Requesting Party tokens in seconds
enterprises.36733.1.2.3.5.1.2.9	Counter64	Average lifetime of UMA Audit Entry tokens in seconds
enterprises.36733.1.2.3.5.1.2.10	Counter64	Average lifetime of Session Blacklist tokens in seconds
enterprises.36733.1.2.3.5.1.2.11	Counter64	Average lifetime of UMA Pending Request tokens in seconds
enterprises.36733.1.2.3.5.1.2.12	Counter64	Average lifetime of Security Token Service tokens in seconds

Average Token Lifetime, by Type	Data Type	Description
enterprises.36733.1.2.3.5.1.2.13	Counter64	Average lifetime of OAuth 2.0 Blacklist tokens in seconds
enterprises.36733.1.2.3.5.1.2.14	Counter64	Average lifetime of OAuth 2.0 client-based tokens in seconds
enterprises.36733.1.2.3.5.1.2.15	Counter64	Average lifetime of Push Notification tokens in seconds
enterprises.36733.1.2.3.5.1.2.16	Counter64	Average lifetime of Cluster-wide Notification tokens in seconds

CTS Reaper Run Information

The CTS reaper deletes unused or expired tokens. Unless AM is in a shutdown cycle, the CTS reaper is designed to run continuously. By default, the CTS reaper runs in fixed intervals, unless AM is in the process of shutting down.

A single OID, `enterprises.36733.1.2.3.6.0`, relates to the CTS reaper. This OID:

- Specifies the average rate of deleted tokens per CTS reaper run
- Has the `Float2dp`data type.

CTS Connection Factory OIDs

Every request for a CTS token is a request to the `CTSConnectionFactory`. Such requests can either succeed or fail. The following OIDs provide measures for both such connections. The CTSConnectionFactory OIDs are also measured using a rate window system, similar to all the other CTS OIDs, except the CTS Reaper.

As there are no indexes required to look up the value of CTSConnectionFactory OIDs, they end in 0. Success or failure of these OIDs are not specific to any operation or token type.

The following tables list the OIDs related to the CTSConnectionFactory.

CTSConnectionFactory, Successful Connections

Successes, CTSConnectionFactory	Data Type	Description
enterprises.36733.1.2.3.7.1.1.0	Counter64	Cumulative number of successful connections
enterprises.36733.1.2.3.7.1.2.0	Float2dp	Average number of successful connections (in period)
enterprises.36733.1.2.3.7.1.3.0	Counter64	Minimum number of successful connections (in period)
enterprises.36733.1.2.3.7.1.4.0	Counter64	Maximum number of successful connections (in period)

CTSConnectionFactory, Failed Connections

Failures, CTSConnectionFactory	Data Type	Description
`enterprises.36733.1.2.3.7.2.1.0`	`Counter64`	Cumulative number of failed connections
`enterprises.36733.1.2.3.7.2.2.0`	`Float2dp`	Average number of failed connections (in period)
`enterprises.36733.1.2.3.7.2.3.0`	`Counter64`	Minimum number of failed connections (in period)
`enterprises.36733.1.2.3.7.2.4.0`	`Counter64`	Maximum number of failed connections (in period)

Glossary

Access control

Control to grant or to deny access to a resource.

Account lockout

The act of making an account temporarily or permanently inactive after successive authentication failures.

Actions

Defined as part of policies, these verbs indicate what authorized identities can do to resources.

Advice

In the context of a policy decision denying access, a hint to the policy enforcement point about remedial action to take that could result in a decision allowing access.

Agent administrator

User having privileges only to read and write agent profile configuration information, typically created to delegate agent profile creation to the user installing a web or Java agent.

Agent authenticator

Entity with read-only access to multiple agent profiles defined in the same realm; allows an agent to read web service profiles.

Application

In general terms, a service exposing protected resources.

In the context of AM policies, the application is a template that constrains the policies that govern access to protected resources. An application can have zero or more policies.

Application type

Application types act as templates for creating policy applications.

Application types define a preset list of actions and functional logic, such as policy lookup and resource comparator logic.

Application types also define the internal normalization, indexing logic, and comparator logic for applications.

Attribute-based access control (ABAC)

Access control that is based on attributes of a user, such as how old a user is or whether the user is a paying customer.

Authentication

The act of confirming the identity of a principal.

Authentication chaining

A series of authentication modules configured together which a principal must negotiate as configured in order to authenticate successfully.

Authentication level

Positive integer associated with an authentication module, usually used to require success with more stringent authentication measures when requesting resources requiring special protection.

Authentication module

AM authentication unit that handles one way of obtaining and verifying credentials.

Authorization

The act of determining whether to grant or to deny a principal access to a resource.

Authorization Server

In OAuth 2.0, issues access tokens to the client after authenticating a resource owner and confirming that the owner authorizes the client to access the protected resource. AM can play this role in the OAuth 2.0 authorization framework.

Auto-federation

Arrangement to federate a principal's identity automatically based on a common attribute value shared across the principal's profiles at different providers.

Bulk federation

Batch job permanently federating user profiles between a service provider and an identity provider based on a list of matched user identifiers that exist on both providers.

Circle of trust

Group of providers, including at least one identity provider, who have agreed to trust each other to participate in a SAML v2.0 provider federation.

Client

In OAuth 2.0, requests protected web resources on behalf of the resource owner given the owner's authorization. AM can play this role in the OAuth 2.0 authorization framework.

Client-based OAuth 2.0 tokens

After a successful OAuth 2.0 grant flow, AM returns a token to the client. This differs from CTS-based OAuth 2.0 tokens, where AM returns a *reference* to token to the client.

Client-based sessions

AM sessions for which AM returns session state to the client after each request, and require it to be passed in with the subsequent request. For browser-based clients, AM sets a cookie in the browser that contains the session information.

For browser-based clients, AM sets a cookie in the browser that contains the session state. When the browser transmits the cookie back to AM, AM decodes the session state from the cookie.

Conditions

Defined as part of policies, these determine the circumstances under which which a policy applies.

Environmental conditions reflect circumstances like the client IP address, time of day, how the subject authenticated, or the authentication level achieved.

Subject conditions reflect characteristics of the subject like whether the subject authenticated, the identity of the subject, or claims in the subject's JWT.

Configuration datastore

LDAP directory service holding AM configuration data.

Cross-domain single sign-on (CDSSO)

AM capability allowing single sign-on across different DNS domains.

CTS-based OAuth 2.0 tokens

After a successful OAuth 2.0 grant flow, AM returns a *reference* to the token to the client, rather than the token itself. This differs from client-based OAuth 2.0 tokens, where AM returns the entire token to the client.

CTS-based sessions

AM sessions that reside in the Core Token Service's token store. CTS-based sessions might also be cached in memory on one or more AM servers. AM tracks these sessions in order to handle events like logout and timeout, to permit session constraints, and to notify applications involved in SSO when a session ends.

Delegation

Granting users administrative privileges with AM.

Entitlement

Decision that defines which resource names can and cannot be accessed for a given identity in the context of a particular application, which actions are allowed and which are denied, and any related advice and attributes.

Extended metadata

Federation configuration information specific to AM.

Extensible Access Control Markup Language (XACML)

Standard, XML-based access control policy language, including a processing model for making authorization decisions based on policies.

Federation

Standardized means for aggregating identities, sharing authentication and authorization data information between trusted providers, and allowing principals to access services across different providers without authenticating repeatedly.

Fedlet
Service provider application capable of participating in a circle of trust and allowing federation without installing all of AM on the service provider side; AM lets you create Java Fedlets.

Hot swappable
Refers to configuration properties for which changes can take effect without restarting the container where AM runs.

Identity
Set of data that uniquely describes a person or a thing such as a device or an application.

Identity federation
Linking of a principal's identity across multiple providers.

Identity provider (IDP)
Entity that produces assertions about a principal (such as how and when a principal authenticated, or that the principal's profile has a specified attribute value).

Identity repository
Data store holding user profiles and group information; different identity repositories can be defined for different realms.

Java agent
Java web application installed in a web container that acts as a policy enforcement point, filtering requests to other applications in the container with policies based on application resource URLs.

Metadata
Federation configuration information for a provider.

Policy
Set of rules that define who is granted access to a protected resource when, how, and under what conditions.

Policy agent
Java, web, or custom agent that intercepts requests for resources, directs principals to AM for authentication, and enforces policy decisions from AM.

Policy Administration Point (PAP)
Entity that manages and stores policy definitions.

Policy Decision Point (PDP)
Entity that evaluates access rights and then issues authorization decisions.

Policy Enforcement Point (PEP)
Entity that intercepts a request for a resource and then enforces policy decisions from a PDP.

Policy Information Point (PIP)
Entity that provides extra information, such as user profile attributes that a PDP needs in order to make a decision.

Principal
Represents an entity that has been authenticated (such as a user, a device, or an application), and thus is distinguished from other entities.

When a Subject successfully authenticates, AM associates the Subject with the Principal.

Privilege

In the context of delegated administration, a set of administrative tasks that can be performed by specified identities in a given realm.

Provider federation

Agreement among providers to participate in a circle of trust.

Realm

AM unit for organizing configuration and identity information.

Realms can be used for example when different parts of an organization have different applications and identity stores, and when different organizations use the same AM deployment.

Administrators can delegate realm administration. The administrator assigns administrative privileges to users, allowing them to perform administrative tasks within the realm.

Resource

Something a user can access over the network such as a web page.

Defined as part of policies, these can include wildcards in order to match multiple actual resources.

Resource owner

In OAuth 2.0, entity who can authorize access to protected web resources, such as an end user.

Resource server

In OAuth 2.0, server hosting protected web resources, capable of handling access tokens to respond to requests for such resources.

Response attributes

Defined as part of policies, these allow AM to return additional information in the form of "attributes" with the response to a policy decision.

Role based access control (RBAC)

Access control that is based on whether a user has been granted a set of permissions (a role).

Security Assertion Markup Language (SAML)

Standard, XML-based language for exchanging authentication and authorization data between identity providers and service providers.

Service provider (SP)

Entity that consumes assertions about a principal (and provides a service that the principal is trying to access).

Authentication Session

The interval while the user or entity is authenticating to AM.

Session

The interval that starts after the user has authenticated and ends when the user logs out, or when their session is terminated. For browser-based clients, AM manages user sessions across one or more applications by setting a session cookie. See also CTS-based sessions and Client-based sessions.

Session high availability

Capability that lets any AM server in a clustered deployment access shared, persistent information about users' sessions from the CTS token store. The user does not need to log in again unless the entire deployment goes down.

Session token

Unique identifier issued by AM after successful authentication. For a CTS-based sessions, the session token is used to track a principal's session.

Single log out (SLO)

Capability allowing a principal to end a session once, thereby ending her session across multiple applications.

Single sign-on (SSO)

Capability allowing a principal to authenticate once and gain access to multiple applications without authenticating again.

Site

Group of AM servers configured the same way, accessed through a load balancer layer. The load balancer handles failover to provide service-level availability.

The load balancer can also be used to protect AM services.

Standard metadata

Standard federation configuration information that you can share with other access management software.

Stateless Service

Stateless services do not store any data locally to the service. When the service requires data to perform any action, it requests it from a data store. For example, a stateless authentication service stores session state for logged-in users in a database. This way, any server in the deployment can recover the session from the database and service requests for any user.

All AM services are stateless unless otherwise specified. See also Client-based sessions and CTS-based sessions.

Subject

Entity that requests access to a resource

When an identity successfully authenticates, AM associates the identity with the Principal that distinguishes it from other identities. An identity can be associated with multiple principals.

Identity store

Data storage service holding principals' profiles; underlying storage can be an LDAP directory service or a custom `IdRepo` implementation.

Web Agent

Native library installed in a web server that acts as a policy enforcement point with policies based on web page URLs.