OpenAM Upgrade Guide

Version 11.0.0

Mark Craig
Vanessa Richie
Mike Jang

ForgeRock AS
33 New Montgomery St.,
Suite 1500
San Francisco, CA 94105, USA
+1 415-599-1100
www.forgerock.com

Abstract

This guide shows you how to upgrade OpenAM. OpenAM provides open source Authentication, Authorization, Entitlement, and Federation software.

Table of Contents

Preface

This guide describes how to upgrade OpenAM servers, policy agents, and tools.

1 Who Should Use this Guide

This guide is for anyone who needs to upgrade an OpenAM deployment. This guide assumes you are familiar with OpenAM installation and configuration, and that you are familiar with the current OpenAM deployment that you plan to upgrade.

You do not need to be an OpenAM wizard to learn something from this guide, though a background in access management and maintaining web application software can help. You do need some background in managing services on your operating systems and in your application servers. You can nevertheless get started with this guide, and then learn more as you go.

2 Formatting Conventions

Most examples in the documentation are created on GNU/Linux or Mac OS X. Where it is helpful to make a distinction between operating environments, examples for UNIX, GNU/Linux, Mac OS X, and so forth are labeled (UNIX). Mac OS X specific examples can be labeled (Mac OS X). Examples for Microsoft Windows can be labeled (Windows). To avoid repetition, however, file system directory names are often given only in UNIX format as in `/path/to/server`, even if the text applies to `C:\path\to\server` as well.

Absolute path names usually begin with the placeholder `/path/to/`. This path might translate to `/opt/`, `C:\Program Files\`, or somewhere else on your system.

Command line, terminal sessions are formatted as follows.

```
$ echo $JAVA_HOME
/path/to/jdk
```

Command output is sometimes formatted for narrower, more readable output even though formatting parameters are not shown in the command. In the following example, the query string parameter `_prettyPrint=true` is omitted.

```
$ curl https://bjensen:hifalutin@opendj.example.com:8443/users/newuser
{
  "_rev" : "000000005b337348",
  "schemas" : [ "urn:scim:schemas:core:1.0" ],
  "contactInformation" : {
    "telephoneNumber" : "+1 408 555 1212",
    "emailAddress" : "newuser@example.com"
  },
  "_id" : "newuser",
  "name" : {
    "familyName" : "New",
    "givenName" : "User"
  },
  "userName" : "newuser@example.com",
  "displayName" : "New User",
  "meta" : {
    "created" : "2014-06-03T09:58:27Z"
  },
  "manager" : [ {
    "_id" : "kvaughan",
    "displayName" : "Kirsten Vaughan"
  } ]
}
```

Program listings are formatted as follows.

```
class Test {
    public static void main(String [] args)  {
        System.out.println("This is a program listing.");
    }
}
```

3 Accessing Documentation Online

ForgeRock core documentation, such as what you are now reading, aims to be technically accurate and complete with respect to the software documented.

Core documentation therefore follows a three-phase review process designed to eliminate errors.

- Product managers and software architects review project documentation design with respect to the users' software lifecycle needs.
- Subject matter experts review proposed documentation changes for technical accuracy and completeness with respect to the corresponding software.
- Quality experts validate implemented documentation changes for technical validity with respect to the software, technical completeness with respect to the scope of the document, and usability for the expected audience.

The review process helps to ensure that documentation published for a ForgeRock release is technically accurate and complete.

Fully reviewed, published core documentation is available at http://docs.forgerock.org/. Use this documentation when working with a ForgeRock Enterprise release.

In-progress documentation can be found at each project site under the Developer Community projects page. Use this documentation when trying a nightly build.

The ForgeRock Community Wikis and provide additional, user-created information. We encourage you to join the community, so that you can update the Wikis, too.

4 Joining the ForgeRock Community

After you sign up to join the ForgeRock community, you can edit the Community Wikis, and also log bugs and feature requests in the issue tracker.

If you have a question regarding a project but cannot find an answer in the project documentation or Wiki, browse to the Developer Community page for the project, where you can find details on joining the project mailing lists, and find links to mailing list archives. You can also suggest updates to documentation through the ForgeRock docs mailing list.

The Community Wikis describe how to check out and build source code. Should you want to contribute a patch, test, or feature, or want to author part of the core documentation, first have a look on the ForgeRock site at how to get involved.

Chapter 1

About Upgrading OpenAM

This chapter covers common aspects of upgrading an OpenAM deployment, whether you are moving to a new maintenance release, upgrading to a new major release, or migrating from a legacy release to a newer OpenAM release.

Release levels, and how much change to expect in a maintenance, minor, or major release, are defined in the *Administration Guide* section, *ForgeRock Product Release Levels*. Release levels are identified by version number.

1.1 Planning the Upgrade

How much you must do to upgrade OpenAM software depends on the magnitude of the differences between the version you currently use and the new version.

- Maintenance releases have a limited effect on current functionality but contain necessary bug and security fixes. You should keep up to date with maintenance releases as the fixes are important and the risk of affecting service is minimal.
- When upgrading to a new major or minor release, always plan and test the changes before carrying out the upgrade in production. Make sure you read release notes for intervening versions with care, identifying any changes likely to affect your deployment, and then plan accordingly.
- These suggestions are true both for OpenAM server components, and also for policy agents.

To upgrade from OpenAM server 9.5 and later you can use the Upgrade Wizard. The OpenAM server Upgrade Wizard, added in OpenAM 10.0.0, appears

when you replace a deployed OpenAM server .war with a newer version and browse to the deployment URL. The Upgrade Wizard brings the OpenAM configuration, including the version number, up to date with the new version. The CLI counterpart of the Upgrade Wizard is openam-upgrade-tool-11.0.0.jar, which you install as described in *To Set Up Configuration Tools*.

For legacy releases, meaning OpenAM server version 9.0 and earlier including Sun Access Manager and OpenSSO releases, you must reinstall and configure OpenAM server rather than upgrade. Moving to a new release from a legacy release is therefore a migration, rather than a simple upgrade.

1.2 Best Practices for Upgrades

Be prepared before you begin an upgrade, even if the upgrade is for a maintenance release.

1.2.1 Route Around Servers During Downtime

Upgrading servers takes at least one of your OpenAM sites down while the server configurations are being brought up to date with the newer version. Plan for this site to be down, routing client applications to another site until the upgrade process is complete and you have validated the result. Make sure client application owners are well aware of the change, and let them know what to expect.

If you only have a single OpenAM site, make sure the downtime happens in a low usage window, and make sure you let client application owners plan accordingly.

During an upgrade you must restrict access to OpenAM Console: The Upgrade Wizard page does not require authorization; any user with access to OpenAM Console immediately after you deploy the new .war can therefore initiate the upgrade process.

1.2.2 Back Up the Deployment

Always back up your deployment before you upgrade, as you must be able to roll back should something go wrong during the upgrade process.

- Backing up your configuration as described in *Backing Up and Restoring OpenAM Configurations* is good for production environments.
- In preparation for upgrading OpenAM servers and their configurations, also take LDIF backups of the configuration store data in the directory servers. If possible, stop servers before upgrading and take a file system backup of the deployed servers and also of their configuration directories as well. This can make it easier to roll back from a failed upgrade.

For example, if you deploy OpenAM server in Apache Tomcat under `/openam`, you might take a file system backup of the following directories for each OpenAM server.

- `/path/to/tomcat/webapps/openam/`
- `~/openam/`
- `~/.openamcfg/`

- When upgrading web policy agents, take a file system backup of the policy agent installation and configuration directories.

  When upgrading Java EE policy agents, it can be easier to uninstall the new version and reinstall the old version than to restore from file system backup.

- When upgrading tools, keep copies of any tools scripts that you have edited for your deployment. Also back up any trust stores used to connect securely.

1.2.3 Apply Customization Before Upgrading

Before you upgrade OpenAM servers, prepare a .war file that contains any customizations you require.

Customizations include any changes you have made to the OpenAM server installation, such as the following.

- Plugin and extensions such as custom authentication modules, response providers, post authentication plugins, SAML 2.0 attribute mappers, and OAuth 2.0 scope implementations

  These are described in the *Developer's Guide*.

- Customized JSPs, redesigned login or service pages, additional CSS and visual content, and modified authentication module callback files

  These are described in the *Installation Guide*.

- Any changes to OpenAM classes

- Any changes or additional Java class libraries (such as .jar files in `WEB-INF/lib`

1.2.4 Plan for Rollback

Sometimes even a well-planned upgrade operation fails to go smoothly. In such cases, you need a plan to roll back smoothly to the pre-upgrade version.

For OpenAM servers, you can roll back by restoring from file system backup. If you use an external configuration directory service, restore the old configuration from LDIF before restarting the old servers.

For web policy agents, you can roll back by restoring from file system backup. If you used configuration only available to newer agents, restore the pre-upgrade configuration before restarting the old agents.

For Java EE policy agents, uninstall the newer agents and reinstall the older agents, including the old configurations.

Chapter 2

Upgrading OpenAM Servers

This chapter covers upgrade from OpenAM core server 9.5 or later to the current version. For other OpenAM components, see *Upgrading OpenAM Components*.

OpenAM server upgrade relies on the Upgrade Wizard to make the necessary changes to the configuration store. You must then restart OpenAM or the container in which it runs. Even a version number change requires that you run the Upgrade Wizard, so needing to run the Upgrade Wizard says nothing about the significance of the changes that have been made to OpenAM. You must run the Upgrade Wizard even for maintenance releases.

Make sure you try upgrading OpenAM in a test environment before applying the upgrade in your production environment.

Procedure 2.1. To Upgrade From OpenAM 9.5 or Later

Follow these steps to upgrade a site of OpenAM servers (version 9.5 or later). During the upgrade process, you must take the OpenAM servers in the site out of production, instead redirecting client application traffic elsewhere. This is required because upgrade involves making changes to OpenAM's configuration model. If the upgrade fails, you must be able to roll back before the configuration changes impact other sites.

1. Prepare your customized OpenAM server .war file.
2. Back up the deployment.
3. Route client application traffic to another site during the upgrade.

4. For servers in the site, stop OpenAM, or if necessary stop the container where OpenAM runs.

5. For servers in the site, deploy your customized server .war file.

   When you deploy the new .war file, you might have to delete working files left by the old installation. For example, if you deploy on Apache Tomcat, replacing `/path/to/tomcat/webapps/openam.war`, then also recursively delete the `/path/to/tomcat/webapps/openam/` and `/path/to/tomcat/work/Catalina/localhost/openam/` directories before restarting the server.

6. For servers in the site, restart OpenAM or the container where it runs.

7. For the first server in the site, follow the instructions in the Upgrade Wizard.

   Alternatively, you can use the `openam-upgrade-tool-11.0.0.jar` command-line tool to upgrade the server configuration. The procedure, *To Set Up Configuration Tools*, describes how to install the tool.

8. If you want to configure the upgraded system with a different directory service for the Core Token Service (CTS), read *Configuring the Core Token Service*.

9. Validate that the service is performing as expected.

10. Allow client application traffic to flow to the upgraded site.

Procedure 2.2. To Complete Upgrade from OpenAM 10.1.0 Xpress

When upgrading from OpenAM 10.1.0 Xpress, the upgrade tool does not change the Dashboard service LDAP schema, although the object IDs used in the Dashboard service LDAP schema definitions are not correct.

You can fix the object IDs manually using the OpenDJ **ldapmodify** command. The command is delivered with OpenDJ directory server.

1. Update the LDAP schema defined in the OpenDJ directory server where OpenAM stores its configuration.

   Make the change on one of the replicated OpenDJ configuration directory servers.

   The example command shown below uses the **ldapmodify** command delivered with the embedded OpenDJ configuration directory server for OpenAM with deployment URI `/openam`. When you use the embedded OpenDJ configuration directory server, the password for the `cn=Directory Manager` account is the same password used by `amadmin`.

The LDAP schema definitions are stored on the LDAP subentry with distinguished name `cn=schema`. You use the following LDIF format definitions to correct the object IDs in the definitions.

```
$ cd ~/openam/opends/bin
$ cat dash.ldif
dn: cn=schema
changetype: modify
delete: objectClasses
objectClasses: ( 1.3.6.1.4.1.1466.101.120.1433 NAME
  'forgerock-am-dashboard-service' AUXILIARY MAY (
   assignedDashboard ) X-ORIGIN 'Forgerock' )
-
delete: attributeTypes
attributeTypes: ( 1.3.6.1.4.1.36733.2.1.9.2.811 NAME ( 'assignedDashboard' )
  DESC 'Dashboard App registry' SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
  X-ORIGIN 'OpenAM' )
-
add: attributeTypes
attributeTypes: ( 1.3.6.1.4.1.36733.2.2.1.3.1 NAME ( 'assignedDashboard' )
  DESC 'Dashboard App registry' SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
  X-ORIGIN 'OpenAM' )
-
add: objectClasses
objectClasses: ( 1.3.6.1.4.1.36733.2.2.2.3.1 NAME
  'forgerock-am-dashboard-service' AUXILIARY MAY (
  assignedDashboard ) X-ORIGIN 'Forgerock' )

$ ./ldapmodify -p 50389 -D "cn=Directory Manager" -w password -f ./dash.ldif
Processing MODIFY request for cn=schema
MODIFY operation successful for DN cn=schema
```

2. If you want to configure the upgraded system with a different directory service for the Core Token Service (CTS), read *Configuring the Core Token Service*.

Chapter 3

Migrating Legacy Servers

Rather than upgrade legacy servers (OpenAM 9.0, OpenSSO and Sun Access Manager servers), you instead migrate from your existing deployment to a new deployment.

For complex legacy deployments, ForgeRock can assist you in the migration process. Send mail to info@forgerock.com for more information.

Procedure 3.1. To Upgrade A Legacy Deployment

1. Prepare your customized OpenAM server .war file.
2. Prepare a new deployment, installing servers from the new, customized .war file as described in the *Installation Guide*, starting with the instructions in *Installing OpenAM Core Services*.
3. After installation, configure the new servers in the same way as the old servers, adapting as necessary.

   You can use the **ssoadm do-batch** command to apply multiple changes with one command.
4. Validate that the new service is performing as expected.
5. Redirect client application traffic from the old deployment to the new deployment.

Chapter 4

Upgrading OpenAM Components

This chapter is concerned with upgrades for policy agents, OpenAM tools, and the OpenAM distributed authentication UI.

Procedure 4.1. To Upgrade Web Policy Agents

1. Back up the policy agent installation and configuration directories.

   Also back up the configuration if it is stored centrally in OpenAM.

2. Redirect client traffic away from the protected application.

3. Stop the web server where the policy agent is installed.

4. Extract the new files over the old installation.

5. Start the web server where the policy agent is installed.

   For new features, the policy agent uses the default configuration until you make changes.

6. Validate that the policy agent is performing as expected.
7. Allow client traffic to flow to the protected application.

Procedure 4.2. To Upgrade Java EE Policy Agents

1. Back up the policy agent installation and configuration directories.

   Also back up the configuration if it is stored centrally in OpenAM.
2. Redirect client traffic away from the protected application.
3. Uninstall the old policy agent.
4. Install the new policy agent.

   For new features, the policy agent uses the default configuration until you make changes.
5. Validate that the policy agent is performing as expected.
6. Allow client traffic to flow to the protected application.

Procedure 4.3. To Upgrade OpenAM Tools

Since OpenAM 10.1, the session tools are no longer needed. Upgrading other tools consists of installing new tools and customizing tools scripts as necessary.

1. Install new versions of the tools.
2. Apply any customizations you made to the scripts, referring to the old tools installation as necessary.
3. Once the new tools are working, you can delete the old tools.

Procedure 4.4. To Upgrade OpenAM Distributed Authentication Server

If you deployed the distributed authentication server (DAS) .war file, then you should upgrade the DAS when you upgrade other OpenAM servers.

1. Redirect client application traffic away from the server.
2. Stop the DAS or the container in which it runs.
3. Deploy the new DAS .war file.

   When you deploy the new .war file, you might have to delete working files left by the old installation.

4. Restart the DAS or the container in which it runs.
5. Validate that the DAS is working as expected.
6. Allow client application traffic to flow back to the server.