



Upgrade Guide

/ ForgeRock Access Management 6

Latest update: 6.0.0.7

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

Copyright © 2011-2018 ForgeRock AS.

Abstract

This guide shows you how to upgrade ForgeRock® Access Management. ForgeRock Access Management provides authentication, authorization, entitlement, and federation software.



This work is licensed under the Creative Commons Attribution-NonCommercial-NoDerivs 3.0 Unported License.

To view a copy of this license, visit <https://creativecommons.org/licenses/by-nc-nd/3.0/> or send a letter to Creative Commons, 444 Castro Street, Suite 900, Mountain View, California, 94041, USA.

ForgeRock® and ForgeRock Identity Platform™ are trademarks of ForgeRock Inc. or its subsidiaries in the U.S. and in other countries. Trademarks are the property of their respective owners.

UNLESS OTHERWISE MUTUALLY AGREED BY THE PARTIES IN WRITING, LICENSOR OFFERS THE WORK AS-IS AND MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND CONCERNING THE WORK, EXPRESS, IMPLIED, STATUTORY OR OTHERWISE, INCLUDING, WITHOUT LIMITATION, WARRANTIES OF TITLE, MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, NONINFRINGEMENT, OR THE ABSENCE OF LATENT OR OTHER DEFECTS, ACCURACY, OR THE PRESENCE OF ABSENCE OF ERRORS, WHETHER OR NOT DISCOVERABLE. SOME JURISDICTIONS DO NOT ALLOW THE EXCLUSION OF IMPLIED WARRANTIES, SO SUCH EXCLUSION MAY NOT APPLY TO YOU.

EXCEPT TO THE EXTENT REQUIRED BY APPLICABLE LAW, IN NO EVENT WILL LICENSOR BE LIABLE TO YOU ON ANY LEGAL THEORY FOR ANY SPECIAL, INCIDENTAL, CONSEQUENTIAL, PUNITIVE OR EXEMPLARY DAMAGES ARISING OUT OF THIS LICENSE OR THE USE OF THE WORK, EVEN IF LICENSOR HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

DejaVu Fonts

Bitstream Vera Fonts Copyright

Copyright (c) 2003 by Bitstream, Inc. All Rights Reserved. Bitstream Vera is a trademark of Bitstream, Inc.

Permission is hereby granted, free of charge, to any person obtaining a copy of the fonts accompanying this license ("Fonts") and associated documentation files (the "Font Software"), to reproduce and distribute the Font Software, including without limitation the rights to use, copy, merge, publish, distribute, and/or sell copies of the Font Software, and to permit persons to whom the Font Software is furnished to do so, subject to the following conditions:

The above copyright and trademark notices and this permission notice shall be included in all copies of one or more of the Font Software typefaces.

The Font Software may be modified, altered, or added to, and in particular the designs of glyphs or characters in the Fonts may be modified and additional glyphs or characters may be added to the Fonts, only if the fonts are renamed to names not containing either the words "Bitstream" or the word "Vera".

This License becomes null and void to the extent applicable to Fonts or Font Software that has been modified and is distributed under the "Bitstream Vera" names.

The Font Software may be sold as part of a larger software package but no copy of one or more of the Font Software typefaces may be sold by itself.

THE FONT SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OF COPYRIGHT, PATENT, TRADEMARK, OR OTHER RIGHT. IN NO EVENT SHALL BITSTREAM OR THE GNOME FOUNDATION BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, INCLUDING ANY GENERAL, SPECIAL, INDIRECT, INCIDENTAL, OR CONSEQUENTIAL DAMAGES, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF THE USE OR INABILITY TO USE THE FONT SOFTWARE OR FROM OTHER DEALINGS IN THE FONT SOFTWARE.

Except as contained in this notice, the names of Gnome, the Gnome Foundation, and Bitstream Inc., shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Font Software without prior written authorization from the Gnome Foundation or Bitstream Inc., respectively. For further information, contact: fonts at gnome dot org.

Arev Fonts Copyright

Copyright (c) 2006 by Tavmjong Bah. All Rights Reserved.

Permission is hereby granted, free of charge, to any person obtaining a copy of the fonts accompanying this license ("Fonts") and associated documentation files (the "Font Software"), to reproduce and distribute the modifications to the Bitstream Vera Font Software, including without limitation the rights to use, copy, merge, publish, distribute, and/or sell copies of the Font Software, and to permit persons to whom the Font Software is furnished to do so, subject to the following conditions:

The above copyright and trademark notices and this permission notice shall be included in all copies of one or more of the Font Software typefaces.

The Font Software may be modified, altered, or added to, and in particular the designs of glyphs or characters in the Fonts may be modified and additional glyphs or characters may be added to the Fonts, only if the fonts are renamed to names not containing either the words "Tavmjong Bah" or the word "Arev".

This License becomes null and void to the extent applicable to Fonts or Font Software that has been modified and is distributed under the "Tavmjong Bah Arev" names.

The Font Software may be sold as part of a larger software package but no copy of one or more of the Font Software typefaces may be sold by itself.

THE FONT SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OF COPYRIGHT, PATENT, TRADEMARK, OR OTHER RIGHT. IN NO EVENT SHALL TAVMJONG BAH BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, INCLUDING ANY GENERAL, SPECIAL, INDIRECT, INCIDENTAL, OR CONSEQUENTIAL DAMAGES, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF THE USE OR INABILITY TO USE THE FONT SOFTWARE OR FROM OTHER DEALINGS IN THE FONT SOFTWARE.

Except as contained in this notice, the name of Tavmjong Bah shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Font Software without prior written authorization from Tavmjong Bah. For further information, contact: tavmjong @ free . fr.

FontAwesome Copyright

Copyright (c) 2017 by Dave Gandy, <http://fontawesome.io>.

This Font Software is licensed under the SIL Open Font License, Version 1.1. This license is available with a FAQ at: <http://scripts.sil.org/OFL>.

Table of Contents

Preface	iv
1. About Upgrading	1
1.1. Supported Upgrade Paths	1
1.2. Planning the Upgrade	2
1.3. Best Practices for Upgrades	2
2. Upgrading Servers	6
3. Upgrading Components	12
3.1. Upgrading Web Agents	12
3.2. Upgrading Java Agents	13
3.3. Upgrading Tools	14
3.4. Upgrading User Self-Services	14
4. Migrating Legacy Servers	17
5. Reference	18
5.1. Command-Line Tool Reference	18
A. Getting Support	21
A.1. Accessing Documentation Online	21
A.2. Using the ForgeRock.org Site	21
A.3. Getting Support and Contacting ForgeRock	22
Glossary	23

Preface

The Upgrade Guide describes how to upgrade ForgeRock Access Management servers, web and Java agents, and tools.

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

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

About Upgrading

This chapter covers common aspects of upgrading an AM 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 AM release.

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

1.1. Supported Upgrade Paths

The following table contains information about the supported upgrade paths to AM 6.0.0.7:

Upgrade Paths

Version	Upgrade Supported?
AM 6.0.x	✓
AM 5.x	✓ ^a
OpenAM 13.x.x	✓
OpenAM 12.0.x	✗

^a

Caution

AM is incompatible with SSO session tokens from OpenAM.

Storage and processing of sessions changed in AM 5: CTS-based (stateful) and client-based (stateless) sessions created by earlier versions of OpenAM are not supported.

After upgrading from an earlier version, any existing SSO tokens created by that version will become invalid, and users will need to reauthenticate.

In mixed version deployments, earlier versions of OpenAM will not be able to read or process SSO session tokens created by AM 5 or later.

This incompatibility only affects SSO session tokens. OAuth 2.0 and OpenID Connect 1.0 tokens are interoperable between versions.

Note

Upgrading between Enterprise and OEM versions is not supported.

For more information, see [Checking your product versions are supported in the ForgeRock Knowledge Base](#).

1.2. Planning the Upgrade

How much you must do to upgrade AM 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 AM server components, and also for web and Java agents.

To upgrade from an AM server, use the Upgrade wizard. The AM server Upgrade wizard appears when you replace a deployed AM server `.war` file with a newer version and browse to the deployment URL. The Upgrade wizard brings the AM configuration, including the version number, up to date with the new version. The CLI counterpart of the Upgrade wizard is `openam-upgrade-tool-14.1.1.5.jar`, which you install as described in "Setting up Configuration Tools" in the *Installation Guide*.

1.3. Best Practices for Upgrades

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

1.3.1. Route Around Servers During Downtime

Upgrading servers takes at least one of your AM 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 AM 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 the AM console: The Upgrade Wizard page does not require authorization; any user with access to the AM console immediately after you deploy the new .war can therefore initiate the upgrade process.

1.3.2. 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 Configurations" in the *Setup and Maintenance Guide* is good for production environments.
- In preparation for upgrading AM 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 AM server in Apache Tomcat under `/openam`, you might take a file system backup of the following directories for each AM server.

- `/path/to/tomcat/webapps/openam/`
- `~/openam/`
- `~/openamcfg/`
- When upgrading web agents, take a file system backup of the web or Java agent installation and configuration directories.

When upgrading Java 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.

1.3.3. Review REST API Versions Before Upgrading

Upgrading `$[am.abbr]` may update the default API version of several AM endpoints. After an upgrade, your applications may experience issues connecting to endpoints if they do not specify API version information in REST calls.

By default, an upgraded AM instance responds to REST calls that do not specify version information with the oldest version available for the endpoint. However, the oldest supported version may not be the one required by the application, as API versions become deprecated or unsupported.

To avoid version conflicts between application calls and REST endpoint APIs, consider specifying the protocol and resource version required by the application in the `Accept-API-Version` header when

making requests to REST endpoints. For more information, see "Specifying an Explicit REST API Version" in the *Development Guide*.

Important

Starting in version 6, AM includes a CSRF protection filter that is enabled by default. REST requests other than GET, HEAD, and OPTIONS made to endpoints under the `json/` root will return 403 Forbidden messages unless one of the `X-Requested-With` or `Accept-API-Version` headers are added to it.

For more information, see "Cross-Site Request Forgery (CSRF) Protection" in the *Setup and Maintenance Guide*.

You can configure AM's default REST API behavior. For more information, see "Configuring the Default REST API Version for a Deployment" in the *Development Guide*.

1.3.4. Review Directory Services Certificates Before Upgrading

Before upgrading, review that the certificates used to establish secure connections between AM and the DS stores.

If the FQDN value from the `subject` field of a non-wildcard certificate does not match the FQDN obtained from DNS for the DS instance, AM will not be able to establish a secure connection with DS. Additionally, if the DS instance responds to multiple FQDNs, they must be specified in the certificate as well.

This step is critical for the configuration store. If AM cannot establish communication with the configuration store, it will fail to start up.

For more information about setting up DS server certificates, see *Setting Up Server Certificates* in the ForgeRock Directory Services Administration Guide.

1.3.5. Apply Customization Before Upgrading

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

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

- Plugins and extensions such as custom authentication modules, response attribute providers, post authentication plugins, SAML v2.0 attribute mappers, and OAuth 2.0 scope implementations.

These are described in the *Development Guide*.

Important

If you are upgrading from OpenAM 12.x and you have custom authentication modules, you must upgrade their service definitions to contain `resourceName` attributes in the `Schema` and `SubSchema` elements.

For an example of a service definition compatible with AM 6, see "The Sample Auth Service Configuration" in the *Authentication and Single Sign-On Guide*.

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

These are described in the *UI Customization Guide*.

- Any changes to AM classes.
- Any changes or additional Java class libraries (such as .jar files in `WEB-INF/Lib`).

1.3.6. 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 AM 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 more information, see "Backing Up and Restoring Configurations" in the *Setup and Maintenance Guide*.

For web 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 agents, uninstall the newer agents and reinstall the older agents, including the old configurations.

Chapter 2

Upgrading Servers

This chapter covers upgrade from core server 12.0.0 or later to the current version. For other components, see "*Upgrading Components*".

AM server upgrade relies on the Upgrade Wizard to make the necessary changes to the configuration store. You must then restart AM 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 AM. You must run the Upgrade Wizard even for maintenance releases.

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

- "To Upgrade From a Supported Version"
- "To Complete Upgrade from OpenAM 13.0.x"

Important

If you are upgrading from an unsupported version of AM to a later version, you must first upgrade to a supported version. In some cases, you may need to upgrade again depending on the upgrade path. For more information about supported upgrade paths, see "Supported Upgrade Paths" in the *Release Notes*.

To Upgrade From a Supported Version

Follow these steps to upgrade a site of servers. For information on the versions that are supported for upgrade, see "Supported Upgrade Paths" in the *Release Notes*.

During the upgrade process, you must take the AM servers in the site out of production, instead redirecting client application traffic elsewhere. This is required because upgrade involves making changes to AM's configuration model. If the upgrade fails, you must be able to roll back before the configuration changes impact other sites.

Important

Do *not* perform an upgrade by deploying the new version and then importing an existing configuration by running the `ssoadm import-svc-config` command. Importing an outdated configuration can result in a corrupted installation.

1. Prepare your customized AM server `.war` file.

2. **Back up your deployment.**
3. Route client application traffic to another site during the upgrade.
4. Stop AM or the container where it runs.
5. 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. Restart AM or the container where it runs.
7. (Optional) You must update the user store XML schema if you are upgrading from OpenAM or a version of AM earlier than 6 and any of the following statements are true:
 - You have configured knowledge-based (KBA) user self-service questions.
 - You have not configured User Self-Service yet, but you added the `user_store_type_kba.ldif` schema to your external user data store when you configured it.

For more information about LDIFs, see "*Supported LDIF Files*" in the *Installation Guide*.

- You are using the embedded DS instance as a user store.

To update the user store schema, perform the following steps:

- a. Change directories to the path where you have deployed the `openam.war` file. For example, `/path/to/tomcat/webapps/openam`.
- b. Locate the following files in the `WEB-INF/template/ldif/opensj` path:
 - `opensj_add_kba_attempts.ldif`
 - `opensj_update_aci_kba_attempts.ldif`

Note

If your user data store is not DS, use these files as examples to create files suitable for your environment.

- c. Open the `opensj_update_aci_kba_attempts.ldif` file and replace `@SM_CONFIG_ROOT_SUFFIX` with the base DN defined during the DS installation procedure (for example, `dc=userstore,dc=example,dc=com`).
- d. Update the user data store schema with the two files. For example, to update a DS instance, run the following command:

```
$ ldapmodify \
--hostname opendj.example.com \
--port 1389 \
--bindDN "cn=Directory Manager" \
opendj_add_kba_attempts.ldif opendj_update_aci_kba_attempts.ldif
Password for user 'cn=Directory Manager':

# Processing MODIFY request for cn=schema
# MODIFY operation successful for DN cn=schema
# Processing MODIFY request for dc=userstore,dc=example,dc=com
# MODIFY operation successful for DN dc=userstore,dc=example,dc=com
```

Note that you will need to update the user store schema again in a later step whether you performed this step or not.

8. To upgrade the data in the configuration store, perform one of the following actions in one of the servers in the site:
 - Navigate to the AM URL, for example <https://openam.example.com:443/openam>, and follow the instructions in the Upgrade Wizard for an interactive upgrade.
 - Use the `openam-upgrade-tool-14.1.1.5.jar` tool for an unattended upgrade:
 1. Install the `openam-upgrade-tool-14.1.1.5.jar` tool as described in "Setting up Configuration Tools" in the *Installation Guide*. A `sampleupgrade` file will be expanded in the directory where you install the tool.
 2. Create a configuration file for the `openam-upgrade-tool-14.1.1.5.jar`. You can use the `sampleupgrade` file as a template to create a configuration file, for example `upgrade.properties`.

An upgrade configuration file may resemble the following:

```
$ grep -v "^#" upgrade.properties
SERVER_URL=http://openam.example.com:8080
DEPLOYMENT_URI=/openam
ACCEPT_LICENSES=true
```

3. Upgrade AM by using the tool with the properties file following this example:

```
$ java -jar openam-upgrade-tool-14.1.1.5.jar --file upgrade.properties
Writing Backup; Done.
Upgrading Services
New service iPlanetAMAuthPersistentCookieService; Done.
New service iPlanetAMAuthOpenIdConnectService; Done.
New service OAuth2Provider; Done.
New service iPlanetAMAuthDevicePrintModuleService; Done.
New service crestPolicyService; Done.
New service RestSecurity; Done.
New service MailServer; Done.
New service dashboardService; Done.
New service iPlanetAMAuthOATHService; Done.
Add Organization schema to sunFAMSAML2Configuration; Done.
Upgrade sunAMAuthHOTPSservice; Done.
Upgrade sunAMAuthADService; Done.
Upgrade sunAMAuthOAuthService; Done.
```

```
Upgrade iPlanetAMAuthCertService; Done.
Upgrade sunIdentityRepositoryService; Done.
Upgrade iPlanetAMPASSWORDResetService; Done.
Upgrade iPlanetAMSessionService; Done.
Upgrade iPlanetAMAuthService; Done.
Upgrade iPlanetAMAuthLDAPService; Done.
Upgrade sunAMAuthDataStoreService; Done.
Upgrade AgentService; Done.
New sub schema sunIdentityRepositoryService; Done.
New sub schema AgentService; Done.
Delete service sunFAMLibertyInteractionService; Done.
Delete service sunFAMLibertySecurityService; Done.
Creating entitlement application type crestPolicyService; Done.
Creating entitlement application crestPolicyService; Done.
Re-enabling Generic LDAPv3 Data Store; Done.
Upgrading data store embedded; Done.
Updating Platform Properties; Done.
Writing Upgrade Log; Done.

Upgrade Complete.
```

For additional information about the command-line tool, see the reference documentation for `upgrade.jar(1)` in the *Reference*.

4. Restart AM or the container where it runs.
9. (Optional) If you installed AM using an external directory server as the configuration store, add an access control instruction (ACI) to the external directory to give the AM administrative user server-side sorting privileges.

The ACI should be similar to the following:

```
aci: (targetcontrol="1.2.840.113556.1.4.473")(version 3.0;
acl "Allow server-side sorting"; allow (read)
(userdn = "ldap:///uid=openam,ou=admins,dc=example,dc=com");)
```

See "Preparing an External Configuration Data Store" in the *Installation Guide* for more information about using an external directory server as the AM configuration store.

10. (Optional) If you installed AM using an external directory server as the user store, update the user store schema as follows:
 - a. Log into AM.
 - b. Navigate to *Realm Name* > *Datastores* > *External User Store*.
 - c. Click Load Schema before saving, and then click Save to apply your changes.
 - d. If you have additional external user stores, repeat the previous steps for each user store.
11. (Optional) If you want to configure the upgraded system for the Core Token Service (CTS), you need to import two schema files for proper CTS operation:

- `cts-add-multivalue.ldif`
- `cts-add-multivalue-indices.ldif`

For more information, read "*Supported Scripts*" in the *Installation Guide*. For a list of supported directory services, see the "*Data Store Requirements*" in the *Release Notes*.

12. Referral policies are not supported in AM 6. If your deployment has referral policies, the following warning message will appear when you upgrade your server to AM 6:

```
Referrals found that require removing
```

Specifically, if your deployment has policies in a sub-level realm that uses a referral in the top level realm, the upgrade process will fail. You must manually delete referrals and policies from your configuration after making note of your current policies. Then, run the upgrade.

Note

Simply disabling the referrals and running the upgrade will also result in a failure. You must delete the referrals and policies manually.

After upgrading to AM 6, you are responsible for reconfiguring AM, so that policy evaluation that previously depended upon referrals continues to function correctly. You might need to take one or both of the following actions:

- Reconfiguring your web or Java agent with the realm and policy set that contain policies to be evaluated when that agent requests a policy decision from AM. For more information about configuring an agent with a realm and policy set, see "*Working With Realms and Access Management Agents*" in the *Setup and Maintenance Guide*.
 - Copy or move a policy or a group of policies. AM 6 has REST API endpoints that let you copy and move policies. This functionality might be helpful when migrating away from policy deployments that use referral policies. For more information about the REST endpoints that let you copy and move policies, see "*Copying and Moving Policies*" in the *Authorization Guide*.
13. If you have post authentication plugins that expect state to be maintained by AM between login and logout, you must rewrite and redeploy them.

In versions prior to AM 5, the Keep Authentication Module Objects for Logout Processing option was available in the Core Authentication module. This option, when enabled, directed AM to maintain state information in server memory throughout a session's duration for post authentication plugin module instances. When logout was triggered, AM invoked the same post authentication plugin module instance, with state information intact. Therefore, developers could access module state stored at login when users logged out.

AM 6 does not maintain state in post authentication plugins between login and logout. Post authentication plugins that rely on module state being maintained in AM's memory between login and logout must be rewritten. You can store any information that you want to save between login

and logout in a session property. AM stores session properties in the CTS token store after login, and retrieves them from the token store as part of the logout process.

To set a session property, call the `setProperty` method on an `SSOToken` object as demonstrated by the post authentication plugin sample code in "Building Your Sample Post-Authentication Plugin" in the *Authentication and Single Sign-On Guide*.

14. Validate that the service is performing as expected.
15. Allow client application traffic to flow to the upgraded site.

To Complete Upgrade from OpenAM 13.0.x

If you configured one or more JDBC audit event handlers in OpenAM 13.0.x, make the following changes to the audit tables' schema:

1. Run the following command on Oracle databases that support AM audit event handlers:

```
ALTER TABLE am_auditaccess ADD (response_detail CLOB NULL);
```

This command adds the `response_detail` column to the `am_auditaccess` table.

2. Run the following commands on MySQL databases that support AM audit event handlers:

```
ALTER TABLE audit.am_auditconfig CHANGE COLUMN configobjectid objectid VARCHAR(255);  
ALTER TABLE audit.am_auditaccess ADD COLUMN response_detail TEXT NULL;
```

The commands change the name of the `configobjectid` column in the `am_auditconfig` table to `objectid` and add the `response_detail` column to the `am_auditaccess` table.

3. If you use databases other than Oracle or MySQL to support AM audit event handlers, review their schema.

If the `am_auditconfig` table has a column named `configobjectid`, change that column's name to `objectid`.

If the `am_auditaccess` table does not have a column named `response_detail`, add that column to the table's schema.

Chapter 3

Upgrading Components

This chapter concerns upgrading web or Java agents, tools, and services.

- "Upgrading Web Agents"
- "Upgrading Java Agents"
- "Upgrading Tools"
- "Upgrading User Self-Services"

3.1. Upgrading Web Agents

Follow the procedure in this section to upgrade web agents to a version earlier than 5. To upgrade web agents to 5, see *ForgeRock Access Management Web Agents User Guide*

Important

AM 6 requires Web Agents 5.x to be in version 5.0.1 or higher.

To Upgrade Web Agents

1. Refer to the *ForgeRock Access Management Web Agents Release Notes* of the version to which you are upgrading for information about changes and new functionality.
2. Back up the web agent installation and configuration directories. For example:

```
cp -r /path/to/web_agents/apache24_agent /path/to/backup
cp -r /path/to/apache/httpd/conf /path/to/backup
```

If the configuration is stored centrally in AM, back it up as described in "Backing Up and Restoring Configurations" in the *Setup and Maintenance Guide*.

3. Redirect client traffic away from the protected application.
4. Stop the web server where the web agent is installed.
5. Remove the old web agent.

For example, to remove an old web agent installed in Apache HTTP server, see *Removing the Apache Web Agent* in the *ForgeRock Access Management Web Agents Guide*. If the uninstall

process has changed, refer to the version of the *ForgeRock Access Management Web Agents Guide* that corresponds to your web agent.

6. Install the new web agent.

For example, to install the new agent in Apache HTTP server, see [Installing the Apache Web Agent](#) in the *ForgeRock Access Management Web Agents User Guide*.

7. Start the webserver where the agent is installed.
8. Validate that the agent is performing as expected.

For example, navigate to a protected page on the web site and confirm whether you can access it according to your configuration.

9. Allow client traffic to flow to the protected application.

3.2. Upgrading Java Agents

Follow the procedure in this section to upgrade Java agents to a version earlier than 5. To upgrade Java agents to 5, see the *ForgeRock Access Management Java Agents User Guide*

To Upgrade Java Agents

1. Refer to the *ForgeRock Access Management Java Agents Release Notes* of the version to which you are upgrading for information about changes and new web agent functionality.
2. Back up the agent installation and configuration directories. For example:

```
cp -r /path/to/j2ee_agents/tomcat_v7_agent /path/to/backup
cp -r /path/to/tomcat/webapps/agentapp /path/to/backup
```

If the configuration is stored centrally in AM, back it up as described in "Backing Up and Restoring Configurations" in the *Setup and Maintenance Guide*.

3. Redirect client traffic away from the protected application.
4. Stop the container where the agent is installed.
5. Remove the old Java agent.

For example, to remove an old agent installed in Apache Tomcat, see [Removing the Tomcat Java Agent](#) in the *ForgeRock Access Management Java Agents User Guide*. If the uninstall process has changed, refer to the version of the *Java Agents User Guide* that corresponds to your web agent.

6. Install the new Java agent.

For example, to install a Java agent in Apache Tomcat, see [Installing the Tomcat Java Agent](#) in the *ForgeRock Access Management Java Agents User Guide*.

7. Start the container where the agent is installed.
8. Validate that the agent is performing as expected.

For example, navigate to a protected page on the web site and confirm whether you can access it according to your configuration.

9. Allow client traffic to flow to the protected application.

3.3. Upgrading Tools

To upgrade the tools, perform the following procedure:

To Upgrade Tools

1. Install the new version of the tools as described in "Installing and Using the Tools" in the *Installation Guide*.
2. Once the new tools are working, you can delete the old tools.

3.4. Upgrading User Self-Services

This section covers upgrading user self-service features.

3.4.1. Upgrading the Keystore for User Self-Service

AM's key management system allows the user self-service feature to successfully operate in a multi-instance server deployment behind a load balancer.

When upgrading from a version previous to OpenAM 13.5, AM deploys a JCEKS keystore that includes demo user self-service keys. This keystore is not configured as the default keystore after the upgrade because your existing deployment might depend on the JKS keystore. For example, you might have deployed SAML v2.0 using key aliases in the JKS keystore.

To help you decide whether to enable a JCEKS keystore after upgrading to AM 6, see the following table:

User Self-Service Feature Upgrade

Upgrading from:	Enabling JCEKS required?
Versions prior to OpenAM 13.0	No
OpenAM 13.0 with the REST-based user self-service feature disabled	No

Upgrading from:	Enabling JCEKS required?
OpenAM 13.0 with the legacy user self-service feature enabled	No
OpenAM 13.0 with the REST-based user self-service feature enabled	Yes
OpenAM 13.5 with the REST-based user self-service feature enabled	It is already enabled.

You should not use the demo user self-service keys included in the JCEKS keystore for production purposes. Instead, create new key aliases for user self-service and configure them in AM. When moving your keystore from JKS to JCEKS, you must also review your existing use of keys in AM, and add existing keys available in the JKS keystore to the JCEKS keystore. For example, if you have a SAML v2.0 deployment that uses keys in AM's JKS keystore, you need to add the keys to the JCEKS keystore.

See the following sections for details:

- For more information about keystores in AM, how to configure a JCEKS keystore, and how to create new user self-service keys, see "*Setting Up Keys and Keystores*" in the *Setup and Maintenance Guide*.
- For more information about configuring user self-service keys in AM, see "*Creating a User Self-Service Service Instance*" in the *User Self-Service Guide*.

3.4.2. Upgrading User Self-Service in Subrealms

AM 6 alters the method for specifying the realm in URLs. Upgrading from a previous version which has user self-service enabled in a subrealm requires that this new method is applied to the URLs used in confirmation emails, as follows:

To Upgrade User Self-service in a Subrealm

1. Log in to the AM console of the upgraded instance as an administrator, for example `amAdmin`.
2. Navigate to Realms > *Subrealm Name* > Services > User Self-Service, and then click the Advanced Configuration tab.

Note that to view the Advanced Configuration tab you may need to click the small downwards-pointing triangle icon.

3. On the Advanced Configuration tab, alter the following properties to include a `realm` parameter, as in the following examples:

User Registration Confirmation Email URL

`http://openam.example.com:8080/openam/XUI/?realm=${realm}#register/`

Forgotten Password Confirmation Email URL

`http://openam.example.com:8080/openam/XUI/?realm=${realm}#passwordReset/`

4. Save your changes.

A clean install of AM will include a `realm` parameter in these properties by default.

Chapter 4

Migrating Legacy Servers

Rather than upgrade legacy servers (running OpenSSO or Sun Access Manager, or an OpenAM or AM version that is no longer supported), you instead manually 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.

To Upgrade A Legacy Deployment

1. Prepare your customized AM server `.war` file.
2. Prepare a new deployment, installing servers from the new, customized `.war` file, starting with the instructions in "*Installing and Starting Servers*" in the *Installation Guide*.
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 5

Reference

5.1. Command-Line Tool Reference

Name

upgrade.jar — upgrade AM using a configuration file

Synopsis

```
upgrade.jar {options}
```

Description

This executable jar file, `openam-upgrade-tool-14.1.1.5.jar`, lets you perform a silent upgrade on a deployed AM server by applying settings from a configuration file or using arguments. This capability allows you to include the `upgrade.jar` from a command line or in an upgrade script.

Options

The following options are supported.

-f | --file *configuration-file*

Upgrade a deployed AM web application archive using the specified configuration file. Upgrade configuration files are described in the sections below. Also, you can specify the system properties on the command line, instead of using the configuration file. See Example 2 below.

--acceptLicense

Auto-accept the software license agreement and suppress the display of the licence acceptance screen to the user. If the configuration file contains the `ACCEPT_LICENSES` property, it will have precedence over the command-line option.

-? | --help

Display the usage message.

Upgrade Configuration File

Base your configuration on the `sampleupgrade` file delivered with AM, and using the hints in this section, or the comments included in the file.

Upgrade Properties

SERVER_URL

URL to the web container where AM runs, such as `http://openam.example.com:8080`.

DEPLOYMENT_URI

URI where AM is deployed on the web container, such as `/openam`.

ACCEPT_LICENSES

Optional boolean property that can be set to always auto-accept the software license agreement and suppress displaying the license acceptance screen to the user. A value of `true` auto-accepts the license; any other value will be assumed to equal `false`, resulting in the presentation of the license. Default value is `false`. This property takes precedence over the `--acceptLicense` option, which can also be passed in to the application with the `openam-upgrade-tool-14.1.1.5.jar` file.

Examples

The following example shows a configuration file and the commands to upgrade a server using the `upgrade.jar`. The configuration file is saved as `/tmp/upgrade.txt`.

```
SERVER_URL=http://openam.example.com:8080
DEPLOYMENT_URI=/openam
ACCEPT_LICENSES=true
```

```
$JAVA_HOME/bin/java -jar ~/openam/tools/openam-upgrade-tool-14.1.1.5.jar \
-f /tmp/upgrade.txt
```

The following example shows how to specify system properties with the `upgrade.jar`.

```
SERVER_URL=http://openam.example.com:8080
DEPLOYMENT_URI=/openam
ACCEPT_LICENSES=true
```

```
$JAVA_HOME/bin/java -jar ~/openam/tools/openam-upgrade-tool-14.1.1.5.jar \
-DSERVER_URL=http://openam.example.com:8080 -DDEPLOYMENT_URI=/openam
```

The following example shows the use of the `--acceptLicense` option with the `upgrade.jar`.

```
SERVER_URL=http://openam.example.com:8080
DEPLOYMENT_URI=/openam
```

```
$JAVA_HOME/bin/java -jar ~/openam/tools/openam-upgrade-tool-14.1.1.5.jar \
-DSERVER_URL=http://openam.example.com:8080 -DDEPLOYMENT_URI=/openam \
--acceptLicense
```

Appendix A. Getting Support

For more information or resources about AM and ForgeRock Support, see the following sections:

A.1. Accessing Documentation Online

ForgeRock publishes comprehensive documentation online:

- The ForgeRock Knowledge Base offers a large and increasing number of up-to-date, practical articles that help you deploy and manage ForgeRock software.

While many articles are visible to community members, ForgeRock customers have access to much more, including advanced information for customers using ForgeRock software in a mission-critical capacity.

- ForgeRock product documentation, such as this document, aims to be technically accurate and complete with respect to the software documented. It is visible to everyone and covers all product features and examples of how to use them.

A.2. Using the ForgeRock.org Site

The [ForgeRock.org](https://forgerock.org) site has links to source code for ForgeRock open source software, as well as links to the ForgeRock forums and technical blogs.

If you are a *ForgeRock customer*, raise a support ticket instead of using the forums. ForgeRock support professionals will get in touch to help you.

A.3. Getting Support and Contacting ForgeRock

ForgeRock provides support services, professional services, training through ForgeRock University, and partner services to assist you in setting up and maintaining your deployments. For a general overview of these services, see <https://www.forgerock.com>.

ForgeRock has staff members around the globe who support our international customers and partners. For details, visit <https://www.forgerock.com>, or send an email to ForgeRock at info@forgerock.com.

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	<p>In general terms, a service exposing protected resources.</p> <p>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.</p>
Application type	<p>Application types act as templates for creating policy applications.</p> <p>Application types define a preset list of actions and functional logic, such as policy lookup and resource comparator logic.</p>

	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 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.

	<p>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.</p>
Conditions	<p>Defined as part of policies, these determine the circumstances under which which a policy applies.</p> <p>Environmental conditions reflect circumstances like the client IP address, time of day, how the subject authenticated, or the authentication level achieved.</p> <p>Subject conditions reflect characteristics of the subject like whether the subject authenticated, the identity of the subject, or claims in the subject's JWT.</p>
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 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	<p>Represents an entity that has been authenticated (such as a user, a device, or an application), and thus is distinguished from other entities.</p> <p>When a Subject successfully authenticates, AM associates the Subject with the Principal.</p>
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 user data 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	<p>Something a user can access over the network such as a web page.</p> <p>Defined as part of policies, these can include wildcards in order to match multiple actual resources.</p>
Resource owner	<p>In OAuth 2.0, entity who can authorize access to protected web resources, such as an end user.</p>
Resource server	<p>In OAuth 2.0, server hosting protected web resources, capable of handling access tokens to respond to requests for such resources.</p>
Response attributes	<p>Defined as part of policies, these allow AM to return additional information in the form of "attributes" with the response to a policy decision.</p>
Role based access control (RBAC)	<p>Access control that is based on whether a user has been granted a set of permissions (a role).</p>
Security Assertion Markup Language (SAML)	<p>Standard, XML-based language for exchanging authentication and authorization data between identity providers and service providers.</p>
Service provider (SP)	<p>Entity that consumes assertions about a principal (and provides a service that the principal is trying to access).</p>
Authentication Session	<p>The interval while the user or entity is authenticating to AM.</p>
Session	<p>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.</p>
Session high availability	<p>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.</p>
Session token	<p>Unique identifier issued by AM after successful authentication. For a CTS-based sessions, the session token is used to track a principal's session.</p>

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	<p>Group of AM servers configured the same way, accessed through a load balancer layer. The load balancer handles failover to provide service-level availability.</p> <p>The load balancer can also be used to protect AM services.</p>
Standard metadata	Standard federation configuration information that you can share with other access management software.
Stateless Service	<p>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.</p> <p>All AM services are stateless unless otherwise specified. See also Client-based sessions and CTS-based sessions.</p>
Subject	<p>Entity that requests access to a resource</p> <p>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.</p>
User data 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.